ISSN 1831-9424

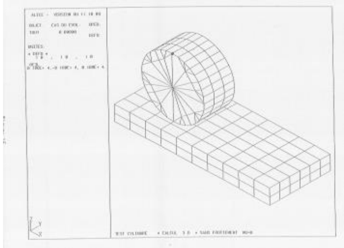
European
Commission

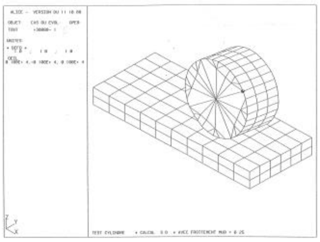
JRC Technical Report

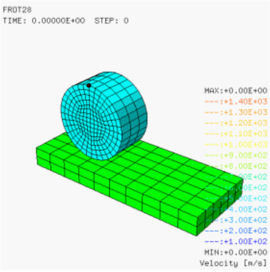
Treatment of friction in the PINB and GPIN contact-impact models of EUROPLEXUS

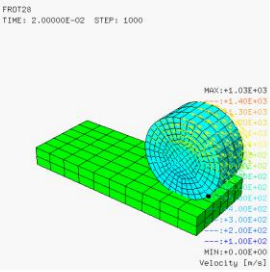
Casadei, F., Valsamos G., Larcher M.

2023









Joint
Research
Centre

EUR 31616 EN

This publication is a Technical report by the Joint Research Centre (JRC), the European Commission's science and knowledge service. It aims to provide evidence-based scientific support to the European policymaking process. The contents of this publication do not necessarily reflect the position or opinion of the European Commission. Neither the European Commission nor any person acting on behalf of the Commission is responsible for the use that might be made of this publication. For information on the methodology and quality underlying the data used in this publication for which the source is neither Eurostat nor other Commission services, users should contact the referenced source. The designations employed and the presentation of material on the maps do not imply the expression of any opinion whatsoever on the part of the European Union concerning the legal status of any country, territory, city or area or of its authorities, or concerning the delimitation of its frontiers or boundaries.

Contact information
Name: Martin Larcher
Address: JRC Ispra
Email: martin.larcher@ec.europa.eu

EU Science Hub
https://joint-research-centre.ec.europa.eu

JRC134229

EUR 31616 EN

PDF ISBN 978-92-68-06071-1 ISSN 1831-9424 doi:10.2760/952517 KJ-NA-31-616-EN-N

Luxembourg: Publications Office of the European Union, 2023

© European Union, 2023

The reuse policy of the European Commission documents is implemented by the Commission Decision 2011/833/EU of 12 December 2011 on the reuse of Commission documents (OJ L 330, 14.12.2011, p. 39). Unless otherwise noted, the reuse of this document is authorised under the Creative Commons Attribution 4.0 International (CC BY 4.0) licence (https://creativecommons.org/licenses/by/4.0/). This means that reuse is allowed provided appropriate credit is given and any changes are indicated.

For any use or reproduction of photos or other material that is not owned by the European Union permission must be sought directly from the copyright holders.

How to cite this report: Casadei, F., Valsamos G., Larcher M., *Treatment of friction in the PINB and GPIN contact-impact models of EUROPLEXUS*, Publications Office of the European Union, Luxembourg, 2023, doi:10.2760/952517, JRC134229.

Treatment of friction
in the PINB and GPIN contact-impact models
of EUROPLEXUS

F. Casadei[1,3], G. Valsamos[2], and M. Larcher[3]

[1]*Active Senior*
[2]*European Commission, Joint Research Centre*
Directorate for Space, Security and Migration
Technology Innovation in Security Unit
[3]*European Commission, Joint Research Centre*
Directorate for Space, Security and Migration
Safety and Security of Buildings Unit

Sunday 6th August, 2023 @ 16:36

Contents

List of Tables

List of Figures

Abstract

This report presents the formulation and implementation of a friction model in the contact-impact algorithm based upon generalized pinballs (`GPIN`) recently introduced in the EUROPLEXUS code (also abbreviated as EPX). Some corrections are also applied to the much older pinball-based (`PINB`) contact-impact model of the code. For completeness, the closely related friction model used in the sliding-surface contact algorithm developed at CEA (`GLIS`), is also briefly presented.

The code is jointly developed by the French Commissariat à l'Energie Atomique (CEA DMT Saclay) and by EC-JRC. Its application domain is the numerical simulation of fast transient phenomena such as explosions, crashes and impacts in complex three-dimensional fluid-structure systems.

Recently, EPX is being used to simulate vehicle crash against obstacles such as road barriers for safety and security studies. These studies involve complex contact-impact scenarios, which in EPX are typically modelled by the method of Lagrange Multipliers (LM), although other methods (e.g. penalty-based) are also available in the code. In some test cases, the effect of friction may be important and it must be included in the numerical simulations. Hence the importance of having robust and validated friction models available in the code.

The contact models with friction are tested in a series of problems ranging from academic cases to realistic applications, including a sliding disk in either 2D or 3D, a blast-loaded clamped plate experiment and a shocktube-loaded clamped plate experiment.

Keywords: *Contact-Impact, Friction, Pinballs, Generalized pinballs.*

Foreword

This report is part of a large series of scientific-technical documents that are meant to provide essential information and documentation to users and developers of the EUROPLEXUS code. EUROPLEXUS (also abbreviated as EPX) is a computer code jointly developed by the French Commissariat à l'Energie Atomique (CEA DMT Saclay) and by EC-Joint Research Centre (JRC Ispra) within the framework of contractual agreements between the two research bodies.

EPX is a mature, general-purpose Finite Element and Finite Volume explicit code under active development since 1999, for the simulation of fast transient dynamic events in complex fluid-structure systems. It is an evolution of its ancestor PLEXIS-3C, which was also jointly developed by CEA and JRC in the 1980s and early '90s.

The code has been traditionally used in safety studies, ranging from nuclear reactors, to energy plants, to chemical and industrial plants, off-shore structures, car and road barrier crashes, among others. More recently it has proven a very useful tool in providing certified and independent numerical solutions in support of EC policies regarding the security of critical infrastructures and public spaces (like buildings, train and metro stations and carriages, etc.), which may be vulnerable to terrorist attacks or to natural disasters.

While being mainly of technical nature, the information contained in this series of reports is an invaluable source of reference for the users (as a complement to the User's manual) but also in particular for the developers of EPX. New models made available in the code are described in detail from the theoretical viewpoint. Several verification and application examples are also usually provided, in order to illustrate the practical use and to verify the correct functioning of the models.

Usually, at the end of each report an Appendix lists the input files that were used to produce the examples presented in the report. This allows users to re-run the test cases with EPX at any time and to use them as a basis for their own numerical simulations.

A complete list of the reports (produced both at JRC and at CEA) in this series can be found in the Bibliography section of the EPX User's manual [1].

1 Introduction

This report presents the formulation and implementation of a friction model in the contact-impact algorithm based upon generalized pinballs (`GPIN`) recently introduced in the EUROPLEXUS code. Some corrections are also applied to the much older pinball-based (`PINB`) contact-impact model of the code. For completeness, the closely related friction model used in the sliding-surface contact algorithm developed at CEA (`GLIS`), is also briefly presented.

EUROPLEXUS [1] (also abbreviated as EPX) is a computer code jointly developed by the French Commissariat à l'Energie Atomique (CEA DMT Saclay) and by EC-JRC. The code application domain is the numerical simulation of fast transient phenomena such as explosions, crashes and impacts in complex three-dimensional fluid-structure systems. The Cast3m [2] software from CEA is used as a pre-processor to EPX when it is necessary to generate complex meshes.

Recently, EPX is being used to simulate vehicle crash against obstacles such as road barriers for safety and security studies. These studies involve complex contact-impact scenarios, which in EPX are typically modelled by the method of Lagrange Multipliers (LM), although other methods (e.g. penalty-based) are also available in the code. In some test cases, the effect of friction may be important and it must be included in the numerical simulations. Hence the importance of having robust and validated friction models available in the code.

1.1 Contact

The most popular contact algorithms available in Finite Element (FE) computer codes are probably the so-called slide line (in 2D) and slide surface (in 3D) algorithms proposed by Hallquist and Benson [3,4]. They are based on the notion of penetration of slave nodes into master segments (in 2D) or into master surfaces (in 3D), like e.g. the `GLIS` contact model in EPX. These algorithms suffer from a number of geometrically pathological cases in which physical penetration is not detected.

1.1.1 The pinball (`PINB`) method

The pinball method proposed by Belytschko and co-workers from the late 80's [5–12] for application in impact problems with perforation is much more robust as concerns penetration detection. The pinball contact-impact method has been implemented in EPX in [14–18], initially based on strong coupling via a Lagrange-multiplier (LM) based solution strategy of the contact constraints (see [13] for details of the method) and more recently by using weak coupling based on a penalty approach, see [19]. The latter report also contains a description of the implementation of Assembled Surface Normals (ASNs) in the pinball model, by an algorithm inspired to the one proposed by Belytschko in reference [5].

In reference [20] the pinball (`PINB`) model is generalized in order to be compatible with mesh adaptivity, i.e. with automatic mesh refinement and un-refinement. Both adaptivity-driven contact and contact-driven adaptivity paradigms are examined in detail.

The original pinball model of Belytschko and its implementation in EPX (`PINB` keyword) are based upon spherical pinballs. This approach is extremely robust and efficient in detecting penetration, but suffers from certain drawbacks when it comes to imposing contact constraints.

When dealing with slender elements (highly deformed continuum elements or structural members such as bars, beams and shells) the basic pinball method which associates a sphere with each element is no longer applicable and a hierarchic pinball method, consisting of splitting each penetrating pinball into a series of smaller pinballs (recursively, until a certain minimum size is reached) must be used instead, to enhance the resolution of penetration detection. Apart from the complexity and relative inefficiency of a recursive approach, difficulties arise when imposing multiple contacts between sub-pinballs with the LM method, because in this case redundant constraints may be generated and the system of constraints may become singular (see e.g. Section 7 of reference [15]). Various techniques have been devised in order to get rid of redundant constraints, but this is a complex and inefficient task, very difficult to perform in general terms. The penalty approach does not suffer from this limitation, but it needs some tuning parameters, which render solutions more laborious (besides being non-unique) and less reliable for the user, than the LM method.

Another difficulty in dealing with spherical-only pinballs is the determination of the local contact direction. The simplistic approach of using the line joining the two contacting pinball centres is sufficient in some cases (e.g. in fast impact with perforation) but may introduce large errors in other situations (such as sliding-dominated contact between smooth bodies). Recent (re-)developmeent of the ASN technique (see [19]) which associates a unique normal direction with each pinball or subpinball and then introduces rules for computing a "better" contact direction than the simple centres-joining line can alleviate these problems in many, but not in all, cases.

The pinball model originally implemented in EPX (until late 2016) assumed friction-less contact. However, friction is a very important phenomenon in many realistic applications. In reference [21] a first attempt was made to include friction in pinball-based (`PINB`) contacts. However, only the decoupled (penalty-based) version of the pinball model (`LINK DECO PINB PENA`) was detailed in reference [21] and in the numerical examples presented therein.

It is believed that, on the same occasion, friction had been tentatively added also to the coupled (LM-based) version of the pinball model (since it is still present in the code as of this writing). However, the LM-based pinball model with friction was never thoroughly tested so far and the decoupled version was used in all applications ever since, whenever friction was present. Furthermore, as of this writing (22 May 2023), there exist no tests involving LM-based pinball contact with friction in the EPX non-regression suite (while there are some tests verifying the penalty based pinballs with friction).

The fact that the contact normals are not accurately computed by the standard pinball (`PINB`) model, and that this model may lead to multiple and possibly redundant constraints, represents a serious drawback in the formulation and implementation of a friction model to be associated with the `PINB` contact algorithm.

1.1.2 The generalized pinball (`GPIN`) method

In a recent and still ongoing work [22] we explore the possibility of using so-called *generalized* pinballs (GPINs), of various geometrical shapes (not only spherical), in an attempt to avoid the problems highlighted above (especially those affecting the LM method), while retaining as far as possible the robustness and simplicity of the original pinball approach. The use of a variety of shapes allows to get rid of the necessity of a hierarchic procedure at the expense, however, of more complex penetration checks than with spheres. This should ensure that no (or fewer) redundant constraints appear and therefore their elimination is no longer necessary, with positive effects also on the treatment of rebound and of friction.

In the present work, the friction model introduced in `PINB` in reference [21], slightly corrected along the lines recently proposed by Bung and Potapov [23], is implemented also in `GPIN`, by taking advantage of the fact that contact normals are much better defined in the `GPIN` model, and also that redundant constraints are avoided by construction. The same corrections proposed in [23] are also applied to the standard `PINB` friction model, as detailed in the following Sections.

2 Formulation of the friction model

The friction model adopted here is inspired by the simple Coulomb-type (dry) friction model already available in EPX for the sliding lines and sliding surfaces contact model (`GLIS`) developed at CEA, see references [24–29].

2.1 Coulomb model

The classical Coulomb friction model, suitable for contact between dry (non-lubricated) solids, distinguishes between *static* friction and *kinetic* friction between two contacting bodies.

The static case occurs when the relative velocity of the two bodies along the contacting surface is null. In this case, friction generates a force directed along the tangent to the contact interface, which contrasts the mutual sliding of the bodies. The friction force is equal and opposite to the relative internal force along the tangent (i.e. the force that would tend to create sliding between the bodies), but only up to a limiting value equal to μ_s times the normal contact force:

$$F_t^s = \min\left(F_t^{\text{int}}, \mu_s F_n\right) \tag{1}$$

The kinetic case occurs when there is relative sliding of the two bodies along the contact surface, i.e. when there is a non-zero relative velocity of the two bodies along the tangent to the contact interface. In this case, friction generates a tangential force component along the direction of the relative velocity and opposed to it, whose magnitude is independent of the relative velocity, and equal to μ_k times the normal contact force:

$$F_t^k = \mu_k F_n \tag{2}$$

2.1.1 Interpolation of the friction coefficient

Experience shows that for most materials it is $\mu_k < \mu_s$ (indicating that once two previously sticking bodies start sliding with respect to each other it becomes easier for them to continue sliding) and that the passage between static and kinetic friction regime is rather abrupt. However, dealing with a discontinuous state of friction in a numerical model may be challenging and may lead to numerical instabilities. Therefore, it is preferred to use a single friction regime in the model, where the friction coefficient is continuously interpolated between the static and the kinetic values. Thus, the kinetic value of the coefficient will be reached only for very large (ideally infinite) relative velocity. From reference [25], the friction coefficient μ is expressed by:

$$\mu = \mu_k + \left(\mu_s - \mu_k\right) e^{-\gamma |v_{rt}|} \tag{3}$$

where v_{rt} is the tangential component of the relative velocity vector $\boldsymbol{v}_r$ and γ is a decay parameter having as dimension the inverse of a velocity.

When $v_{rt} = 0$ we have $\mu = \mu_s$ while, for increasing values of $|v_{rt}|$, the value of μ tends asymptotically to μ_k which, as already noted, is typically smaller than μ_s. The greater is the decay parameter γ, the faster is the decay of μ towards the limit value μ_k.

A practical procedure for choosing the value of γ is as follows. By indicating with α the value of the exponential:

$$\alpha = e^{-\gamma |v_{rt}|} \tag{4}$$

eq. (3) becomes:

$$\mu = \mu_k + \alpha\left(\mu_s - \mu_k\right) \tag{5}$$

From this one obtains for α:

$$\alpha = \frac{\mu - \mu_k}{\mu_s - \mu_k} \tag{6}$$

For example, if $\alpha = 0.01$ then the value of μ becomes equal to the asymptotic value μ_k within 1 % of the difference $(\mu_s - \mu_k)$ for the values of γ and v_{rt} which, combined, give α according to (4).

To determine the γ parameter the user may proceed as follows. From (4) one has:

$$-\gamma\,|v_{rt}| = \ln\alpha \tag{7}$$

or:

$$\gamma = -\frac{\ln\alpha}{|v_{rt}|} \tag{8}$$

By choosing the desired tolerance α and the corresponding value of the relative tangential velocity $|v_{rt}|$, one can compute γ from (8). For example, assume that we want the friction coefficient to reach the kinetic value with a tolerance of 1 % (of the difference between the static and kinetic coefficients) at a relative velocity of 2 m/s or higher. Then we get from (8):

$$\gamma = -\frac{\ln 0.01}{2.0} = 2.3025 \tag{9}$$

As an example, Figure 1 shows the variation of μ as a function of $|v_{rt}|$ (in the range 0.0 to 2.0 m/s) for $\mu_s = 0.3$, $\mu_k = 0.1$ and various values of γ. Note that, for $\gamma = 0$, from (3) one obtains $\mu = \mu_S$, *independently* from the velocity $|v_{rt}|$ *and* from the value of μ_k that has been chosen.

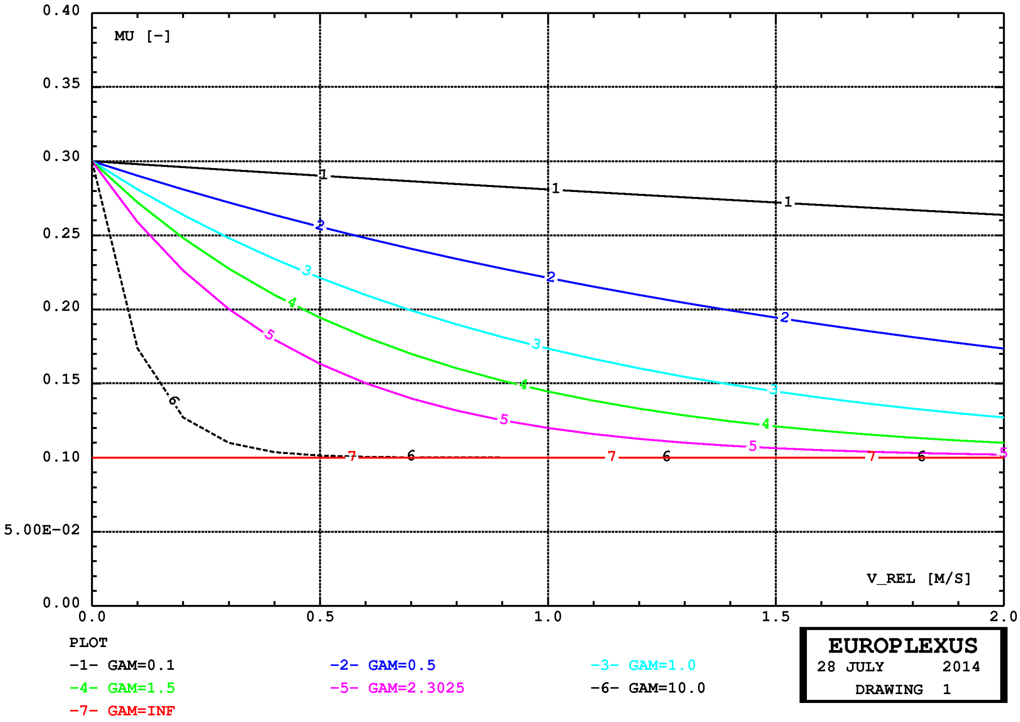


Figure 1: Friction coefficient *vs.* relative velocity for $\mu_s = 0.3$, $\mu_k = 0.1$ and various values of γ.

By the input directives, different friction values can be independently assigned to each potentially contacting body. If no `FRIC` keyword is specified in a `BODY` or `SELF` directive, then no friction is associated by default to the corresponding body. If two bodies come into contact, and none or only one of them has an associated friction, then no friction is computed for the contact. If *both* bodies have an associated friction, then the values μ_1 and μ_2 are evaluated with (3) for each body and then the *minimum* of the two results is retained for the present contact:

$$\mu = \min(\mu_1, \mu_2) \tag{10}$$

Note that this rule corrects and replaces the strategy that was adopted in the `PINB` model prior to the present work, namely that the values μ_s, μ_k and γ of the body with the *lowest* μ_k were retained for the contact between these two bodies.

3 Friction algorithms

We first describe the numerical algorithm for friction used in the sliding lines and sliding surfaces (`GLIS` directive) of EPX. Then, the model is adapted for use in conjunction with pinballs (`PINB` directive) and with generalized pinballs (`GPIN` directive). Both the LM-based and the penalty-based variant of each model is considered, whenever possible.

3.1 The `GLIS` model

3.1.1 The uncorrected LM-based `GLIS` friction algorithm

According to Bung (see [30] in Appendix A) the friction algorithm originally implemented in the `GLIS` (and `IMPA`) directives of EPX (i.e., before the corrections proposed in [23] by Bung and Potapov) was formulated as follows.

First, contact is *detected* by a master / slave technique. The code checks the penetration of a *slave node* into a *master surface* represented by a so-called *defender* (fictitious) node and corresponding to a facet of the penetrated body (i.e. a triangle or a quadrilateral, in 3D).

If penetration occurs (or if it *would* occur at the next step, in the absence of any contact forces), then according to the method of Lagrange Multipliers (LM) one must impose some *constraints* between the degrees of freedom of the two nodes, the slave node s and the defender node d (i.e., a linear combination of the M nodes m_i, $i = 1, \cdots, M$ forming the master face, with $M = 3$ or 4), along a specific direction. Writing the (normal) constraints requires the knowledge of:

- The slave (penetrating) node.
- The defender (penetrated) node. This is a fictitious node, whose degrees of freedom are linear combinations (through suitable position-interpolating functions) of the degrees of freedom of the master nodes, i.e. the nodes of the penetrated master face.
- A direction, defined by the (unit) *contact normal* $\hat{\boldsymbol{n}}$.

If *no friction* is present, only one constraint is needed which expresses the equality of the velocity of the slave node $\boldsymbol{v}_s$ and of the defender (master) node $\boldsymbol{v}_d$, projected along the contact normal:

$$\boldsymbol{v}_s \cdot \hat{\boldsymbol{n}} = \boldsymbol{v}_d \cdot \hat{\boldsymbol{n}} = \left(\sum_{i=1}^{M} c_i \boldsymbol{v}_{m_i} \right) \cdot \hat{\boldsymbol{n}} \tag{11}$$

where $\boldsymbol{v}_{mi}$ is the velocity of the generic master node belonging to the master face (which has M nodes) and c_i is the corresponding weight. That is, the c_i are the shape functions interpolating the position of the defender node onto the master face.

More precisely, assuming that we are computing the solution at t^{n+1} in the discrete time integration procedure, then all quantities are known at time t^n. The configuration $\boldsymbol{x}^{n+1}$ is also known, while the last known velocity is the mid-step value $\boldsymbol{v}^{n+1/2}$. Therefore, the velocities appearing in eq. (11) are those at the next mid-step, that is $\boldsymbol{v}^{n+3/2}$.

As concerns the normal direction, for consistency with the velocity one should also take the normal at $n + 3/2$. However, the configuration at $n + 3/2$ (or at $n + 2$) is not known yet and will depend upon the reaction forces at $n + 1$ that we are just trying to compute. Therefore, the problem would become implicit and would require an iterative solution. In order to simplify the implementation, and in consideration of the fact that in an explicit code such as EPX the time increment Δt is small anyway due to stability requirements, the normal $\hat{\boldsymbol{n}}^{n+1}$ on the current configuration is used instead.

Therefore, eq. (11) can be written more precisely:

$$\boldsymbol{v}_s^{n+3/2} \cdot \hat{\boldsymbol{n}}^{n+1} = \boldsymbol{v}_d^{n+3/2} \cdot \hat{\boldsymbol{n}}^{n+1} = \left(\sum_{i=1}^{M} c_i \boldsymbol{v}_{m_i}^{n+3/2} \right) \cdot \hat{\boldsymbol{n}}^{n+1} \qquad \text{Normal} \tag{12}$$

By imposing this constraint using the method of Lagrange multipliers, the corresponding multiplier λ is found and then the reaction forces are computed. In order to allow for *rebound*, the sign of the multiplier is checked *a posteriori*. If $\lambda > 0$ then attractive (instead of repulsive) reaction forces would result between the two bodies. Since this behaviour would be non-physical, the multiplier is nullified ($\lambda = 0$) in this case, thus resulting in zero reaction forces.

If *friction* is present in the contact between the two bodies, then the same contact constraint along the contact normal as in the case without friction (say constraint I) is imposed. Then a *second constraint* J is (tentatively) added, which imposes the equality of the velocities also along the *tangential* direction $\hat{\boldsymbol{\tau}}$:

$$\boldsymbol{v}_s^{n+3/2} \cdot \hat{\boldsymbol{\tau}}^{n+1} = \boldsymbol{v}_d^{n+3/2} \cdot \hat{\boldsymbol{\tau}}^{n+1} = \left(\sum_{i=1}^{M} c_i \boldsymbol{v}_{m_i}^{n+3/2} \right) \cdot \hat{\boldsymbol{\tau}}^{n+1} \qquad \text{Tangential} \tag{13}$$

In two space dimensions, the tangential direction is uniquely defined as being perpendicular to the contact normal ($\hat{\tau} \cdot \hat{n} = 0$). However, in full 3D any vector lying in the *plane* perpendicular to the contact normal may represent a valid tangential direction. In order to evaluate a single tangential vector, the algorithm described in [30] computes the projection of the relative velocity $\boldsymbol{v}_r$ onto the tangential plane and then normalizes the result in order to obtain the tangent vector. The *relative velocity* is given by:

$$\boldsymbol{v}_r^{n+1/2} = \boldsymbol{v}_s^{n+1/2} - \boldsymbol{v}_d^{n+1/2} \tag{14}$$

which, projected along the normal gives:

$$\boldsymbol{v}_{rn} = \left(\boldsymbol{v}_r^{n+1/2} \cdot \hat{\boldsymbol{n}}^{n+1} \right) \hat{\boldsymbol{n}}^{n+1} \tag{15}$$

The tangential relative velocity is obtained by difference:

$$\boldsymbol{v}_{r\tau} = \boldsymbol{v}_r^{n+1/2} - \boldsymbol{v}_{rn} \tag{16}$$

and the tangential unit vector results from normalization:

$$\hat{\boldsymbol{\tau}}^{n+1} = \frac{\boldsymbol{v}_{r\tau}}{v_{r\tau}} = \frac{\boldsymbol{v}_{r\tau}}{||\boldsymbol{v}_{r\tau}||} \tag{17}$$

where $v_{r\tau} = ||\boldsymbol{v}_{r\tau}||$ represents the norm (length) of the vector $\boldsymbol{v}_{r\tau}$.

A couple of remarks can be raised concerning this procedure for the determination of the tangent direction. First, in (13) we use the value of $\hat{\boldsymbol{\tau}}$ at $n+1$ instead of the value at $n+3/2$, in analogy with the use of the normal $\hat{\boldsymbol{n}}^{n+1}$ in (12): however, the value of $\hat{\boldsymbol{\tau}}^{n+1}$ given by (17) is just an approximation because the expressions (15–17) use (for efficiency) a mix of quantities at $n+1/2$ (the velocities) and at $n+1$ (the normal). Second, the procedure (14–17) to compute the tangent vector works only if the relative velocity $\boldsymbol{v}_r^{n+1/2}$ given by (14) does not vanish.

If $\boldsymbol{v}_r^{n+1/2}$ is zero (or very small), then the direction of the tangent is undetermined (or very inaccurate at best). Reference [30] does not specify how the algorithm should behave in this case. The code should be inspected to find out.

To summarize, in the case with friction, for each detected contact (penetration) we have two constraints: a constraint I along the normal, associated with a Lagrange multiplier λ_I, and a (provisional) constraint J along the tangent, associated with another Lagrange multiplier λ_J. In the implementation, it is $J = I + 1$, that is, the tangent constraint always follows directly the corresponding normal constraint in the list of links.

The system of links is solved by the Lagrange multipliers method, thus obtaining the two Lagrange multipliers λ_I and λ_J. The reactions are *provisionally* computed by:

$$\boldsymbol{R} = \boldsymbol{C}^T \boldsymbol{\lambda} \tag{18}$$

where $\boldsymbol{C}$ is the matrix of constraint coefficients. Next:

1. We check the normal constraint for rebound. If $\lambda_I \geq 0$, then there would be traction along the normal, i.e. rebound is occurring. We set $\lambda_I = 0$ and $\lambda_J = 0$ (if present), thus eliminating *both* constraints from the system, and we re-compute the reactions by (18). The reactions concerning the contact under consideration will vanish out because the corresponding multipliers are zero.

2. Else, $\lambda_I < 0$ and contact (not rebound) is actually occurring:
   - If no friction is involved in this contact (so that no J constraint is present), then the computed contact reactions (along the normal) for both the slave and the master nodes are already correct and the calculation is complete.
   - Otherwise, there is friction. The reactions along the normal are already the final ones, but those along the tangent need to be corrected if they do not satisfy Coulomb's condition. Let R_{sn} denote the (final) normal reaction and $R_{s\tau}$ the (provisional) tangential reaction at the slave node s resulting, respectively, from the two constraints I and J. Then:
     - If $R_{s\tau} \leq \mu R_{sn}$, then the Coulomb condition is satisfied. All the computed reactions are already correct and the calculation is complete.
     - Else $R_{s\tau} > \mu R_{sn}$ and the Coulomb condition is not satisfied, i.e. the provisional tangential reaction force causes the total provisional reaction force to lie outside Coulomb's cone. We evaluate a *reduction coefficient* α:

$$\alpha = \frac{\mu R_{sn}}{R_{s\tau}} < 1 \tag{19}$$

and then we scale the tangential multiplier accordingly ($\lambda_J \Leftarrow \alpha\lambda_J$). Thus, for the slave node we get:

$$\boldsymbol{R}_{s\tau} \Leftarrow \alpha \boldsymbol{R}_{s\tau} \tag{20}$$

and, similarly, for the master nodes m_i:

$$\boldsymbol{R}_{m_i\tau} \Leftarrow \alpha \boldsymbol{R}_{m_i\tau} \qquad i = 1, \cdots, M \tag{21}$$

3.1.2 Recent correction of the LM-based `GLIS` friction algorithm [23]

Recently it was noted [23] that the friction model of the `GLIS` directive failed in correctly representing a simple compression test whereby a block of concrete is pushed (normally) against a rigid surface. Due to the Poisson effect, the block tends to deform (also) laterally. In the real world this lateral deformation is inhibited along the surface of the block in contact with the rigid surface, if the friction is sufficiently strong. Note that in this case, static friction is activated, i.e. the relative velocity between the block and the surface is zero (sticking contact). The EPX model using `GLIS` was unable to represent this behavior, for whatever value of the friction coefficient chosen in the simulation.

In general, the algorithm described in the previous Section gives correct results whenever a well-defined relative tangential velocity exists between the two contacting bodies, as it occurs in most contact applications. However, it fails when the direction of relative sliding is uncertain or undetermined, such as in the case of sticking contact. This shortcoming had been foreseen in the remarks of the previous Section.

The solution proposed in [23] is to refrain from using the tangential relative velocity $\boldsymbol{v}_{r\tau}$ in order to define the *tangential sliding direction*, thus avoiding the problem that arises when $|\boldsymbol{v}_{r\tau}| \approx 0$.

By restricting ourselves to the general 3D case (`GLIS` is not available in 2D), a local orthonormal reference frame is built for every contact, having one axis $\hat{\boldsymbol{n}}$ directed along the contact normal, like in the original model. Then, instead of just one tangent axis $\hat{\boldsymbol{\tau}}$, *two* tangent axes $\hat{\boldsymbol{\tau}}_1$ and $\hat{\boldsymbol{\tau}}_2$ are constructed, which define the *tangent plane* τ (perpendicular to $\hat{\boldsymbol{n}}$). The choice of these two particular tangent axes is irrelevant, as long as they form an orthonormal basis together with $\hat{\boldsymbol{n}}$. The solution will not depend upon the particular axes chosen. Then in the case of friction, instead of considering just one tangential link, one considers *two* tangential links of the form (13), along $\hat{\boldsymbol{\tau}}_1$ and $\hat{\boldsymbol{\tau}}_2$.

Another less important modification with respect to the original version of the algorithm is that the two tangential links are written (i.e., added to the set of links) *before* and not after the corresponding normal link in the list of imposed links. Thus the constraints become:

$$\boldsymbol{v}_s^{n+3/2} \cdot \hat{\boldsymbol{\tau}}_1{}^{n+1} = \boldsymbol{v}_d^{n+3/2} \cdot \hat{\boldsymbol{\tau}}_1{}^{n+1} = \left(\sum_{i=1}^{M} c_i \boldsymbol{v}_{m_i}^{n+3/2} \right) \cdot \hat{\boldsymbol{\tau}}_1{}^{n+1} \qquad \text{First tangential} \tag{22}$$

$$\boldsymbol{v}_s^{n+3/2} \cdot \hat{\boldsymbol{\tau}}_2{}^{n+1} = \boldsymbol{v}_d^{n+3/2} \cdot \hat{\boldsymbol{\tau}}_2{}^{n+1} = \left(\sum_{i=1}^{M} c_i \boldsymbol{v}_{m_i}^{n+3/2} \right) \cdot \hat{\boldsymbol{\tau}}_2{}^{n+1} \qquad \text{Second tangential} \tag{23}$$

$$\boldsymbol{v}_s^{n+3/2} \cdot \hat{\boldsymbol{n}}^{n+1} = \boldsymbol{v}_d^{n+3/2} \cdot \hat{\boldsymbol{n}}^{n+1} = \left(\sum_{i=1}^{M} c_i \boldsymbol{v}_{m_i}^{n+3/2} \right) \cdot \hat{\boldsymbol{n}}^{n+1} \qquad \text{Normal} \tag{24}$$

This results in the following modified algorithm (in 3D). In the case with friction, for each detected contact (penetration) we have *three* constraints: two (provisional) constraints J_1 and J_2 along the tangents $\hat{\boldsymbol{\tau}}_1$ and $\hat{\boldsymbol{\tau}}_2$, associated with Lagrange multipliers λ_{J_1} and λ_{J_2}, respectively, followed by a constraint I along the normal $\hat{\boldsymbol{n}}$, of Lagrange multiplier λ_I. In the implementation, it is $I = J_1 + 2 = J_2 + 1$, that is, in the list of links the first tangent constraint is directly followed by the second tangent constraint, which in turn is followed directly by the corresponding normal constraint.

The system of links is solved by the Lagrange multipliers method, thus obtaining the three Lagrange multipliers λ_{J_1}, λ_{J_2} and λ_I. The reactions are *provisionally* computed by:

$$\boldsymbol{R} = \boldsymbol{C}^T \boldsymbol{\lambda} \tag{25}$$

where $\boldsymbol{C}$ is the matrix of constraint coefficients. Next:

1. We check the normal constraint for rebound. If $\lambda_I \geq 0$, then there would be traction along the normal, i.e. rebound is occurring. We set $\lambda_I = 0$ and $\lambda_{J_1} = \lambda_{J_2} = 0$ (if present), thus eliminating *all three* constraints from the system, and we re-compute the reactions by (25). The reactions concerning the contact under consideration will vanish out because the corresponding multipliers are zero.

2. Else, $\lambda_I < 0$ and contact (not rebound) is actually occurring:

   - If no friction is involved in this contact (so that neither J_1 nor J_2 constraint is present), then the computed contact reactions (along the normal) for both the slave and the master nodes are already correct and the calculation is complete.
   - Otherwise, there is friction. The reactions along the normal are already the final ones, but those in the tangent plane need to be corrected if they do not satisfy Coulomb's condition. Let R_{sn} denote the (final) normal reaction and $R_{s\tau_1}$, $R_{s\tau_2}$ the (provisional) tangential reactions at the slave node s resulting, respectively, from the constraints I, J_1 and J_2. Compute the magnitude of the tangential reaction:

     $$R_{s\tau} = \sqrt{R_{s\tau_1}^2 + R_{s\tau_2}^2} \tag{26}$$

     Then:

     - If $R_{s\tau} \leq \mu R_{sn}$, then the Coulomb condition is satisfied. All the computed reactions are already correct and the calculation is complete.
     - Else $R_{s\tau} > \mu R_{sn}$ and the Coulomb condition is not satisfied, i.e. the provisional tangential reaction force causes the total provisional reaction force to lie outside Coulomb's cone. We evaluate a *reduction coefficient* α:

       $$\alpha = \frac{\mu R_{sn}}{R_{s\tau}} < 1 \tag{27}$$

       and then we scale the two tangential reactions accordingly. To this end, we scale down the Lagrange multipliers of the tangential constraints:

       $$\lambda_{J_1} \Leftarrow \alpha \lambda_{J_1} \qquad \lambda_{J_2} \Leftarrow \alpha \lambda_{J_2} \tag{28}$$

       which results in the following effect. For the slave node we obtain, formally:

       $$\boldsymbol{R}_{s\tau} \Leftarrow \alpha \boldsymbol{R}_{s\tau} \tag{29}$$

       and, similarly, also the tangential reactions on the master nodes m_i are scaled by the same factor α:

       $$\boldsymbol{R}_{m_i\tau} \Leftarrow \alpha \boldsymbol{R}_{m_i\tau} \qquad i = 1, \cdots, M \tag{30}$$

3.1.3 The penalty-based `GLIS` friction algorithm

A decoupled version of the `GLIS` friction algorithm is believed to be available in EPX, at least according to the Users' manual [1]. However, no information on the model's formulation (in particular on the availability of friction) is known.

It is not even clear whether the model is based on penalty (similarly to the decoupled PINB model) or on another type of decoupled formulation. There do exist some (very few) benchmarks which use `LINK DECO GLIS` in the standard EPX non-regression set. It should be checked with CEA whether there exists any written documentation on the subject and/or any application examples.

3.2 The `PINB` model

3.2.1 The uncorrected LM-based `PINB` friction model

The LM-based `PINB` friction model was first implemented by using the single-tangent formulation (in 3D), similar to that used in the uncorrected friction model of `GLIS` described in Section 3.1.1. Note that the model was never fully documented. In fact reference [21], despite its (somewhat misleading) title, describes only the penalty version of friction in `PINB`.

Producing a precise documentation of the model now is not worthwhile, since it is obsolete and no longer available in EPX since May 2023. However, the two main differences between (uncorrected) `GLIS` and `PINB` concerning the friction may be briefly summarized as follows. First, the rebound is treated *a posteriori* (i.e., based on the sign of the normal multiplier) in `GLIS` while it is treated *a priori*, by default, in `PINB`. The second difference was in the calculation of the tangent direction, where `GLIS` only used the relative velocity $v_{r\tau}$ (with the corresponding pathological case if $v_{r\tau} \approx 0$) while `PINB` attempted using the relative internal force as a second chance when $v_{r\tau} \approx 0$.

Note that the single-tangent (uncorrected) version is the model used in all numerical examples with `PINB` and friction included in the present report (Section 4), because such examples were run before the upgrade of the model described in Section 3.2.2. As a consequence, all the examples of Section 4 with `PINB` and friction should eventually be re-run, to check whether there are any significant changes in the results.

3.2.2 Recent correction of the LM-based `PINB` friction model

Recently, the `PINB` model has been updated and now (from May 2023) it uses the two-tangent (in 3D) formulation (see the correction for `GLIS` in Section 3.1.2). The upgraded `PINB` model is described in reference [31] and also briefly summarized below for completeness.

The upgraded, two-tangent version of the friction algorithm for the PINB contact model is now basically identical to that used by the other contact models, in particular that of GLIS described in Section 3.1.2. However, it also presents some particularities, which will be highlighted during the description.

The first particularity concerns the rebound. In GLIS, the rebound is always checked *a posteriori*. In the PINB model, rebound is checked *a priori* by default. In that case, if rebound is preliminarly detected, the contact is discarded before being treated (no associated links are imposed). Therefore, only the case of *a posteriori* rebound is treated in the contact algorithm described below. Another particularity is that the PINB model is available both in 3D and in 2D, while GLIS is only available in 3D.

By considering first the updated PINB model in the general 3D case, a local orthonormal reference frame is built for every (non a-priori rebounding) contact, with one axis $\hat{\boldsymbol{n}}$ directed along the contact normal, like in the original model. Then, instead of just one tangent axis $\hat{\boldsymbol{\tau}}$, *two* tangent axes $\hat{\boldsymbol{\tau}}_1$ and $\hat{\boldsymbol{\tau}}_2$ are constructed, which define the *tangent plane* τ (perpendicular to $\hat{\boldsymbol{n}}$). The choice of these two particular tangent axes is irrelevant, as long as they form an orthonormal basis together with $\hat{\boldsymbol{n}}$. The solution will not depend upon the particular axes chosen. Then in the case of friction, instead of considering just one tangential link, we consider *two* tangential links, along $\hat{\boldsymbol{\tau}}_1$ and $\hat{\boldsymbol{\tau}}_2$.

Another less important modification with respect to the original version of the algorithm is that the two tangential links are written (i.e., added to the set of links) *before* and not after the corresponding normal link in the list of imposed links. Thus the constraints become:

$$\boldsymbol{v}_s^{n+3/2} \cdot \hat{\boldsymbol{\tau}}_1{}^{n+1} = \boldsymbol{v}_d^{n+3/2} \cdot \hat{\boldsymbol{\tau}}_1{}^{n+1} = \left(\sum_{i=1}^{M} c_i \boldsymbol{v}_{m_i}^{n+3/2} \right) \cdot \hat{\boldsymbol{\tau}}_1{}^{n+1} \qquad \text{First tangential} \tag{31}$$

$$\boldsymbol{v}_s^{n+3/2} \cdot \hat{\boldsymbol{\tau}}_2{}^{n+1} = \boldsymbol{v}_d^{n+3/2} \cdot \hat{\boldsymbol{\tau}}_2{}^{n+1} = \left(\sum_{i=1}^{M} c_i \boldsymbol{v}_{m_i}^{n+3/2} \right) \cdot \hat{\boldsymbol{\tau}}_2{}^{n+1} \qquad \text{Second tangential} \tag{32}$$

$$\boldsymbol{v}_s^{n+3/2} \cdot \hat{\boldsymbol{n}}^{n+1} = \boldsymbol{v}_d^{n+3/2} \cdot \hat{\boldsymbol{n}}^{n+1} = \left(\sum_{i=1}^{M} c_i \boldsymbol{v}_{m_i}^{n+3/2} \right) \cdot \hat{\boldsymbol{n}}^{n+1} \qquad \text{Normal} \tag{33}$$

This results in the following modified algorithm (in 3D). In the case with friction, for each detected contact (penetration) we have *three* constraints: two (provisional) constraints J_1 and J_2 along the tangents $\hat{\boldsymbol{\tau}}_1$ and $\hat{\boldsymbol{\tau}}_2$, associated with Lagrange multipliers λ_{J_1} and λ_{J_2}, respectively, followed by a constraint I along the normal $\hat{\boldsymbol{n}}$, of Lagrange multiplier λ_I. In the implementation, it is $I = J_1 + 2 = J_2 + 1$, that is, in the list of links the first tangent constraint is directly followed by the second tangent constraint, which in turn is followed directly by the corresponding normal constraint.

The system of links is solved by the Lagrange multipliers method, thus obtaining the three Lagrange multipliers λ_{J_1}, λ_{J_2} and λ_I. The reactions are *provisionally* computed by:

$$\boldsymbol{R} = \boldsymbol{C}^T \boldsymbol{\lambda} \tag{34}$$

where $\boldsymbol{C}$ is the matrix of constraint coefficients. At this point, in the other EPX contact models (GLIS/IMPA, GPIN and SLID), the reaction $\boldsymbol{R}_s = (R_{sn}, R_{s\tau_1}, R_{s\tau_2})$ at the slave node (ISLAVE) is evaluated directly. However, in the case of the PINB model, there is no slave node and, in general, the reactions act over all nodes of the elements containing the two pinballs A and B involved in the contact.

Therefore, we arbitrarily select the first contacting pinball (denoted the A pinball) as the “slave” pinball, only as far as the present rebound and friction calculations are concerned, and compute $\boldsymbol{R}_s = \boldsymbol{R}_A$ by assembling the reactions at all the n_A nodes i_A of the element containing this pinball, obtaining $\boldsymbol{R}_s = \boldsymbol{R}_A = \sum_{i_A=1}^{n_A} \boldsymbol{R}_{i_A}$ (see details in Section 3.2.3). Next:

1. If rebound should be checked *a posteriori* (`REB2` option), we check the normal reaction R_{sn} for rebound. If $R_{sn} < 0$, then there would be traction along the normal, i.e. a posteriori rebound is occurring. We set $\lambda_I = 0$ and $\lambda_{J_1} = \lambda_{J_2} = 0$ (if present), thus eliminating *all three* constraints from the system, and we re-compute the reactions by (34). The reactions concerning the contact under consideration will vanish out because the corresponding multipliers are zero.

2. Else, $R_{sn} \geq 0$ and contact (not a posteriori rebound) is actually occurring:

   - If no friction is involved in this contact (so that neither J_1 nor J_2 constraint is present), then the computed contact reactions (along the normal) for all nodes of both pinballs are already correct and the calculation is complete.
   - Otherwise, there is friction. The reactions along the normal are already the final ones, but those in the tangent plane need to be corrected if they do not satisfy Coulomb’s condition. Let R_{sn} denote the (final) normal reaction and $R_{s\tau_1}$, $R_{s\tau_2}$ the (provisional) tangential reactions at the “slave” pinball $s \equiv A$ resulting, respectively, from the constraints I, J_1 and J_2. Compute the magnitude of the tangential reaction:

     $$R_{s\tau} = \sqrt{R_{s\tau_1}^2 + R_{s\tau_2}^2} \tag{35}$$

     Then:

     - If $R_{s\tau} \leq \mu R_{sn}$, then the Coulomb condition is satisfied. All the computed reactions are already correct and the calculation is complete.

- Else $R_{s\tau} > \mu R_{sn}$ and the Coulomb condition is not satisfied, i.e. the provisional tangential reaction force causes the total provisional reaction force to lie outside Coulomb's cone. We evaluate a *reduction coefficient* α:

  $$\alpha = \frac{\mu R_{sn}}{R_{s\tau}} < 1 \tag{36}$$

  and then we scale the two tangential reactions accordingly. To this end, we scale down the Lagrange multipliers of the tangential constraints:

  $$\lambda_{J_1} \Leftarrow \alpha\lambda_{J_1} \qquad \lambda_{J_2} \Leftarrow \alpha\lambda_{J_2} \tag{37}$$

  which results in the following effect. For the nodes of the "slave" pinball we obtain, formally:

  $$\boldsymbol{R}_{s_i\tau} \Leftarrow \alpha\boldsymbol{R}_{s_i\tau} \qquad i = 1, \cdots, n_A \tag{38}$$

  and, similarly, also the tangential reactions on the nodes m_i of the "master" pinball B are scaled by the same factor α:

  $$\boldsymbol{R}_{m_i\tau} \Leftarrow \alpha\boldsymbol{R}_{m_i\tau} \qquad i = 1, \cdots, n_B \tag{39}$$

In the 2D case with friction, for each detected contact we have *two* constraints: one (provisional) constraint J along the tangent $\hat{\boldsymbol{\tau}}$, associated with Lagrange multiplier λ_J, followed by a constraint I along the normal $\hat{\boldsymbol{n}}$, of Lagrange multiplier λ_I. In the implementation, it is $I = J + 1$, that is, in the list of links the tangent constraint is directly followed by the corresponding normal constraint.

The rest of the algorithm is perfectly analogous to the 3D case, but with only one tangent direction instead of two. Thus, the tangent reaction $R_{s\tau}$ is computed directly and not through Eq. (35).

3.2.3 Computing the total reaction on the "slave" pinball

As anticipated in the previous Section, the total reaction on the "slave" (A) pinball $\boldsymbol{R}_s = \boldsymbol{R}_A$ must be computed by assembling the reactions at all the n_A nodes i_A of the element containing this pinball:

$$\boldsymbol{R}_s = \boldsymbol{R}_A = \sum_{i_A=1}^{n_A} \boldsymbol{R}_{i_A} \tag{40}$$

This task is performed in the subroutine LINK_FRNORM_PINB of module M_LINK_PINB and is not straightforward. In the other EPX contact models (GLIS/IMPA, GPIN and SLID) there is a (single) slave node (ISLAVE) in the very definition of the model. Therefore, it is sufficient to pre-calculate the reaction at this node.

In the PINB case, the reactions at all "slave" nodes (the nodes of the element containing the A pinball) must be assembled. However, such nodes are not directly identifiable from the subroutine LINK_FRNORM_PINB, which has only access to the COEFL array. This array is dimensioned over the total number of dofs involved by the PINB link, i.e. it contains dofs of both the A pinball and the B pinball.

When the link is first written in subroutine BL_LINL_PINBALLS of module M_PINBALLS, the link has the form:

$$\boldsymbol{v}_A \cdot \hat{\boldsymbol{n}}_{AB} - \boldsymbol{v}_B \cdot \hat{\boldsymbol{n}}_{AB} = \boldsymbol{0} \tag{41}$$

where $\hat{\boldsymbol{n}}_{AB}$ is the unit contact normal oriented from pinball A towards pinball B. By expanding the pinball velocities in terms of the corresponding nodal velocities using the shape functions, this results in:

$$\left(\sum_{i_A=1}^{n_A} N_{iA}\boldsymbol{v}_{iA}\right) \cdot \hat{\boldsymbol{n}}_{AB} - \left(\sum_{i_B=1}^{n_B} N_{iB}\boldsymbol{v}_{iB}\right) \cdot \hat{\boldsymbol{n}}_{AB} = \boldsymbol{0} \tag{42}$$

that is, the link contains first the dofs related to the A pinball, then those related to the B pinball.

However, this "natural" order is typically destroyed by the so-called *reordering* of the link, which is performed in subroutine REORDER_LIAISON of module M_LINKS_ADD, which is called from ADD_TRUE_LIAISON in the same module, that is called from ADD_PINB_LINK within the routine BL_LINK_PINBALLS that builds up the pinball links in module M_PINBALLS.

The links reordering (which is different from and should not be confused with the links *renumbering*), consists simply in arranging the DLIE (linked dofs) and the COEF (corresponding link coefficients) arrays (but *not* the PLIE array of the linked nodes) of a link in *growing* order. Apparently, this particular ordering is needed by the links renumbering algorithm. The reordering process acts within a single link, while the renumbering process alters the order in which the list of links is traversed (without re-arranging the list itself).

Therefore, in a data structure of type LIAISON (a link), after the reordering process:

- The PLIE array (linked nodes) remains in the natural order (as when the link is first written).
- The DLIE array is in growing order.
- The COEF array is in growing order (matching DLIE).
- The RANG array allows to pass from the growing to the natural order. RANG(I), I=1,NDLIE (with NDLIE the total number of dofs in this link) is the rank that the I-th dof, as listed in the growing order list DLIE, would have in a naturally ordered list of dofs (say DNAT, which is not available in the data structure) according to the natural list of nodes PLIE.

Thus, the RANG array is, besides the knowledge of the number of nodes of the A pinball n_A, the key to evaluating $\boldsymbol{R}_s$ according to (62). The RANG is added, together with NDLIE, to the exchange list of the LINK_FRNORM_PINB routine, while n_A is added to the LDATA array of the link and is passed to the LINK_FRNORM_PINB routine as NBNA (first argument) in place of ISLAVE.

Further details on this procedure may be found in reference [31].

3.2.4 The penalty-based `PINB` friction model

The implementation of friction in the uncoupled, penalty-based pinball contact model (`PINB PENA`) is somewhat simpler than in the fully coupled, Lagrangian multiplier based pinball model. The modifications take place in routine `PENA_CONTACT_FORCES` of module `M_PINBALLS`.

For each detected inter-penetrating couple of pinballs, the *normal* penalty force $\boldsymbol{F}_n$ is computed first, independently from the presence or not of friction. This force is assumed to act at the centre of the corresponding pinball, and is distributed to the nodes of the element containing the pinball according to the element's shape functions in order to obtain nodal force contributions.

Next, if friction has been specified, the *tangent* (friction) force $\boldsymbol{F}_t$ is evaluated. Again, this force is assumed to act at the center of the corresponding pinball and is distributed to the nodes similarly to the normal contact force.

To perform the friction calculations, the relative velocity $\boldsymbol{v}_r$ between the two contacting bodies is evaluated first. Like in the case of forces, this velocity is evaluated from the velocities at the centres of the two inter-penetrating pinballs, obtained by interpolation from the nodal velocities. By indicating with A and B the two pinballs and by N_A and N_B the corresponding element shape functions at the pinball centre one has:

$$\boldsymbol{v}_A = \sum N_A \boldsymbol{v}_{IA} \qquad \boldsymbol{v}_B = \sum N_B \boldsymbol{v}_{IB} \tag{43}$$

$$\boldsymbol{v}_r = \boldsymbol{v}_A - \boldsymbol{v}_B \tag{44}$$

The relative velocity $\boldsymbol{v}_r$ is then separated into normal and tangential vectors. If $\hat{\boldsymbol{n}}_{AB}$ is the unit *contact normal*, oriented from A towards B, then:

$$\boldsymbol{v}_{rn} = (\boldsymbol{v}_r \cdot \hat{\boldsymbol{n}}_{AB})\, \hat{\boldsymbol{n}}_{AB} \tag{45}$$

$$\boldsymbol{v}_{rt} = \boldsymbol{v}_r - \boldsymbol{v}_{rn} \tag{46}$$

Then, if the tangent relative velocity is not vanishing ($|\boldsymbol{v}_{rt}| > 0$), then the tangent direction is taken along it:

$$\hat{\boldsymbol{t}} = \boldsymbol{v}_{rt} / |\boldsymbol{v}_{rt}| \tag{47}$$

Otherwise $|\boldsymbol{v}_{rt}| = 0$. In this case the internal relative force is evaluated, by interpolating the internal forces at the pinball centres similarly to the velocities, and is then separated into normal and tangential vectors:

$$\boldsymbol{F}_A^{\text{int}} = \sum N_A \boldsymbol{F}_{IA}^{\text{int}} \qquad \boldsymbol{F}_B^{\text{int}} = \sum N_B \boldsymbol{F}_{IB}^{\text{int}} \tag{48}$$

$$\boldsymbol{F}_r^{\text{int}} = \boldsymbol{F}_A^{\text{int}} - \boldsymbol{F}_B^{\text{int}} \tag{49}$$

$$\boldsymbol{F}_{rn}^{\text{int}} = \left(\boldsymbol{F}_r^{\text{int}} \cdot \hat{\boldsymbol{n}}_{AB}\right) \hat{\boldsymbol{n}}_{AB} \tag{50}$$

$$\boldsymbol{F}_{rt}^{\text{int}} = \boldsymbol{F}_r^{\text{int}} - \boldsymbol{F}_{rn}^{\text{int}} \tag{51}$$

If the tangent internal relative force is not vanishing ($\left|\boldsymbol{F}_{rt}^{\text{int}}\right| > 0$), then the tangent direction is taken along it:

$$\hat{\boldsymbol{t}} = \boldsymbol{F}_{rt}^{\text{int}} / \left|\boldsymbol{F}_{rt}^{\text{int}}\right| \tag{52}$$

otherwise, we set $\hat{\boldsymbol{t}} = \boldsymbol{0}$ (this will result in a null tangential force).

Finally, the tangential (friction) contact force $\boldsymbol{F}_t$ is evaluated as follows. If the tangent relative velocity is not vanishing ($|\boldsymbol{v}_{rt}| > 0$), then we are in the *kinetic friction* case. The tangential force is taken equal to:

$$|\boldsymbol{F}_t| = \mu \, |\boldsymbol{F}_n| \tag{53}$$

with μ given from (3), and is supposed to act along the tangent $\hat{\boldsymbol{t}}$ (which in this case has the same direction as the tangent relative velocity), in the opposite sense to the tangent relative velocity.

Else, we are in the *static friction* case. The tangential force is taken equal to the tangent internal relative force, but limitedly to a value μ times the normal contact force:

$$|\boldsymbol{F}_t| = \min\left(\left|\boldsymbol{F}_{rt}^{\text{int}}\right|, \mu \, |\boldsymbol{F}_n|\right) \tag{54}$$

and is supposed to act along the tangent $\hat{\boldsymbol{t}}$ (which in this case has the same direction as the tangent internal relative force), in the opposite sense to the tangent internal relative force.

Note that, in the kinetic friction case, the fact of taking a friction force depending only on the normal contact force might produce ping-pong numerical effects, i.e. the reversal of the relative velocity over a single step, which might in turn lead to strong oscillations and eventually to numerical instabilities (this will have to be verified in the numerical examples). The present approach is quite simplistic, but this is in line with the simplicity of the penalty formulation which is used also for the evaluation of the normal contact force.

This should be compared with the coupled approach based upon Lagrange multipliers. In that case, the fact of computing the tangent friction force by a link which imposes the equality of tangent velocities (i.e., zero relative tangent velocity) might help avoid (or reduce) the ping-pong effects.

Another source of uncertainties and of possible pathological behaviour in complex applications is the fact that the contact force evaluations are done at the pinball centres rather than directly at nodes (like in the `GLIS` model, at least as concerns the slave node), and this involves interpolation. This aspect, however, is common to both the coupled (Lagrange multipliers) and uncoupled (penalty) formulations of the pinball model.

Finally, the situation becomes more complex and the effective working of the algorithms becomes more obscure in the case of *hierarchic* pinballs, because these tend generate multiple contact conditions affecting the same set of nodes.

As a final remark, it might be worthwhile upgrading the penalty-based friction algorithm similarly to the GLIS and LM-PINB cases, namely by using a two-tangent model rather than a single-tangent model (in 3D). However, the necessity of such an upgrade is still being investigated and the development (if necessary) is left for a future work.

3.3 The `GPIN` model

A first draft of the (coupled, LM-based version of the) `GPIN` contact model has been recently implemented in EPX, see [22].

3.3.1 The LM-based `GPIN` friction algorithm

The LM-based version of the friction algorithm for the `GPIN` contact model closely resembles that of the `GLIS` model described in Section 3.1.1, apart from a few details, that are discussed below.

Influence of the rebound algorithm

A distinctive feature of the `GPIN` model, which is found also in `PINB` but not in `GLIS`, is the possibility of choosing among various types of *rebound algorithm* (see [22] for full details):

- The *simplified a priori* rebound algorithm, whereby a penetration rate $\dot{p}$ is estimated simply based upon the current velocities. Then, any constraint with $\dot{p} \leq 0$ is removed from the system of constraints *before* solving the linear problem. This algorithm can be activated by `OPTI GPNS REB0`.

- The *a priori* rebound algorithm, where the penetration rate for the preliminary elimination of constraints is estimated in a more sophisticated way, based upon the estimated "free" position of the GPINs at the next time step in the absence of contact forces. This is the default, or it can be explicitly activated by `OPTI GPNS REB1`.

- The *a posteriori* rebound algorithm where constraints are retained or withdrawn after the solution of the linear system, based upon the sign of the corresponding Lagrange multiplier. A positive multiplier ($\lambda > 0$) indicates incipient rebound (traction between the two bodies) and leads to elimination of the corresponding constraint by setting $\lambda = 0$. This algorithm can be activated by `OPTI GPNS REB2`.

- Do not apply any contact rebound detection algorithm. This option, which is useful only for testing and debugging, can be activated by `OPTI GPNS NORB`.

Note that `PINB` does not have the simplified a priori rebound algorithm (`REB0`), while `GLIS` admits no choice and always uses an a posteriori rebound algorithm, based upon the sign of the Lagrange multiplier. As a matter of fact, in `GLIS` the rebound algorithm is embedded in (and cannot be separated from) the friction algorithm, see point 1. of the procedure detailed in Section 3.1.1.

Thus for `GPIN` the question arises of how to reconcile the chosen rebound algorithm with the friction algorithm, which is basically the same as in `GLIS` and therefore already embeds an a posteriori rebound check. If the a posteriori rebound is selected in `GPIN` (like always in `GLIS`), then there is no problem and the friction procedure detailed in Section 3.1.1 can be used without modification. If no rebound is chosen (an "extreme" option only useful for very exotic tests and for debugging purposes), then the a posteriori rebound check at point 1. of the friction algorithm (see Section 3.1.1) is simply dropped.

It remains to decide what to do when an a priori rebound algorithm (either simplified or not) is chosen in a `GPIN` calculation. One possibility is to maintain also in this case the a posteriori rebound check embedded in the friction algorithm, thus resulting into a sort of *combined pre-post* rebound algorithm. The other possibility is to drop the a posteriori check by noting, however, that if friction is present in a situation with $\lambda > 0$, a tangential friction force will also act in addition to the (attractive) normal force, hopefully not destroying the entire numerical solution.

Since in principle it is difficult to foresee what would be the best strategy (which might also depend on the particular application), the possibility of choosing between the two mentioned procedures is left as an additional option (for experimenting): `OPTI GPNS REBC` activates the above mentioned *combined pre-post* form of a (nominally) a priori rebound algorithm while by default (no additional option) the "genuine" version of the a priori rebound is applied. Clearly, the presence or not of the `REBC` keyword only has an influence if the a priori rebound algorithm, either simplified (`REB0`) or not (`REB1`, default), has been chosen in the first place.

Slave node and master nodes

Another difference between the `GPIN` and the `GLIS` models concerns the nature (classification) of the nodes involved in a contact. In `GLIS` one distinguishes between slave node and master nodes, while this distinction is not needed in `GPIN`. The notion of slave node is explicitly used in the `GLIS` friction algorithm of Section 3.1.1, so one must decide how to proceed when applying this algorithm to `GPIN`.

In the `GPIN` model we have two main classes of contacts (see [22] for details), whereby each contact involves two GPINs:

- P-O contacts where P stays for a point GPIN or P-GPIN (in practice, a node), and O stays for an "other" type of GPIN, which can be either a P-GPIN itself, or an L-GPIN (a two-noded line), or a T-GPIN (a three-noded triangle) or a Q-GPIN (a four-noded quadrangle).
- L-L contacts, involving two L-GPINs. Such contacts may only occur in 3D.

The first situation (P-O) closely resembles that of `GLIS` and, as far as the friction algorithm is concerned, we can simply assume the (node of the) first P-GPIN plays the role of the slave node while the (nodes of the) other GPIN (of which there are 1, 2, 3 or 4) are the master nodes, in `GLIS` terminology.

In the second situation (L-L) it is less evident how to proceed. We arbitrarily decide to consider the first L-GPIN as slave and the second one as master. We do not have a (single) slave node in this case, because each L-GPIN has *two* nodes, but we do have a slave *point*, the point of the first L-GPIN at which the contact occurs. This point is known from the position of the two nodes of the L-GPIN through suitable interpolation functions. Therefore, the friction algorithm will have to be slightly adapted for the case of L-L contact between GPINs.

The 2D case

The final difference between `GPIN` and `GLIS` is that `GPIN` is formulated in both 2D and 3D, while `GLIS` is only 3D.

In the 2D case, the friction algorithm has to be adapted since in that case there is only one tangential direction $\hat{\boldsymbol{\tau}}$, and thus only one tangential constraint, instead of two. The unit tangent vector $\hat{\boldsymbol{\tau}}$ is obtained by rotating clockwise the normal vector $\hat{\boldsymbol{n}}$ by 90 degrees.

Final form of the friction algorithm for `GPIN`

This results in the following algorithm (in 3D). In the case with friction, for each detected contact (penetration) we have *three* constraints: two (provisional) constraints J_1 and J_2 along the tangents $\hat{\boldsymbol{\tau}}_1$ and $\hat{\boldsymbol{\tau}}_2$, associated with Lagrange multipliers λ_{J_1} and λ_{J_2}, respectively, followed by a constraint I along the normal $\hat{\boldsymbol{n}}$, of Lagrange multiplier λ_I. In the implementation, it is $I = J_1 + 2 = J_2 + 1$, that is, in the list of links the first tangent constraint is directly followed by the second tangent constraint, which in turn is followed directly by the corresponding normal constraint.

We consider as slave the first GPIN and as master the second GPIN involved in each contact. We define the *slave point* as the node of the first GPIN, if this is a P-GPIN, or the contact point (not a node, in general) on the first GPIN, if this is an L-GPIN. If we denote $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ the (positions of the) nodes of the first L-GPIN, N_1 and N_2 the interpolating (shape) function of the segment $\boldsymbol{P}_1\boldsymbol{P}_2$ evaluated at the contact point (which are available from the GPIN contact detection algorithm), then the position of the slave point is given by:

$$\boldsymbol{s} = N_1\boldsymbol{P}_1 + N_2\boldsymbol{P}_2 \tag{55}$$

We define as master nodes the nodes of the second GPIN involved in each contact.

The system of links is solved by the Lagrange multipliers method, thus obtaining the three Lagrange multipliers λ_{J_1}, λ_{J_2} and λ_I. The reactions are *provisionally* computed by:

$$\boldsymbol{R} = \boldsymbol{C}^T\boldsymbol{\lambda} \tag{56}$$

where $\boldsymbol{C}$ is the matrix of constraint coefficients. Next:

1. We check the normal constraint for rebound. If $\lambda_I \geq 0$, then there would be traction along the normal, i.e. rebound is occurring. We set $\lambda_I = 0$ and $\lambda_{J_1} = \lambda_{J_2} = 0$ (if present), thus

eliminating *all three* constraints from the system, and we re-compute the reactions by (56). The reactions concerning the contact under consideration will vanish out because the corresponding multipliers are zero.

2. Else, $\lambda_I < 0$ and contact (not rebound) is actually occurring:
   - If no friction is involved in this contact (so that neither J_1 nor J_2 constraint is present), then the computed contact reactions (along the normal) for both the slave and the master nodes are already correct and the calculation is complete.
   - Otherwise, there is friction. The reactions along the normal are already the final ones, but those in the tangent plane need to be corrected if they do not satisfy Coulomb's condition. Let R_{sn} denote the (final) normal reaction and $R_{s\tau_1}$, $R_{s\tau_2}$ the (provisional) tangential reactions at the slave point s resulting, respectively, from the constraints I, J_1 and J_2. Compute the magnitude of the tangential reaction:

$$R_{s\tau} = \sqrt{R_{s\tau_1}^2 + R_{s\tau_2}^2} \tag{57}$$

     Then:
     - If $R_{s\tau} \leq \mu R_{sn}$, then the Coulomb condition is satisfied. All the computed reactions are already correct and the calculation is complete.
     - Else $R_{s\tau} > \mu R_{sn}$ and the Coulomb condition is not satisfied, i.e. the provisional tangential reaction force causes the provisional total reaction force lie outside Coulomb's cone. We evaluate a *reduction coefficient* α:

$$\alpha = \frac{\mu R_{sn}}{R_{s\tau}} < 1 \tag{58}$$

       and then we scale the tangential reactions accordingly. To this end, we scale down the Lagrange multipliers of the tangential constraints:

$$\lambda_{J_1} \Leftarrow \alpha\lambda_{J_1} \qquad \lambda_{J_2} \Leftarrow \alpha\lambda_{J_2} \tag{59}$$

       which has the following effect. For the slave point we obtain, formally:

$$\boldsymbol{R}_{s\tau} \Leftarrow \alpha\boldsymbol{R}_{s\tau} \tag{60}$$

       If the slave point is not a node (case of the L-L contact), this is reflected also on the reactions at the two slave nodes P_1 and P_2:

$$\boldsymbol{R}_{P_1\tau} \Leftarrow \alpha\boldsymbol{R}_{P_1\tau} \qquad \boldsymbol{R}_{P_2\tau} \Leftarrow \alpha\boldsymbol{R}_{P_2\tau} \tag{61}$$

       Similarly, also the tangential reactions on the master nodes m_i result scaled down by the same factor α:

$$\boldsymbol{R}_{m_i\tau} \Leftarrow \alpha\boldsymbol{R}_{m_i\tau} \qquad i = 1, \cdots, M \tag{62}$$

In the 2D case with friction, for each detected contact we have *two* constraints: one (provisional) constraint J along the tangent $\hat{\boldsymbol{\tau}}$, associated with Lagrange multiplier λ_J, followed by a constraint I along the normal $\hat{\boldsymbol{n}}$, of Lagrange multiplier λ_I. In the implementation, it is $I = J + 1$, that is, in the list of links the tangent constraint is directly followed by the corresponding normal constraint.

We consider as slave the first GPIN and as master the second GPIN involved in each contact. We define the *slave point* as the node of the first GPIN, which in 2D is always a P-GPIN. We define as master nodes the nodes of the second GPIN involved in each contact (1 or 2 nodes in 2D).

The rest of the algorithm is perfectly analogous to the 3D case, but with only one tangent direction instead of two. Thus, the tangent reaction $R_{s\tau}$ is computed directly and not through Eq. (57).

3.3.2 The penalty-based `GPIN` friction algorithm

At the moment of this writing, there exists no penalty-based version of the `GPIN` contact model. When one is formulated and implemented, the friction model shall be added to it similarly to the `GLIS` and `PINB` contact models.

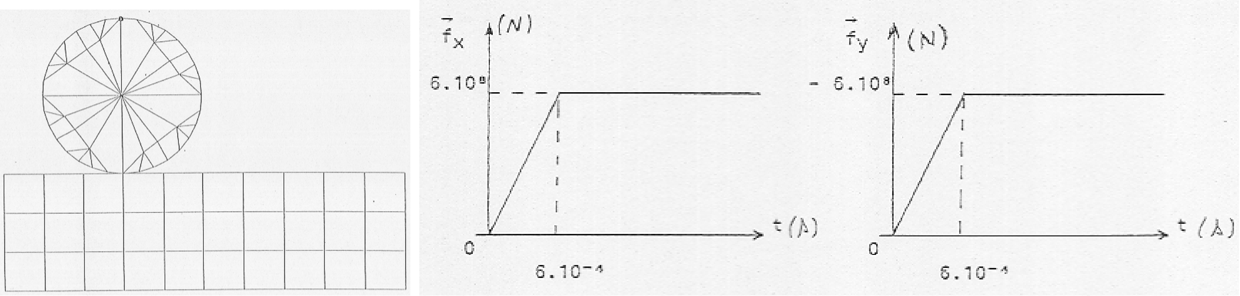
4 Numerical examples using LM-based contact models

We start by attempting to reproduce the interesting tests from reference [25], obtained exactly 30 years ago with the PLEXUS code (ancestor of EPX). The tests (the first one in 2D, the second one in 3D) represent the sliding, without or with friction, of a solid circular disk along a plane. The Cast3m mesh generation files are not available, so the meshes have been tentatively reconstructed from the data (some of which appear to be slightly inconsistent) and from the pictures contained in the report. Fortunately, the EPX files were listed in the report. They have been slightly adapted to conform to current EPX syntax.

4.1 Sliding disk problem definition in 2D and original solutions from [25]

In the 2D problem the disk (projectile) has a radius $R = 2$ m, with 32 elements along the circumference, meshed by CAR4 and TRIA elements. The plane (target) is represented by a rectangle of length $L = 10$ m and height $h = 3$ m. It is meshed by 10×3 regular square elements (CAR4) of side 1 m. The disk is placed initially in contact with the plane (zero gap) and at a position of 3 m along the plane, see Figure 2(a). The lower base of the target is blocked in both directions. The initial velocity is zero everywhere. A plane strain (`DPLA`) formulation is assumed.

The material is assumed linear elastic (LINE) for both bodies, with $\rho = 7800$ kg/m^3, $E = 2 \times 10^{11}$ Pa, $\nu = 0$. There are two time-dependent forces applied to the centre of the disk: a horizontal force of nominal value $F_X = 6 \times 10^8$ N and a vertical force of nominal value $F_Y = -6 \times 10^8$ N. Both forces start from zero at the initial time $t_0 = 0$, grow up linearly until they reach the nominal value at $t_m = 0.6$ ms and then remain constant until the final time $t_f = 60$ ms, as shown in Figure 2(b).



(a) Initial mesh

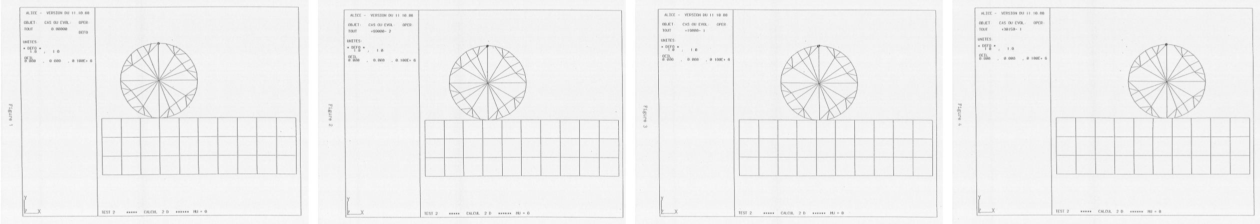
(b) Applied forces

Figure 2: Sliding disk problem in 2D (from [25])

The calculation is performed with a constant step $\Delta t = 5$ μs, requiring 12 000 steps until t_f. A first simulation is performed without friction. In the second simulation friction is assumed with $\mu_s = \mu_k = 0.25$ and $\gamma = 0$ (according to the problem description in the report [25]). In the input file listed in the same report we find $\gamma = 0.1$ instead. However, this discrepancy is irrelevant since when $\mu_s = \mu_k$ Eq. (3) gives a constant friction coefficient ($\mu = 0.25$ here), irrespective of the actual value chosen for γ.

Figure 3 shows the original result obtained in the case without friction, while Figure 4 shows the original result obtained in the case with friction. In each image a small hand-made dot marks the position of the point initially located at the topmost position of the disk, in order to be able to appreciate the rotation of the disk (when present).

It can be seen that in the first solution the disk slides along the plane (almost) without any rotation, while in the second case it rotates (due to friction), although some sliding seems to remain especially in the first part of the motion (perhaps due to the fact that the vertical force is not yet big enough to cause enough friction in the initial period). Also, the overall translation of the disk is larger in the case without friction.


(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 30$ ms

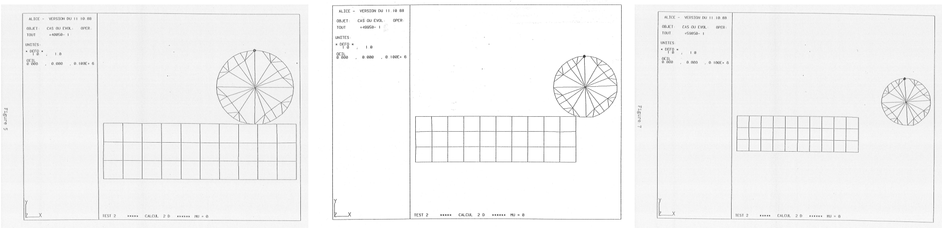
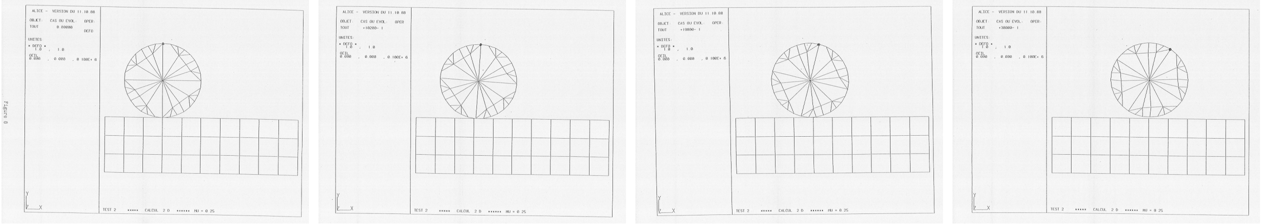

(e) $t = 40$ ms (f) $t = 50$ ms (g) $t = 60$ ms

Figure 3: Result of the sliding disk problem in 2D without friction (from [25])


(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 30$ ms

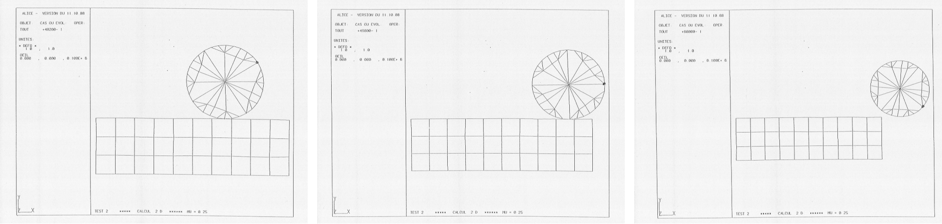

(e) $t = 40$ ms (f) $t = 50$ ms (g) $t = 60$ ms

Figure 4: Result of the sliding disk problem in 2D with friction (from [25])

4.2 Solutions with EPX using `GLIS`

The calculations performed with EPX by using the `GLIS` contact model (so-called *sliding lines* in 2D) are summarized in Table 1.

Case	Mesh	Description	Final time [ms]	Steps	CPU [s]
FROT23	222 CAR4 1 PMAT	`GLIS`, no friction	60	30 000	18.5
FROT24	222 CAR4 1 PMAT	`GLIS`, with friction	60	30 000	18.4

Table 1: Calculations for the sliding disk problem in 2D using `GLIS`.

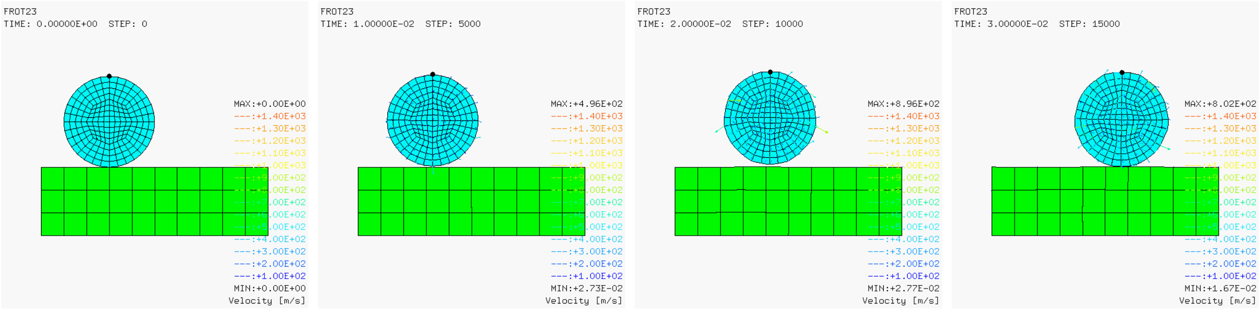
4.2.1 Case FROT23

This is an attempt at reproducing with EPX the 2D solution without friction. The mesh is shown in Figure 5(a). Note that the disk has been meshed more regularly than in the original solution, by using exclusively quadrangles. Although the same subdivision of the disk circumference as in the original mesh has been maintained (32 segments), the internal mesh of the disk uses more and smaller elements. The constant time step used in this simulation has therefore been reduced to $\Delta t = 2.0$ μs, resulting in 30 000 steps to reach the final time $t_f = 60$ ms. This is a bit strange since the code reports a stability step of 23 μs. However, attempts at obtaining a solution with either $\Delta t = 20.0$ μs or $\Delta t = 5.0$ μs fail (élément croisé).

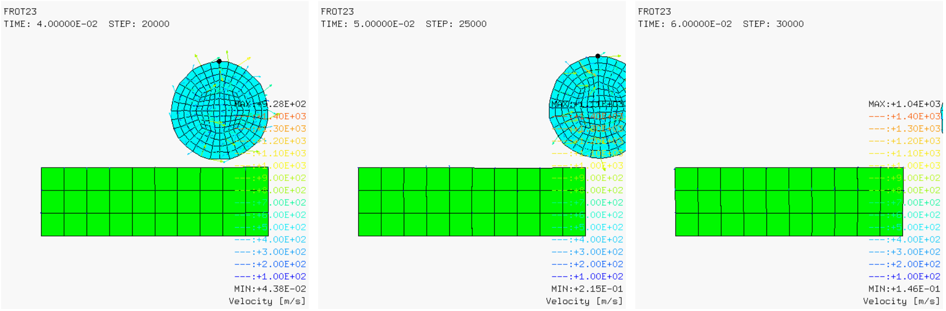
Using the `GLIS` model in 2D (*lignes de glissement*) is notoriously difficult in EPX due to its archaic input syntax. The model has apparently not yet been converted to the new links formulation (`LINK COUP`) and so use has to be made of the old `LIAI` directive (which even requires a dimensioning for the sliding lines):

```
DIME GLIS 1 200 TERM
 . . .
!LINK COUP
LIAI
     BLOQ 12 LECT bas TERM
     GLIS 1
          MAIT LECT pb2 TERM
          ESCL LECT pc1 pc2 pc3 pc4 TERM
```

A zero-mass material point (`PMAT`) is attached to the upper point of the disk in order to appreciate the disk rotation. It appears as a small black dot in the pictures, see Figure 5.



(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 30$ ms

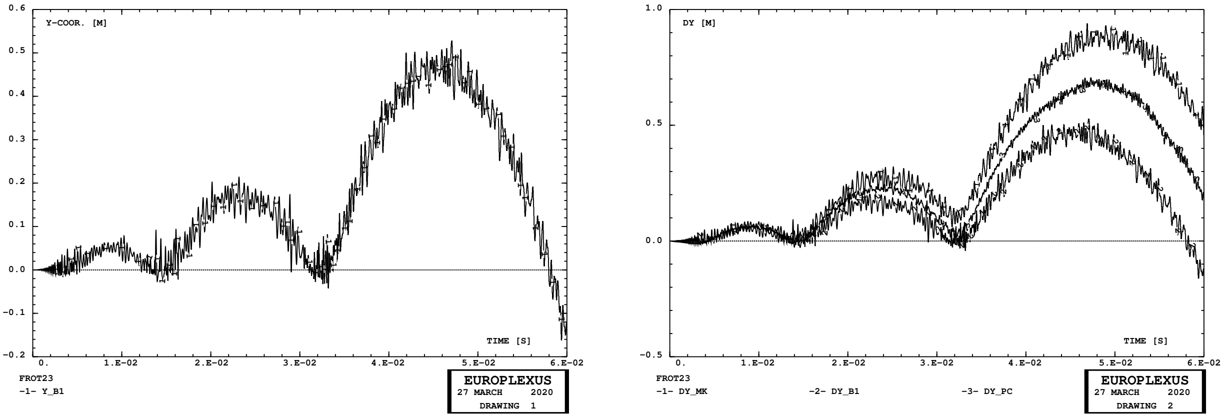


(e) $t = 40$ ms (f) $t = 50$ ms (g) $t = 60$ ms

Figure 5: Result of test FROT23

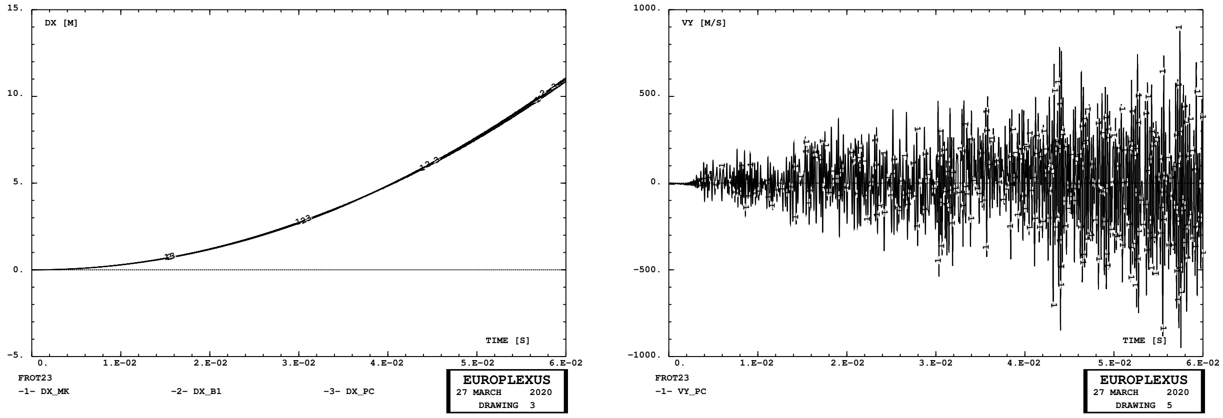
Other results of this calculation are presented in Figure 6. The first image shows the vertical position of the lowest point of the disk (i.e., the point initially in contact with the plane). The second image shows the vertical displacements of three points of the disk: the lowest one, the central one and the highest one (marker). There seems to be an excessive vertical deformation of the disk since the displacements are largely different (and there is only very little rotation). The next image shows the horizontal displacements of the same points, which are almost equal (very small rotation). The last picture shows the vertical velocity of the disk centre, and is very noisy.

The solution (especially when looking at an animation) appears more “bouncy” than the original one, but this might depend on the limited number of frames available in the original report. At certain moments the disk slightly penetrates the plane and at others it detaches considerably.



(a) Vertical coordinate

(b) Vertical displacements



(c) Horizontal displacements

(d) Vertical velocity

Figure 6: Further results of case FROT23

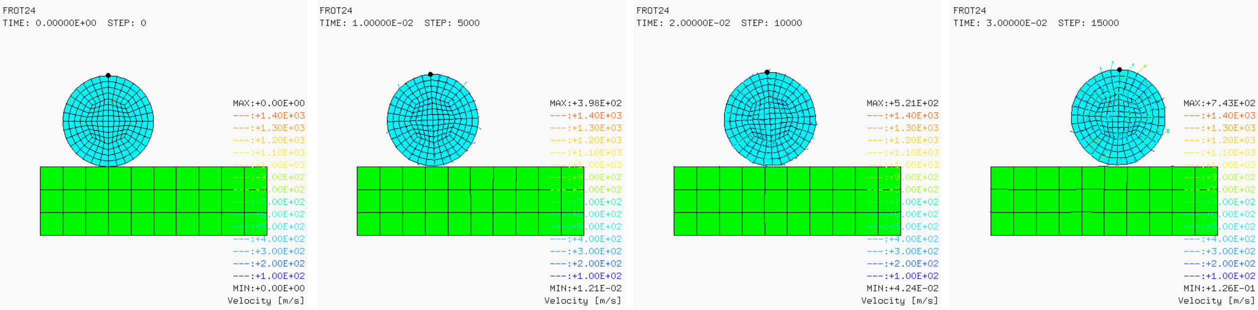
Overall, there is (almost) no rotation, only translation. Since a relatively large constant horizontal force is applied during the second part of the transient (from t_m to t_f), the disk reaches a large horizontal velocity, of the order of 350 m/s (but with large oscillations) at the end of the transient.

4.2.2 Case FROT24

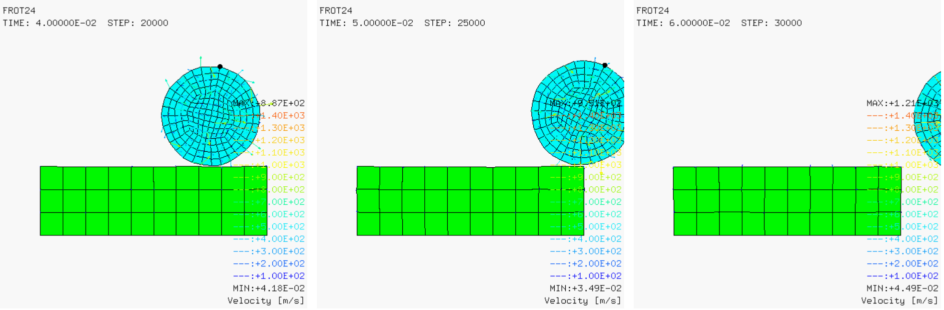
This is a repetition of the previous solution by activating the friction. The friction is tentatively specified as part of the complements of geometry `COMP` directive, as in the original PLEXUS input from reference [25], and not as part of the `GLIS` directive itself (like in 3D tests using `LINK GLIS`).

```
COMP EPAI 0.2 LECT mark TERM ! Only for visualization
     NGRO 1 'b1' LECT circ TERM COND NEAR POIN 3 0
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
     FROT 1 MU0 0.25 MU1 0.25 GAMM 0.1
```

As it appears from the solution, shown in Figure 7, there is a certain rotation of the disk, but much less than in the original solution. For example, at $t = 50$ ms, the disk appears to have rotated about 20 degrees, while it had rotated almost 90 degrees in the original solution. On the contrary, the translation in this solution is larger than in the reference. This may also be an indication that there is less overall contact than in the reference between the disk and the plane (“bouncy” solution). Other results of this calculation are presented in Figure 8.



(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 30$ ms



(e) $t = 40$ ms (f) $t = 50$ ms (g) $t = 60$ ms

Figure 7: Result of test FROT24

Overall, the solutions FROT23 and FROT24 obtained with `GLIS` in 2D look very poor. However, this might depend on the fact that using the sliding lines model is difficult and requires some insight that is perhaps beyond the limits and scope of the present report.

4.3 Solutions with EPX using `GPIN`

The calculations performed with EPX by using the `GPIN` contact model in 2D are summarized in Table 2.

Case	Mesh	Description	Final time [ms]	Steps	CPU [s]
FROT81	222 CAR4 1 PMAT	No friction	60	3000	21.3
FROT82	222 CAR4 1 PMAT	With friction	60	3000	22.0
FROT83	212 CAR4 1 PMAT	3D equivalent, no friction	40	2000	14.7
FROT84	212 CAR4 1 PMAT	3D equivalent, with friction	40	2000	15.3
FROT91	212 CAR4 1 PMAT	Idem 83, `SPLT NONE`	40	2000	20.5
FROT92	212 CAR4 1 PMAT	Idem 84, `SPLT NONE`	40	2000	15.9

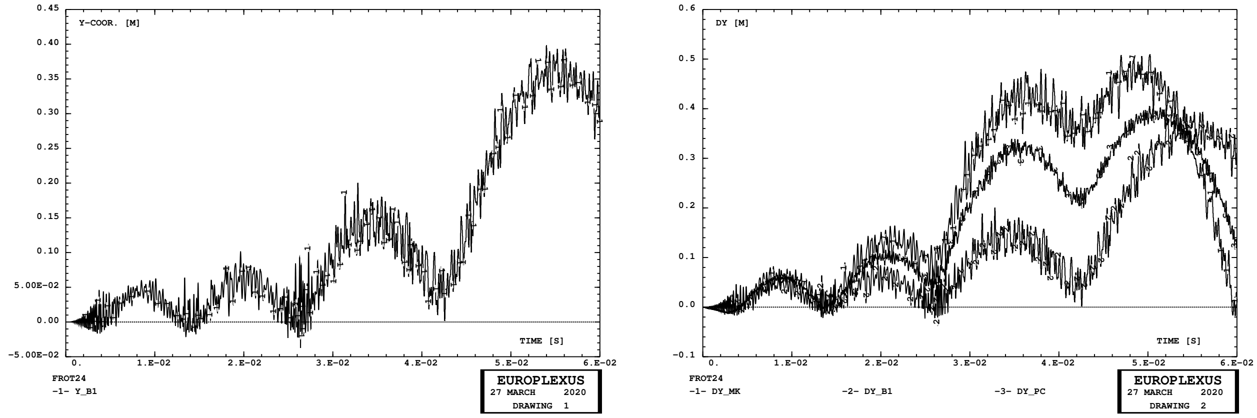
Table 2: Calculations for the sliding disk problem in 2D using `GPIN`.

4.3.1 Case FROT81

This is the equivalent of case FROT23 by using `GPIN` instead of `GLIS` for the contact:

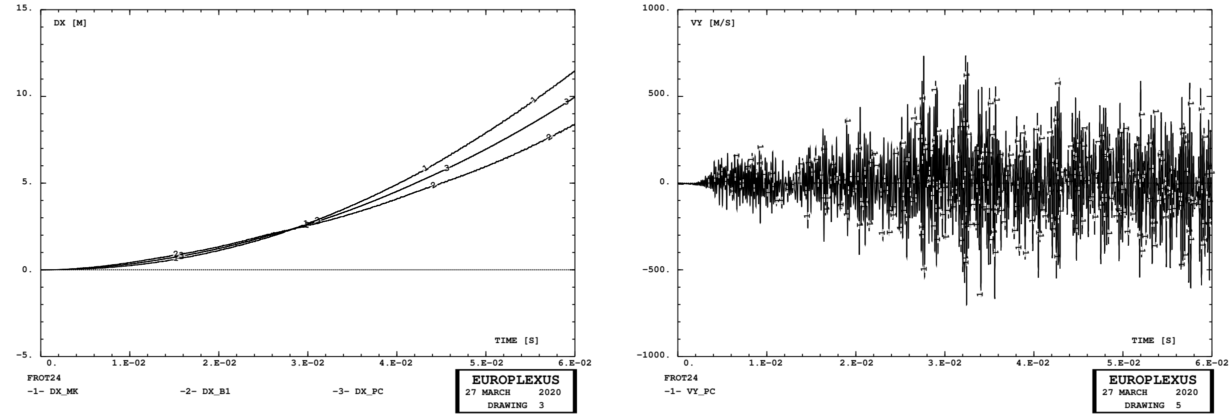
```
LINK COUP
     BLOQ 12 LECT bas TERM
     GPIN BODY LECT circ TERM
          BODY LECT base TERM
          DIAM 0.05 LECT cir basup TERM
```

A stable solution is obtained by a time step $\Delta t = 20$ μs, ten times larger than with `GLIS`, thus requiring 3000 steps (instead of 30 000) to reach the final time $t_f = 60$ ms, see Figure 9. The thick red lines in the pictures are the link joints (they connect the liked nodes). The fixed-length green arrows



(a) Vertical coordinate

(b) Vertical displacements



(c) Horizontal displacements

(d) Vertical velocity

Figure 8: Further results of case FROT24

show the linked dofs. Finally, the variable-length arrows (of color proportional to the intensity) are the link forces.

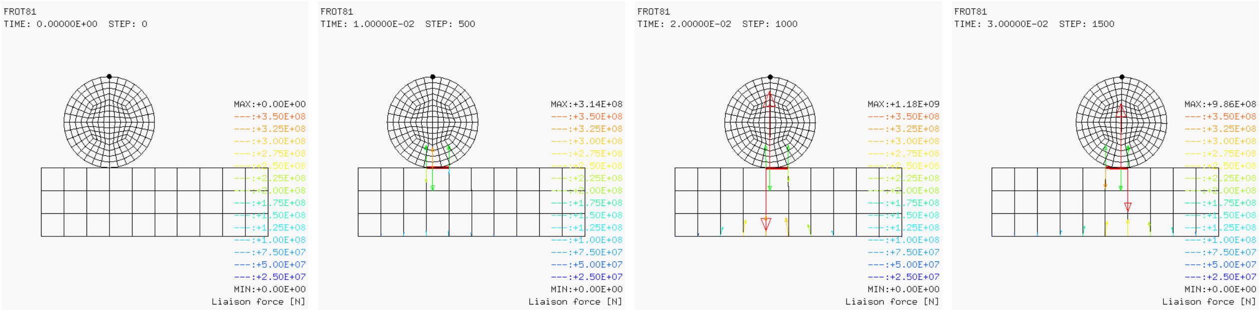
Other results of this calculation are presented in Figure 10. The curves are shown only up to $t = 40$ ms for clarity because at towards the end of the solution the disk exists from the plane and so some values (vertical displacements) become much larger.

The first image shows the vertical position of the lowest point of the disk (i.e., the point initially in contact with the plane). The second image shows the vertical displacements of three points of the disk: the lowest one, the central one and the highest one (marker). There seems to be an excessive vertical deformation of the disk since the displacements are largely different (and there is only very little rotation). The next image shows the horizontal displacements of the same points, which are almost equal (very small rotation). The last picture shows the vertical velocity of the disk centre.

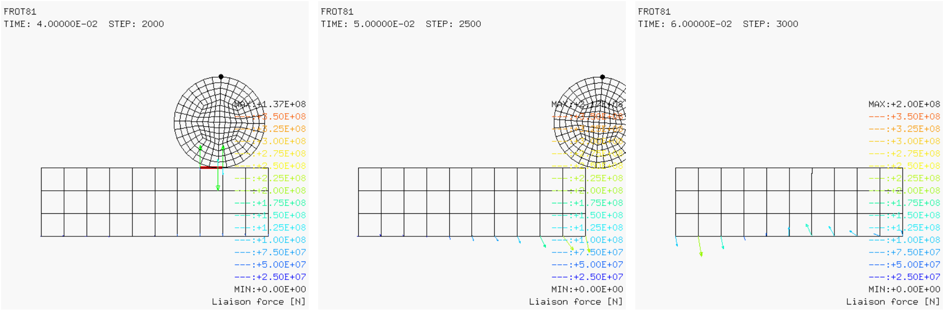
Finally, some interesting information can be obtained by activating the option `OPTI GPNS STAT`. This causes the writing on a file with the extension `.gpn` of a line at each times step containing the step number `STEP`, the time `T`, the number of raw penetrations `N_RAW`, the number of penetrations remaining after the *a priori* rebound `N_REBO`, the maximum penetration `PENEMX` during the current step, the maximum penetration `PENEMAX` so far, the maximum penetration rate `PDOTMX` during the current step, and the maximum penetration rate `PDOTMAX` so far.

A utility program `gpn2plot.exe` is written, which allows to read the `.gpn` file and to produce a corresponding `_gpn.pun` file in the EPX "punch" format. The punch file can then be read by EPX to produce the drawings of the various quantities, as shown in the next Figure for the current test case.

Figure 11(a) shows the number of penetrations at each step, mostly 0 or 1 but reaching a maximum of 2 at a particular step. Figure 11(b) shows the maximum penetration at each step. The maximum



(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 30$ ms



(e) $t = 40$ ms (f) $t = 50$ ms (g) $t = 60$ ms

Figure 9: Result of test FROT81

value is 1.75 mm and is well below the chosen radius of the GPINs $R = D/2 = 50/2 = 25$ mm. Figure 11(c) shows the maximum penetration rate at each step, mostly around 10 m/s but with an occasional maximum value of about 50 m/s.

4.3.2 Case FROT82

This solution is similar to FROT81 but we add the friction. Therefore, this case is the equivalent of case FROT24 by using `GPIN` instead of `GLIS` for the contact:

```
LINK COUP
     BLOQ 12 LECT bas TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT cir basup TERM
```

Some results are presented in Figure 12. Note the large rotation of the disk. The reaction forces ate the base of the target are much larger than in the case without friction near the end of the transient.

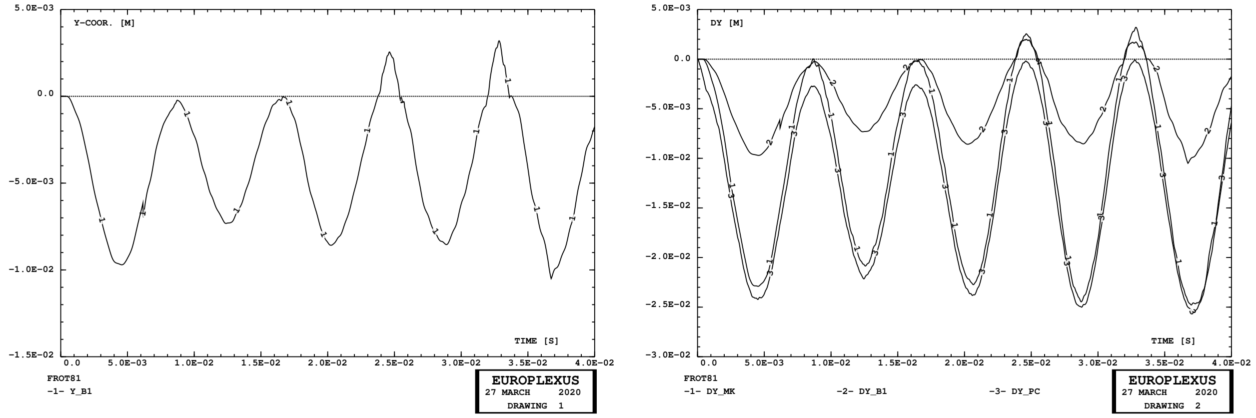
Other results of this calculation are presented in Figure 13. The solution (especially the velocity) becomes a bit noisy after 20 ms, but less than in the case with `GLIS`.

The statistics results for this calculation are presented in Figure 14 and are similar to those of case FROT81 (up to 3 penetrations, maximum penetration 1.95 mm and maximum penetration rate 70 m/s).

4.3.3 Case FROT83

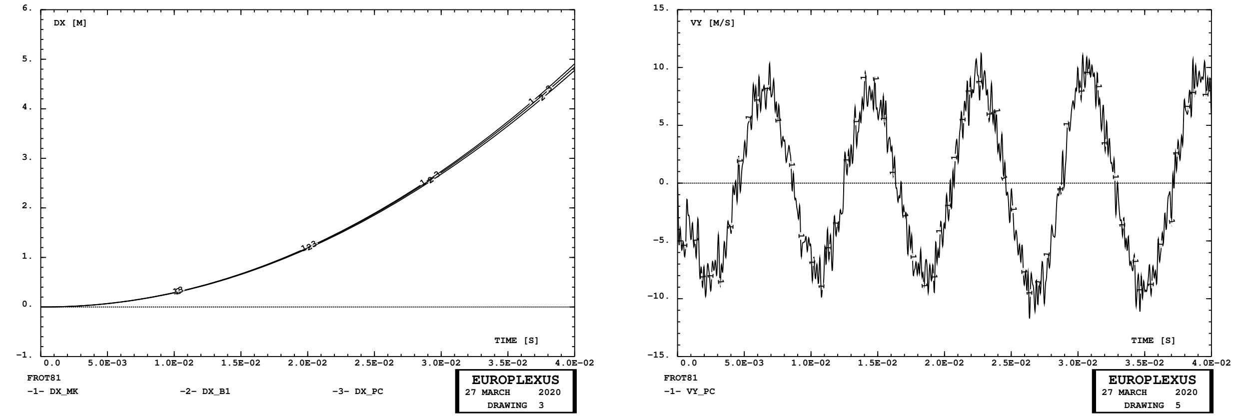
We now slightly re-formulate the 2D problem in order to obtain a version as “equivalent” as possible to the 3D version that will be presented in the following Sections. This will allow a more direct comparison between 2D and 3D solutions. The differences with respect to the original 2D problem are as follows:

- We use a plane stress formulation (`CPLA`) instead of a plane strain formulation.
- The load ramping time is set to $t_f = 1.2$ ms instead of $t_f = 0.6$.



(a) Vertical coordinate

(b) Vertical displacements



(c) Horizontal displacements

(d) Vertical velocity

Figure 10: Further results of case FROT81

- The nominal applied load (both vertical and horizontal) is set to 1/2 the total load of the 3D case because in 3D the disk thickness is $s = 2$ m while in 2D it is $s = 1$ m. Therefore, the applied load is $F_{2D} = F_{3D}/2 = 2.0 \times 10^9/2 = 1.0 \times 10^9$ N, instead of $F_{2D} = 0.6 \times 10^9$ N.
- The final time is set to $t_f = 40$ ms instead of $t_f = 60$ ms and the number of steps is set to 2000 instead of 3000.

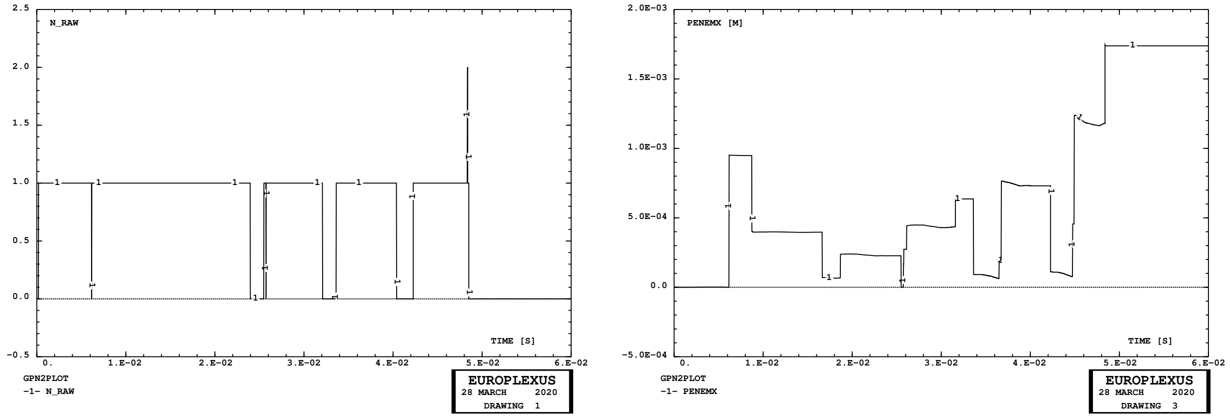
Some results of this simulation are presented in Figures 15 and 16. Figure 17 shows the statistics. The maximum penetration reached is $p_{\max} = 3.358$ mm while the maximum penetration rate is $\dot{p} = 99.49$ m/s.

The solution is as smooth as that of case FROT81 (original model) for the first 28 ms or so, but it becomes quite noisy thereafter, until the final time.

4.3.4 Case FROT84

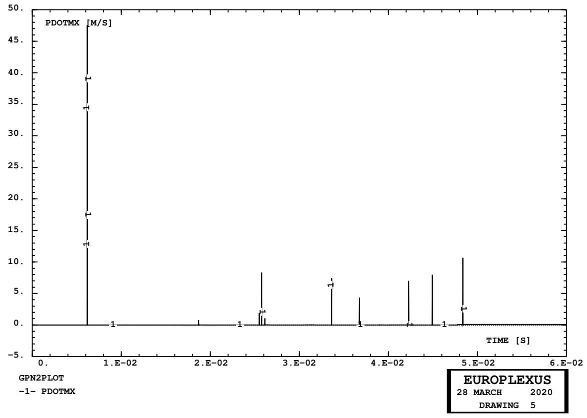
This case is similar to FROT83 but we add the friction. Some results of this simulation are presented in Figures 18 and 19. Figure 20 shows the statistics. The maximum penetration reached is $p_{\max} = 1.676$ mm while the maximum penetration rate is $\dot{p} = 34.71$ m/s.

Also in this case the solution is as smooth as that of case FROT82 (original model) for the first 28 ms or so. It becomes quite noisy thereafter, until the final time, but this happened also in the original model.



(a) Raw penetrations

(b) Max penetration at this step



(c) Max penetration rate at this step

Figure 11: Statistics results for case FROT81

4.3.5 Cases FROT91 and FROT92

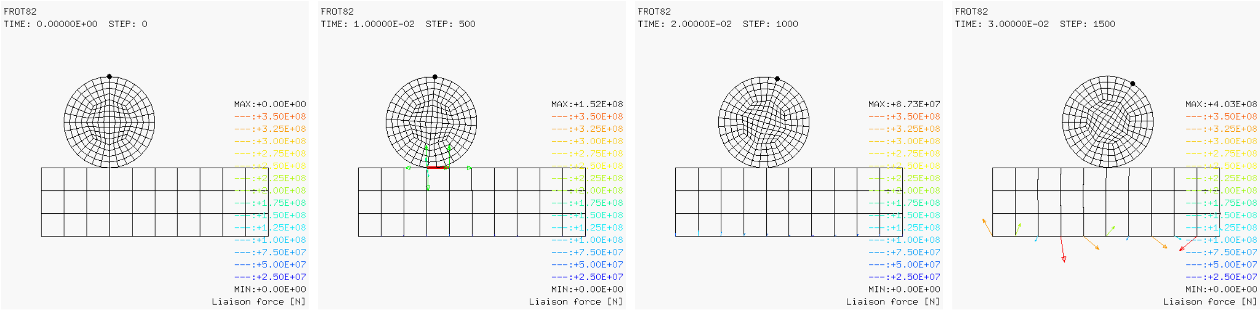
These tests are repetitions of cases FROT83 (no friction) and FROT84 (with friction), respectively, by adding the optional keywords `SPLT NONE` to the `LINK COUP` directive. Recall that, by default, the system of links is split on a number of mutually independent sub-systems which are then solved individually. The solution is done by an ad-hoc (analytical) technique whenever a sub-system contains just one link (subroutine SOLVE_SINGLE_LIAISON) or two links (subroutine SOLVE_TWO_LIAISONS). For thre or more links the sub-system is solved by the general technique in subroutine SOLVE_LISISONS_GROUP.

In the 2D case with friction, it is likely that links end up in groups of just one or two constraints. Therefore, they are probably solved in one of the dedicated routines. The above mentioned option force the system to be solved as a monolithic block. Since there are also many blockage conditions, this will force the code to use SOLVE_LISISONS_GROUP.

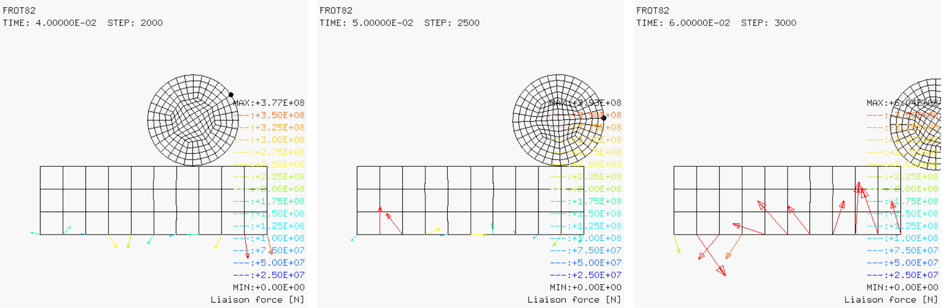
The solutions obtained are identical (by comparing the resulting `.ps` files) to those of cases FROT83 and FROT84, and are not shown for brevity. This seems to exclude errors in the special routines for just one or two links, although further checks should be conducted in this sense.

4.4 Analysis of the interlocking problems encountered

We look in detail at the results of test FROT83 in an attempt to clarify the nature of the oscillations that suddenly appear in the solution. By looking at the reaction forces (the contact forces in particular) we see that in the initial part of the solution they are quite smooth and always nearly *vertical* (i.e.,



(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 30$ ms



(e) $t = 40$ ms (f) $t = 50$ ms (g) $t = 60$ ms

Figure 12: Result of test FROT82

perpendicular to the target plane), thanks to the fact that there is no friction in this problem.

But then at step 1433, $t = 28.66$ ms, something strange happens, as shown in Figure 21. At this moment the bottom node of the projectile, which is sliding rightwards (at high relative speed) in contact with the upper face of the target, approaches one of the nodes of the target. Suddenly, the contact forces become quite bizarre, exhibiting very large *horizontal* components, see Figure 21(f).

It is like if the bottom node of the projectile and one of the top nodes of the target would become "interlocked" along the horizontal direction, which is contrary to physical intuition because of the absence of friction in the problem. In this particular case the phenomenon lasts only for a couple of steps, and at step 1435 the situation (contact forces) is back to normal (nearly vertical reactions), but meanwhile a large spurious oscillation has been injected in the solution. A similar phenomenon occurs also in the rest of the solution, nearly each time the lowest node of the projectile approaches an upper node of the target, thus perturbing more and more the numerical solution.

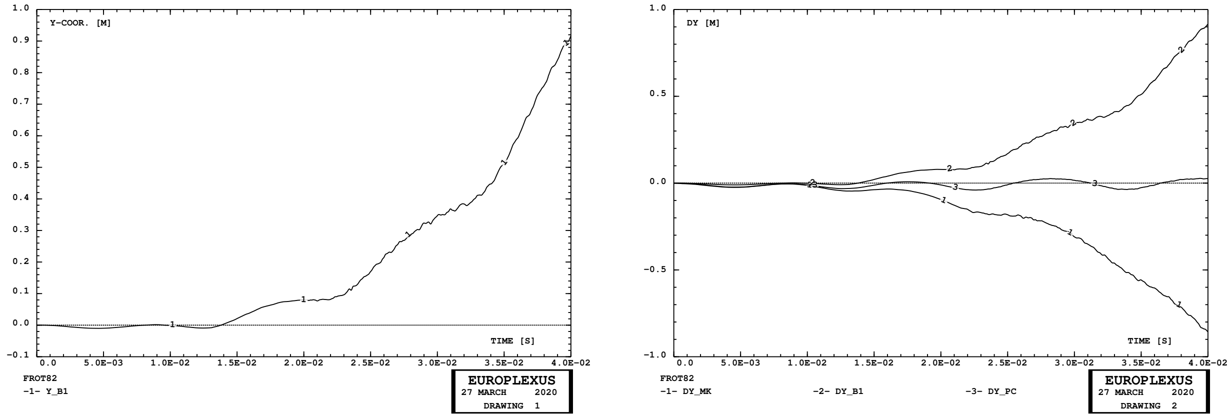
Interestingly, the same phenomenon did *not* happen before, despite the fact that the lower projectile node passed several times near an upper node of the target. However, the difference may be due to the fact that relative (horizontal) velocities between projectile and target were lower in the initial part of the solution (such velocity grows linearly during the simulation) and also to the fact that the contacting surfaces were more regular because the deformation of the bodies was lower so far.

Figure 22 shows a detail of the interlocking phenomenon at step 1433. Only the four involved elements are shown, with the corresponding GPINs. The green arrows are the reaction forces. Other details of the velocities and of the contact reaction forces are presented in Figures 23 and 24.

With reference to Figure 22(b), we see that at this step *for the first time* we have *two* P-L penetrations (while in the whole simulation so far there were either 0 or 1 penetrations):

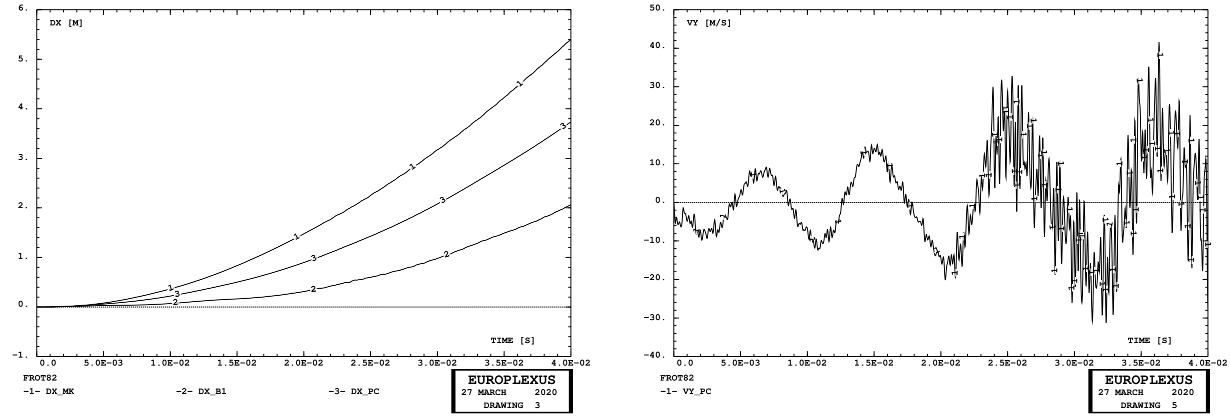
- P-GPIN P1 (red dot) penetrates L-GPIN L1 (cyan rectangle), which has nodes P2 and P3. The contact normal $\hat{\boldsymbol{n}}_1$ is $(-0.002, -1.000)$ and is almost perfectly vertical. The penetration occurs close to one of the extremities of the L-GPIN but not enough close to produce a P-P contact. Recall that in such a situation the contact normal is the normal to the L-GPIN.

- P-GPIN P2 (cyan dot) penetrates L-GPIN L2 (red rectangle), which has nodes P1 and P4. The contact normal $\hat{\boldsymbol{n}}_2$ is $(-0.059, 0.998)$ and is slightly tilted. Also in this case the penetration



(a) Vertical coordinate

(b) Vertical displacements



(c) Horizontal displacements

(d) Vertical velocity

Figure 13: Further results of case FROT82

occurs close to one of the extremities of the L-GPIN but not enough close to produce a P-P contact. The contact normal is the normal to the L-GPIN.

This leads to the following set of constraints (by ignoring the blockage constraints imposed on the base of the target, which anyway involve completely different nodes):

$$\begin{aligned} \boldsymbol{v}_1 \cdot \hat{\boldsymbol{n}}_1 - (N_2\boldsymbol{v}_2 + N_3\boldsymbol{v}_3) \cdot \hat{\boldsymbol{n}}_1 = 0 \\ \boldsymbol{v}_2 \cdot \hat{\boldsymbol{n}}_2 - (M_1\boldsymbol{v}_1 + M_4\boldsymbol{v}_4) \cdot \hat{\boldsymbol{n}}_2 = 0 \end{aligned} \tag{63}$$

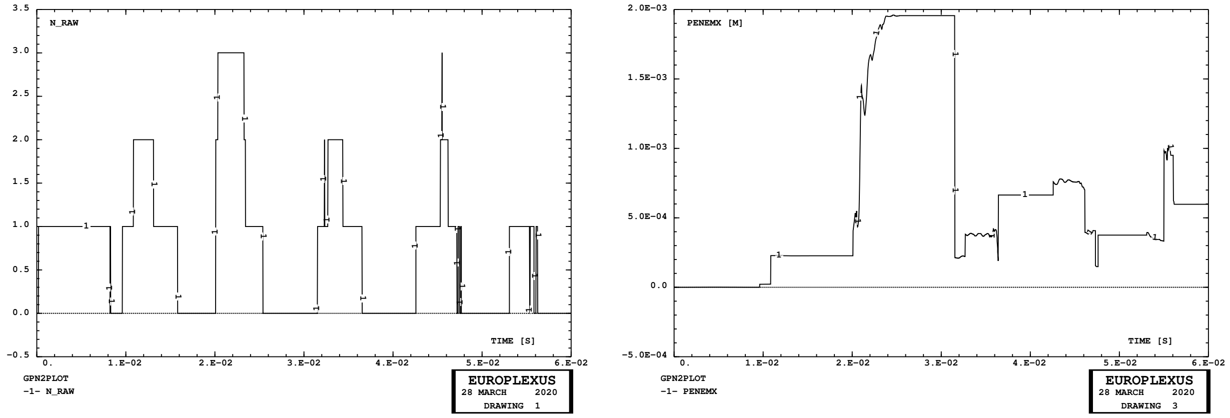
where $\boldsymbol{v}_I$ are the velocities of the nodes involved by the two constraints (P1 to P4), N_K are the shape functions of the position (projection) of P1 on L1 and M_L are the shape functions of the position (projection) of P2 on L2. Note that $\hat{\boldsymbol{n}}_1$ and $\hat{\boldsymbol{n}}_2$ are *not* perfectly co-linear in this case, i.e. $\hat{\boldsymbol{n}}_1 \neq -\hat{\boldsymbol{n}}_2$ or in other words:

$$|\hat{\boldsymbol{n}}_1 \cdot \hat{\boldsymbol{n}}_2| < 1 \tag{64}$$

Note also that the velocities $\boldsymbol{v}_1$ and $\boldsymbol{v}_4$ of the projectile appearing in (63) are much larger (in particular, as concerns their *horizontal* components) than the velocities of the target ($\boldsymbol{v}_2$ and $\boldsymbol{v}_3$).

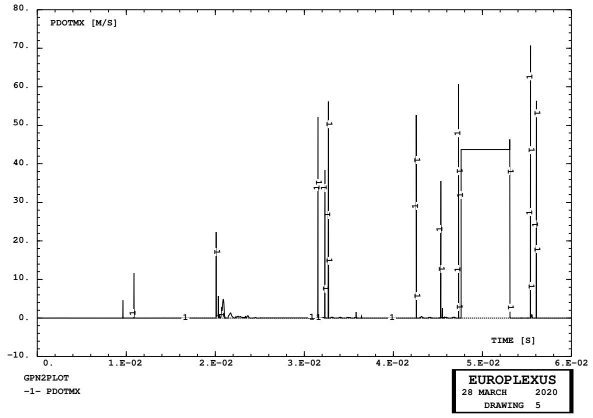
The solution of the system of constraints by Lagrange multipliers produces the following reaction forces (expressed in GN):

$$\begin{aligned} \boldsymbol{R}_1 &= (-1.567, -0.014) \\ \boldsymbol{R}_2 &= (+1.692, -2.878) \\ \boldsymbol{R}_3 &= (+0.002, +0.766) \\ \boldsymbol{R}_4 &= (-0.127, +2.126) \end{aligned} \tag{65}$$



(a) Raw penetrations

(b) Max penetration at this step



(c) Max penetration rate at this step

Figure 14: Statistics results for case FROT82

This result looks consistent, since it is $\sum_I R_{xI} = 0$ and $\sum_I R_{yI} = 0$, as it should be for *internal* reactions such as contact forces. However, it is at first sight very surprising that the horizontal (tangent) components of the forces are so large, of the same order of magnitude as the vertical (normal) components, see also Figures 22(a) and 24. Roughly speaking, with reference to Figure 22(b), it is almost like if the node P1 would hit node P2 “horizontally”. Node P1 is somewhat slowed down while node P2 is accelerated horizontally.

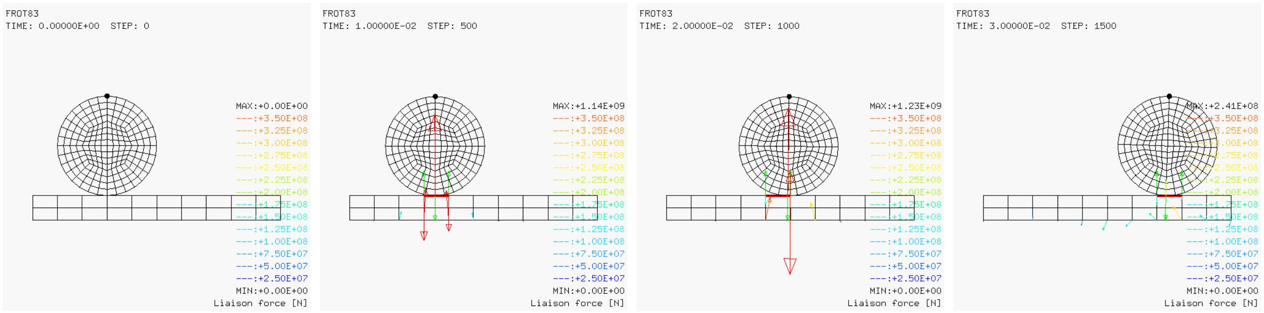
The old full-step velocities at step 1432 (left column) and the new full-step velocities at step 1433 (right column) are, in m/s:

$$\begin{aligned} \boldsymbol{v}_1 &= (281.2, 2.5) & \qquad \boldsymbol{v}_1 &= (239.2, -9.4) \\ \boldsymbol{v}_2 &= (0.5, 3.3) & \qquad \boldsymbol{v}_2 &= (9.3, -9.3) \\ \boldsymbol{v}_3 &= (0.9, 0.4) & \qquad \boldsymbol{v}_3 &= (1.1, 4.2) \\ \boldsymbol{v}_4 &= (283.8, 4.6) & \qquad \boldsymbol{v}_4 &= (280.9, 59.5) \end{aligned} \tag{66}$$

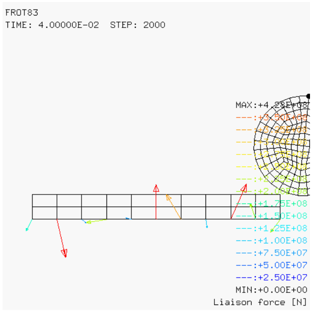
Another quite surprising thing is that the direction of the reaction forces at P1 and P2 deviates so much from the direction of the contact normal(s) $\hat{\boldsymbol{n}}_1$ and $\hat{\boldsymbol{n}}_2$ (which are not exactly but *almost* coincident, apart from the sign), in the absence of friction.

4.4.1 Verifications

The first verification to do is that the constraints (63) are properly written by the code, i.e. that the coefficients are correct (see below). Then, one should verify that the numerical solution of the system



(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 30$ ms



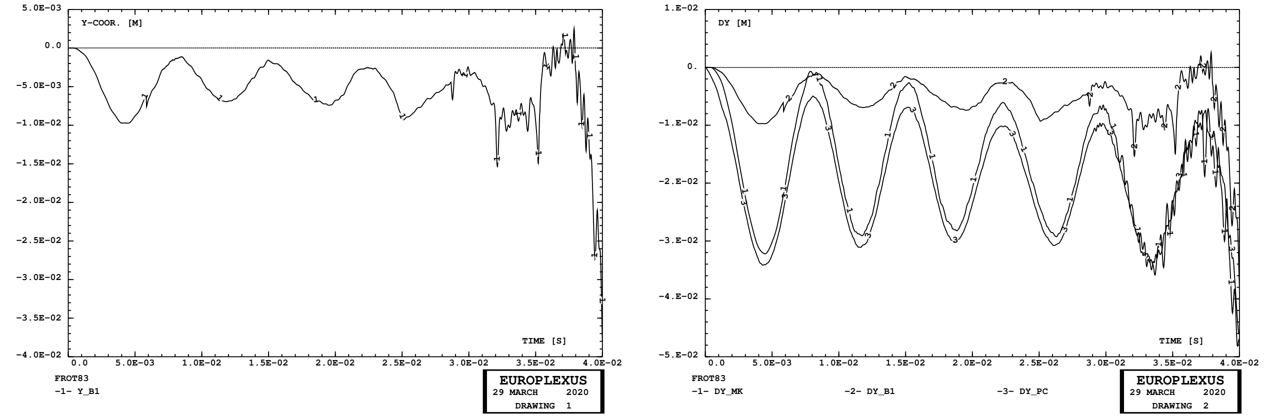
(e) $t = 40$ ms

Figure 15: Result of test FROT83

is correct. To this end, the optional `VERI` keyword is added to the `LINK COUP` directive, which triggers the *a posteriori* verification of the imposed constraints. For the two links under consideration, at step 1433, we obtain:

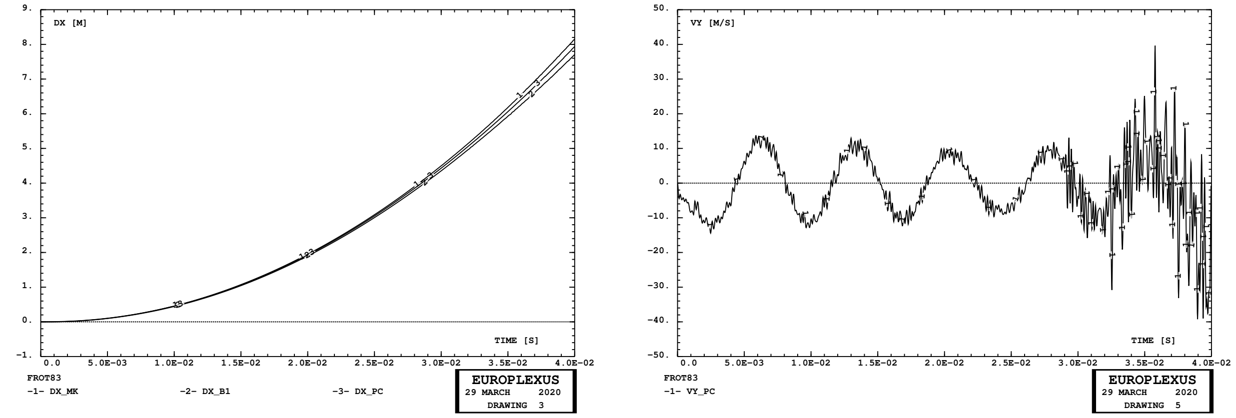
```
***> A POSTERIORI LIAISON VERIFICATION:
 LIAISON OF TYPE: 76 CLASS:  2 NPLIE=  3 NDLIE=  6 BTERM=  0.00E+00 TRUP=  1.00E+12
   PLIE =         25        239        238
   DLIE =         49         50        475        476        477        478
   RANG =          1          2          5          6          3          4
   COEF =  -2.46E-03 -1.00E+00  7.45E-05  3.03E-02  2.39E-03  9.70E-01
   LDATA=         49         -1          0
   RDATA= -2.46061E-03-9.99997E-01 0.00000E+00 0.00000E+00-2.53057E+10 0.00000E+00 0.00000E+00
   READABLE: +0.00000E+00 = -2.46061E-03*V_1,00025 -9.99997E-01*V_2,00025 +2.38611E-03*V_1,00239 +9.69719E-01*V_2,00239
   READAB. :                +7.45033E-05*V_1,00238 +3.02782E-02*V_2,00238
NPAS =  1433 T = 2.86600E-02 DTSUM = 4.00000E-05
  I       K FE(K)       FI(K)       XM(K)       V(K)        ACCE        GAMMAK      VNEW        COEF(I)     LSUM
  1      49-1.56764E+09-5.69490E+06 3.70165E+02 2.81355E+02-4.21960E+06 2.00000E-05 1.96964E+02-2.46061E-03-4.84651E-01
  2      50-1.42788E+07 4.28505E+08 3.70165E+02 2.52620E+00-1.19618E+06 2.00000E-05-2.13974E+01-9.99997E-01 2.09127E+01
  3     475 1.88536E+06-1.73712E+07 1.95000E+03 9.79469E-01 9.87514E+03 2.00000E-05 1.17697E+00 7.45033E-05 2.09128E+01
  4     476 7.66213E+08 9.10789E+06 1.95000E+03 2.78959E-01 3.88259E+05 2.00000E-05 8.04414E+00 3.02782E-02 2.11563E+01
  5     477 1.69232E+09-3.91075E+06 1.95000E+03 5.78995E-01 8.69864E+05 2.00000E-05 1.79763E+01 2.38611E-03 2.11992E+01
  6     478-2.87839E+09-4.20853E+08 1.95000E+03 3.34429E+00-1.26028E+06 2.00000E-05-2.18612E+01 9.69719E-01 3.09086E-13
 ERR =-1.45801E-14

***> A POSTERIORI LIAISON VERIFICATION:
 LIAISON OF TYPE: 76 CLASS:  2 NPLIE=  3 NDLIE=  6 BTERM=  0.00E+00 TRUP=  1.00E+12
   PLIE =        239         25         24
   DLIE =         47         48         49         50        477        478
   RANG =          5          6          3          4          1          2
   COEF =   4.61E-03 -7.74E-02  5.48E-02 -9.21E-01 -5.94E-02  9.98E-01
   LDATA=        477         -1          0
   RDATA= -5.94160E-02 9.98233E-01 0.00000E+00 0.00000E+00 2.74664E+10 0.00000E+00 0.00000E+00
   READABLE: +0.00000E+00 = -5.94160E-02*V_1,00239 +9.98233E-01*V_2,00239 +5.48079E-02*V_1,00025 -9.20813E-01*V_2,00025
   READAB. :                +4.60816E-03*V_1,00024 -7.74204E-02*V_2,00024
NPAS =  1433 T = 2.86600E-02 DTSUM = 4.00000E-05
  I       K FE(K)       FI(K)       XM(K)       V(K)        ACCE        GAMMAK      VNEW        COEF(I)     LSUM
  1      47-1.26569E+08-5.34670E+06 3.82231E+02 2.84024E+02-3.17145E+05 2.00000E-05 2.77681E+02 4.60816E-03 1.27960E+00
  2      48 2.12646E+09 1.38973E+07 3.82231E+02 4.25138E+00 5.52692E+06 2.00000E-05 1.14790E+02-7.74204E-02-7.60748E+00
  3      49-1.56764E+09-5.69490E+06 3.70165E+02 2.81355E+02-4.21960E+06 2.00000E-05 1.96964E+02 5.48079E-02 3.18767E+00
  4      50-1.42788E+07 4.28505E+08 3.70165E+02 2.52620E+00-1.19618E+06 2.00000E-05-2.13974E+01-9.20813E-01 2.28907E+01
  5     477 1.69232E+09-3.91075E+06 1.95000E+03 5.78995E-01 8.69864E+05 2.00000E-05 1.79763E+01-5.94160E-02 2.18226E+01
  6     478-2.87839E+09-4.20853E+08 1.95000E+03 3.34429E+00-1.26028E+06 2.00000E-05-2.18612E+01 9.98233E-01 5.86198E-13
 ERR =-2.56086E-14
```

Since the errors are numerically zero, the conditions are verified. Note that the `VERI` keyword produces also the dump on the listing of the contents of the links, as listed above. We also add the `LNKS DUMP` option in order to produce even more detailed output, including a human-readable (natural) form of the constraint marked in red in the above text. This allows us to check also that the links were appropriately *written* in the first place, besides being correctly *solved*.



(a) Vertical coordinate

(b) Vertical displacements



(c) Horizontal displacements

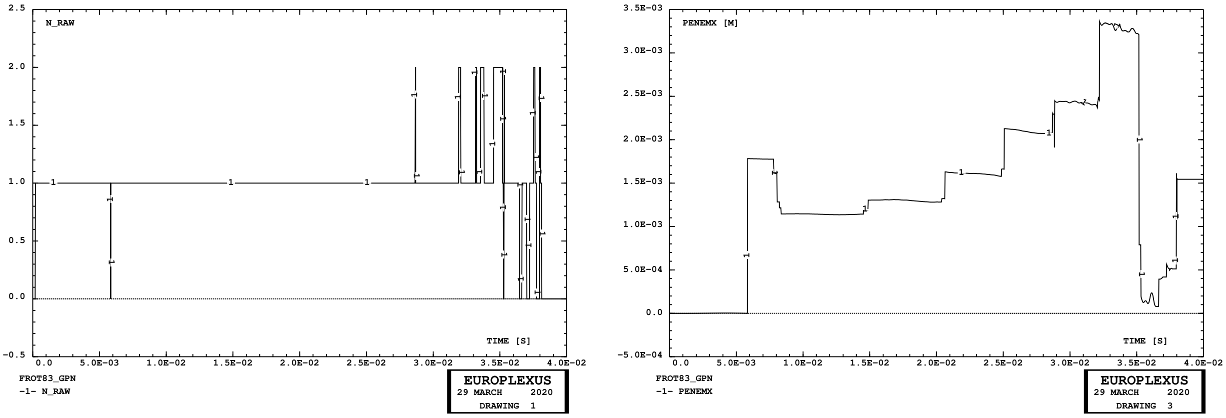
(d) Vertical velocity

Figure 16: Further results of case FROT83

For the first of the two `GPIN` links, we find (these are some of the fields of a derived type `LIAISON`):

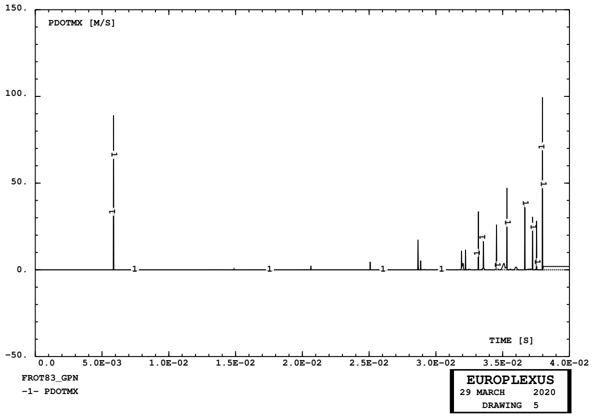
$$
\begin{aligned}
\texttt{BTERM} &= 0.0 \\
\texttt{PLIE} &= 25, 239, 238 \\
\texttt{DLIE} &= 49, 50, 475, 476, 477, 478 \quad \text{(In growing order)} \\
\texttt{RANG} &= 1, 2, 5, 6, 3, 4 \\
\texttt{COEF} &= -0.002, -1.000, 0.000, 0.030, 0.002, 0.970 \quad \text{(Same order as \texttt{DLIE})}
\end{aligned}
$$

The `BTERM` is the right-hand side of the constraint. The `PLIE` are the indexes of the constrained nodes, in the order they appear in the constraint written in *natural form* (i.e., *not* ordered by growing value). In this case, the first value is the penetrating (or slave) entity (node of the P-GPIN) and the other two are the penetrated (or master) entities (nodes of the L-GPIN). The `DLIE` are the linked degrees of freedom (dofs), listed *in growing order*. These are all the dofs of the `PLIE`, since the constraint includes also any dofs having a zero coefficient. The `RANG` array allows to pass from the ordered numbering of `DLIE` to the unordered numbering of `PLIE`. More precisely, `RANG(I)` is the rank that the `I`-th dof listed in `DLIE` had in the list of dofs before this was renumbered in growing order, i.e. in the sequence of dofs listed according to the `PLIE` list. In this example, the unordered `DLIE` would be: $49, 50, 477, 478, 475, 476$, that is, first the 2 dofs of node 25, then those of node 239 and finally those of node 238 (i.e., nodes as appearing in the `PLIE` list). Finally, the `COEF` are the coefficients of the link, listed in the same order as `DLIE`, i.e. by growing dof index.



(a) Raw penetrations

(b) Max penetration at this step



(c) Max penetration rate at this step

Figure 17: Statistics results for case FROT83

From the listing, we find for this constraint the following additional information:

$$\begin{aligned}
\hat{\boldsymbol{n}} &= (-0.002, -1.000) && \text{(Contact unit normal)} \\
\xi &= -0.939 && \text{(Normalized coordinate of contact point)} \\
\boldsymbol{N} &= (0.970, 0.030) && \text{(Weighting coefficients)}
\end{aligned}$$

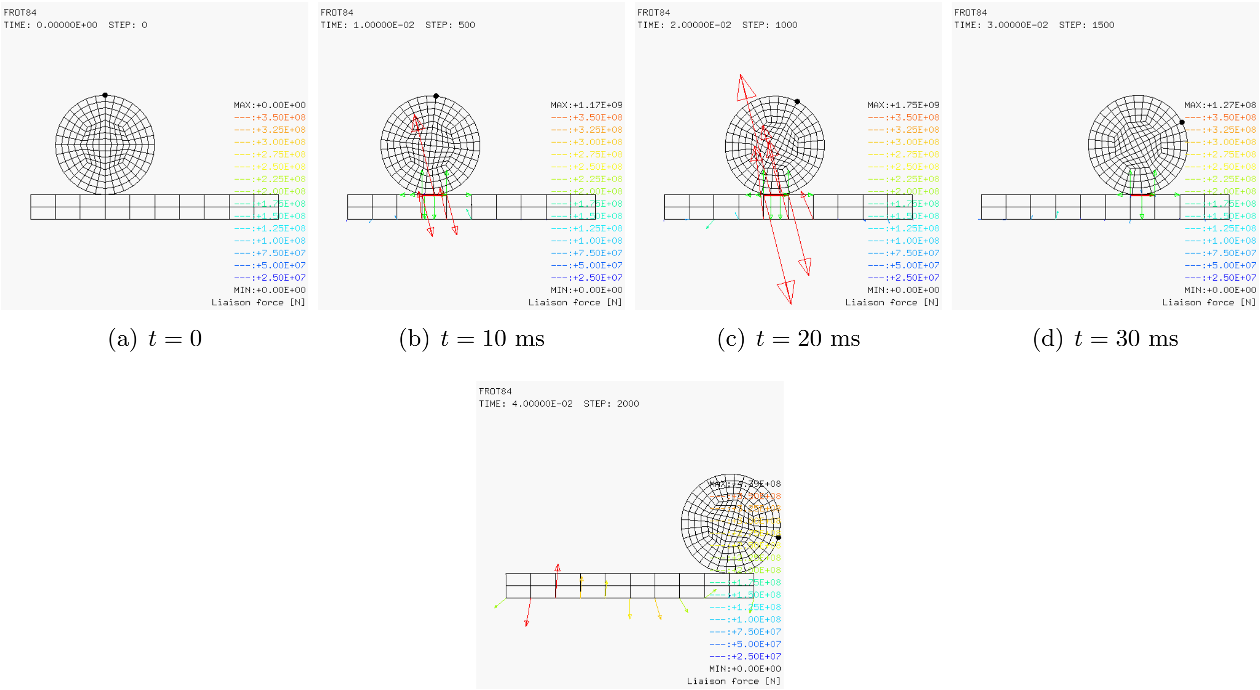
where ξ is the normalized coordinate of the contact point (projection of the P-GPIN onto the L-GPIN) with respect to the L-GPIN ($-1 \leq \xi \leq 1$) and $\boldsymbol{N}$ are the corresponding (line) shape functions: $N_1 = (1-\xi)/2$, $N_2 = (1+\xi)/2$. The constraint can be formally written (in natural form):

$$\boldsymbol{v}_{25} \cdot \hat{\boldsymbol{n}} - (N_1 \boldsymbol{v}_{239} + N_2 \boldsymbol{v}_{238}) \cdot \hat{\boldsymbol{n}} = 0 \tag{67}$$

By expanding this equation we obtain:

$$\begin{aligned}
0 &= -0.002 v_{x,25} - 1.000 v_{y,25} \\
&\quad + 0.970 \times 0.002 v_{x,239} + 0.970 \times 1.000 v_{y,239} \\
&\quad + 0.030 \times 0.002 v_{x,238} + 0.030 \times 1.000 v_{y,238} \\
&= -0.002 v_{x,25} - 1.000 v_{y,25} + 0.002 v_{x,239} + 0.970 v_{y,239} + 0.000 v_{x,238} + 0.030 v_{y,238}
\end{aligned} \tag{68}$$

The resulting coefficients from the last line of the above equation correspond to the `COEF` listed above, if one takes into account the `RANG`. Therefore, this shows that the first constraint is correctly written.



(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 30$ ms

(e) $t = 40$ ms

Figure 18: Result of test FROT84

Similarly, for the second `GPIN` constraint we find from `VERI`:

$$
\begin{aligned}
\texttt{BTERM} &= 0.0 \\
\texttt{PLIE} &= 239, 25, 24 \\
\texttt{DLIE} &= 47, 48, 49, 50, 477, 478 \quad \text{(In growing order)} \\
\texttt{RANG} &= 5, 6, 3, 4, 1, 2 \\
\texttt{COEF} &= 0.005, -0.078, 0.055, -0.921, -0.059, 0.998 \quad \text{(Same order as \texttt{DLIE})}
\end{aligned}
$$

and from the dump option:

$$
\begin{aligned}
\hat{\boldsymbol{n}} &= (-0.059, 0.998) && \text{(Contact unit normal)} \\
\xi &= -0.845 && \text{(Normalized coordinate of contact point)} \\
\boldsymbol{N} &= (0.922, 0.078) && \text{(Weighting coefficients)}
\end{aligned}
$$

The constraint can be formally written (in natural form):

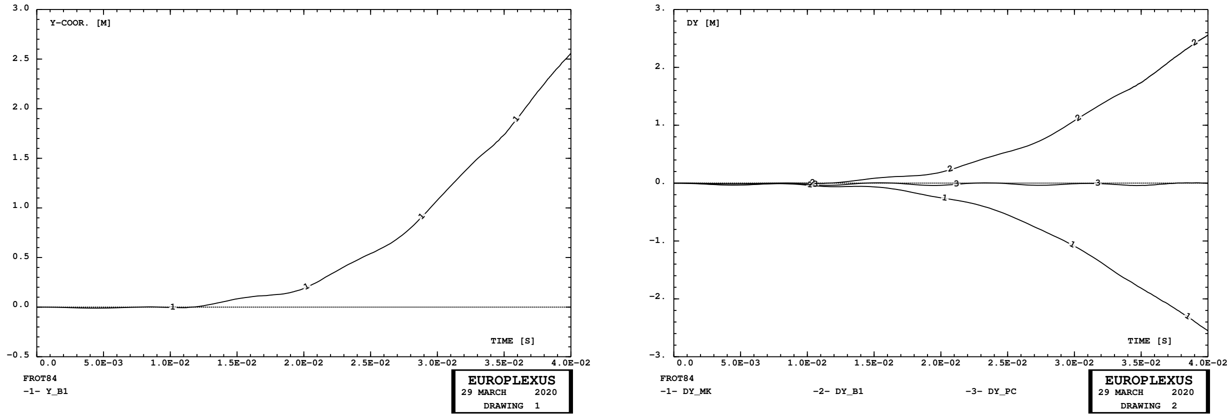
$$
\boldsymbol{v}_{239} \cdot \hat{\boldsymbol{n}} - (N_1 \boldsymbol{v}_{25} + N_2 \boldsymbol{v}_{24}) \cdot \hat{\boldsymbol{n}} = 0 \tag{69}
$$

By expanding this equation we obtain:

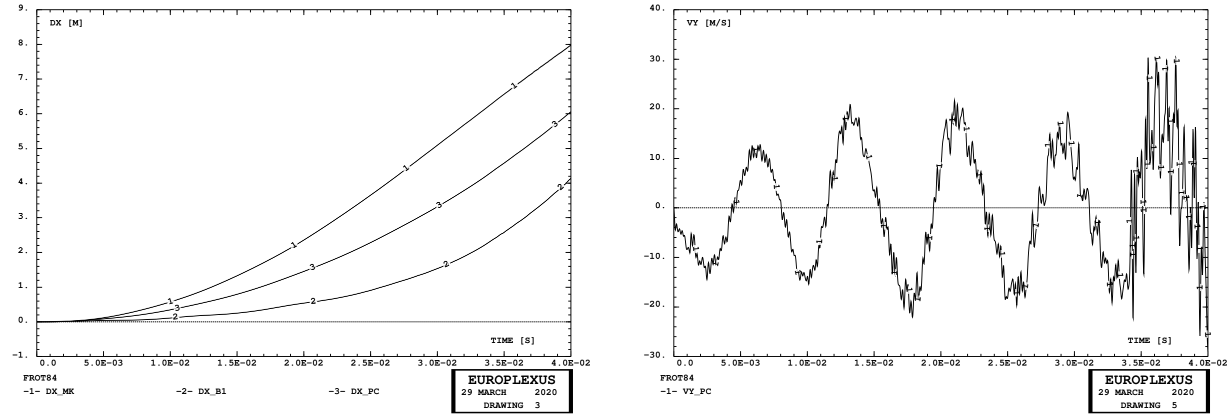
$$
\begin{aligned}
0 &= -0.059 v_{x,239} + 0.998 v_{y,239} \\
&\quad + 0.922 \times 0.059 v_{x,25} - 0.922 \times 0.998 v_{y,25} \\
&\quad + 0.078 \times 0.059 v_{x,24} - 0.078 \times 0.998 v_{y,24} \\
&= -0.059 v_{x,239} + 0.998 v_{y,239} + 0.055 v_{x,25} - 0.921 v_{y,25} + 0.005 v_{x,24} - 0.078 v_{y,24}
\end{aligned} \tag{70}
$$

Also in this case the resulting coefficients coincide with those in the `COEF` vector, by taking due account of `RANG`. So, also the second constraint has been correctly written.

Since any implementation or numerical errors seem to be excluded by the above verifications, we must conclude that the observed interlocking phenomenon is a consequence of some redundancy in the constraints (over-constraining) imposed on the problem through the Lagrange multipliers approach.



(a) Vertical coordinate (b) Vertical displacements



(c) Horizontal displacements (d) Vertical velocity

Figure 19: Further results of case FROT84

Case	Mesh	Description	Final time [ms]	Steps	CPU [s]
FROT87	4 CAR4	`GPIN`	0	0	—
FROT88	4 CAR4	Add `MASL`	0	0	—
FROT89	4 CAR4	`GLIS`, one line	0	0	—
FROT90	4 CAR4	`GLIS`, two lines	0	0	—

Table 3: Calculations to check over-constraining on a simplified problem in 2D.

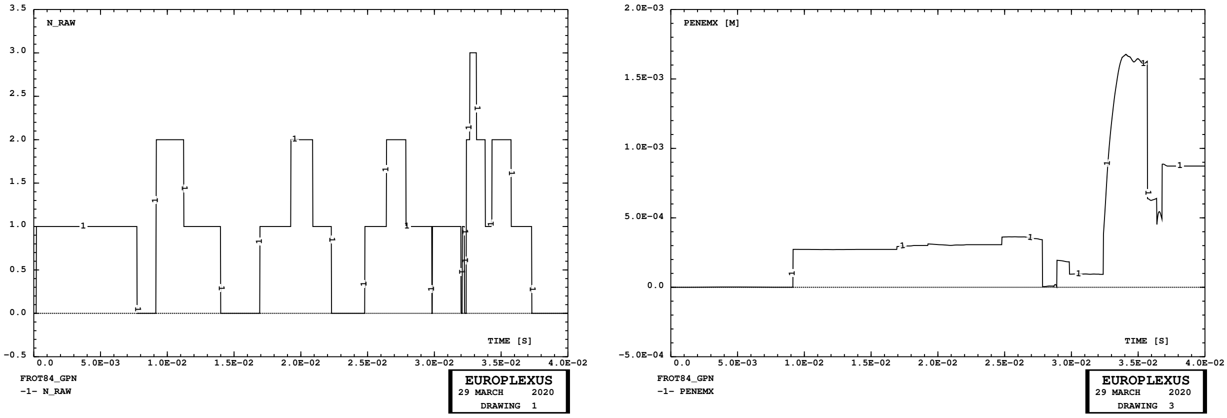
As a further confirmation of the above verifications, we carry out simulations with both `GPIN` and `GLIS` of a much simplified problem, which closely mimics the faulty configuration observed in case FROT83 at step 1433. The tests performed are summarized in Table 3.

The test geometry is shown in Figure 25. A uniform horizontal velocity of 100 m/s and a uniform vertical velocity of -1 m/s are assigned to all nodes of the projectile, while the target is initially at rest. Only one step (step 0) is performed, then the resulting link forces are inspected.

4.4.2 Case FROT87

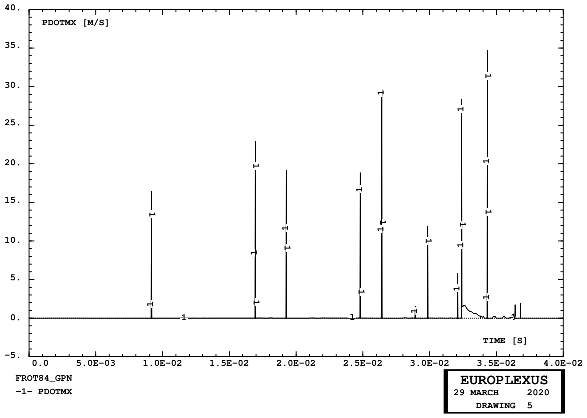
This test uses `GPIN` to treat the contact in the simplified problem. Some results are presented in Figure 26. Note in particular from Figure 26(b) that the contact forces at the two penetrating nodes have very large horizontal components, like in case FROT83. This is not justified by physical intuition, since there is no friction in this problem.

Another strange aspect is that both reactions at the two penetrating nodes have *negative* vertical components. Again, this is contrary to physical intuition.



(a) Raw penetrations

(b) Max penetration at this step



(c) Max penetration rate at this step

Figure 20: Statistics results for case FROT84

4.4.3 Case FROT88

This test is similar to FROT87 but we add `MASL PAIR 2 1` in order to activate a master/slave technique whereby the second body (here the target) acts as master while the first body (here the projectile) acts as slave.

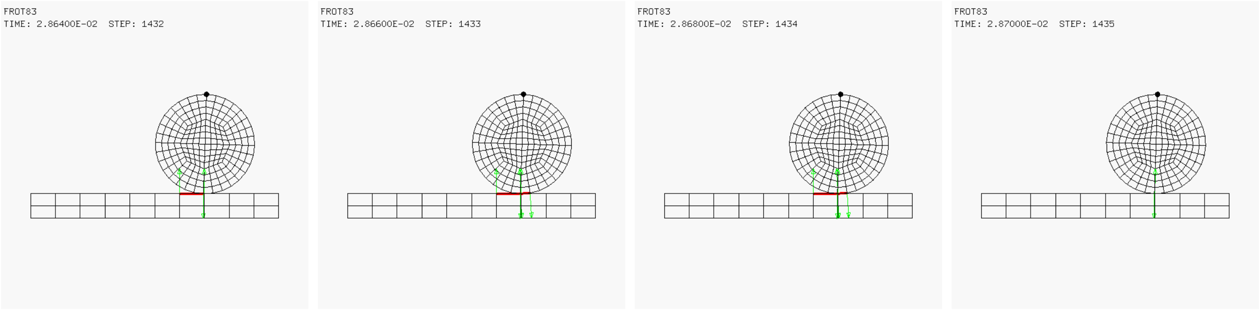
Some results are presented in Figure 27. In this case only one constraint is written instead of two, and the reaction forces are perfectly vertical.

4.4.4 Case FROT89

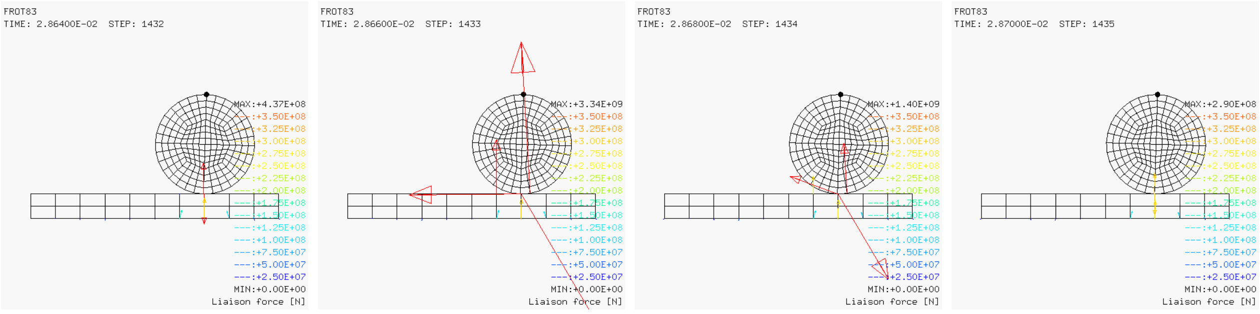
This test is similar to FROT87 but we want to use `GLIS` instead of `GPIN` as a contact model. Recall that the `GLIS` model in 2D (sliding lines) is not as developed as the 3D version (sliding surfaces). In particular, it cannot be used with the `LINK COUP` directive, and requires the old `LIAI` links formulation. Furthermore, some dimensioning is needed.

In an attempt to reproduce the redundancy observed with `GPIN` in test FROT87, we prescribe *two* sliding lines, by swapping the roles of master and slave:

```
DIME GLIS 2 200 TERM
 . . .
LIAI GLIS 2
          MAIT LECT 6 5 4 TERM
          ESCL LECT 7 8 9 TERM
          MAIT LECT 7 8 9 TERM
          ESCL LECT 6 5 4 TERM
```

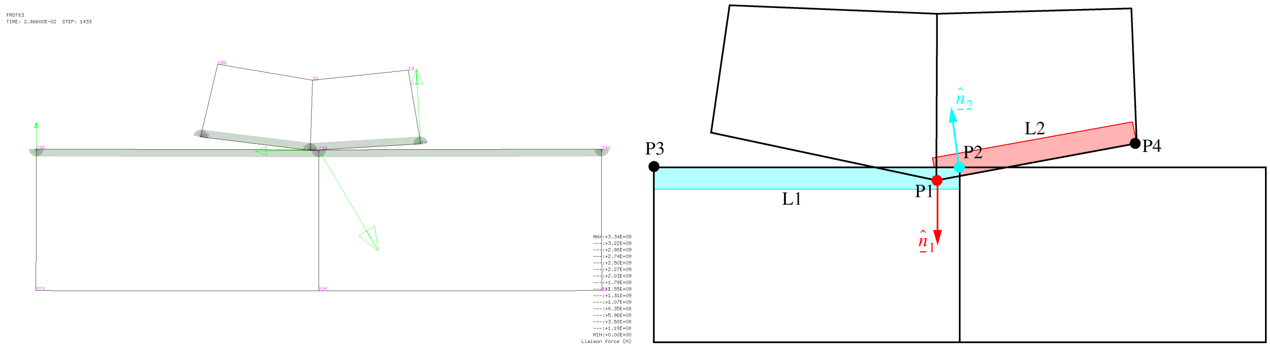


(a) Step 1432 (b) Step 1433 (c) Step 1434 (d) Step 1435



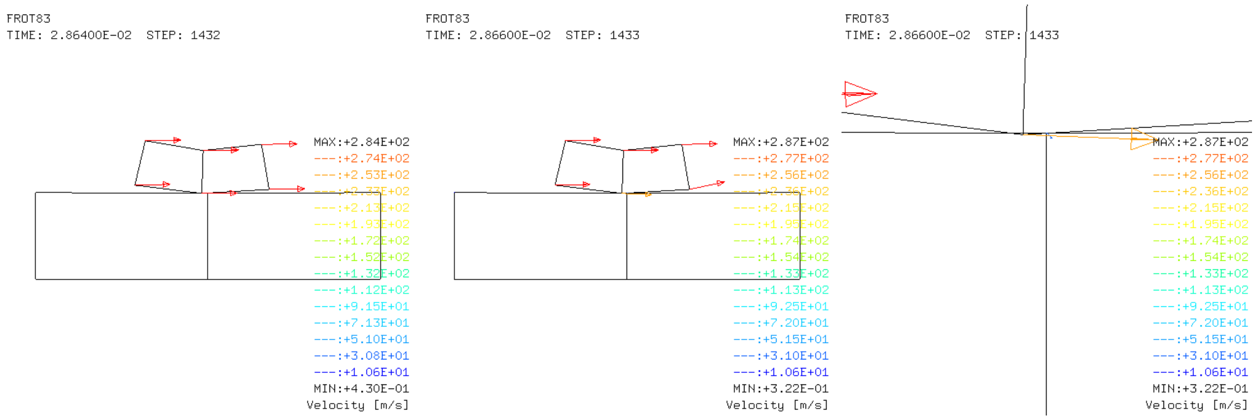
(e) Step 1432 (f) Step 1433 (g) Step 1434 (h) Step 1435

Figure 21: Links of type `GPIN` (top) and reaction forces (bottom) in test FROT83



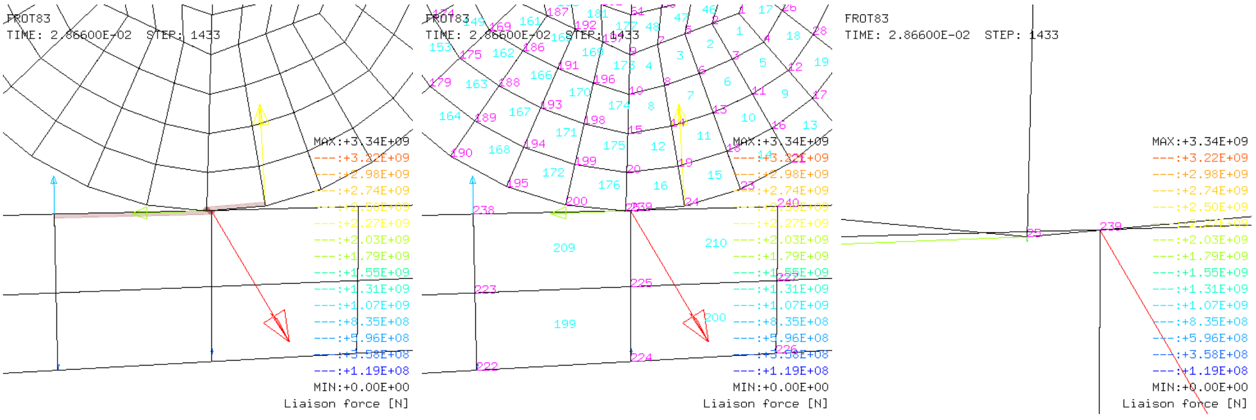
(a) Real situation (b) Schematic representation

Figure 22: Interlocking at step 1433 in test FROT83



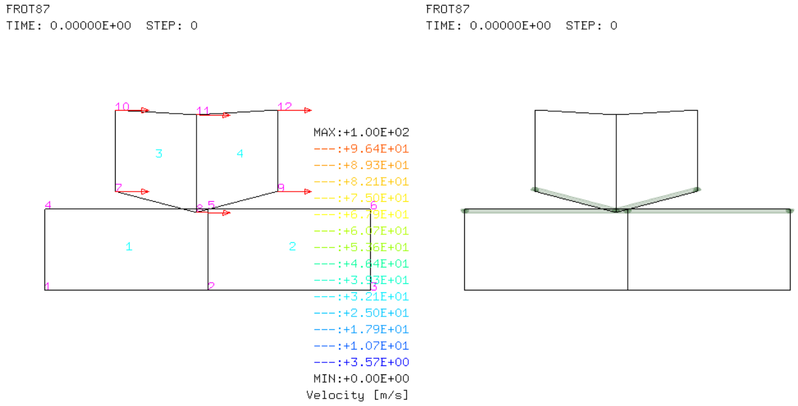
(a) Step 1432 (b) Step 1433 (c) Step 1433 (detail)

Figure 23: Detail of velocities in test FROT83



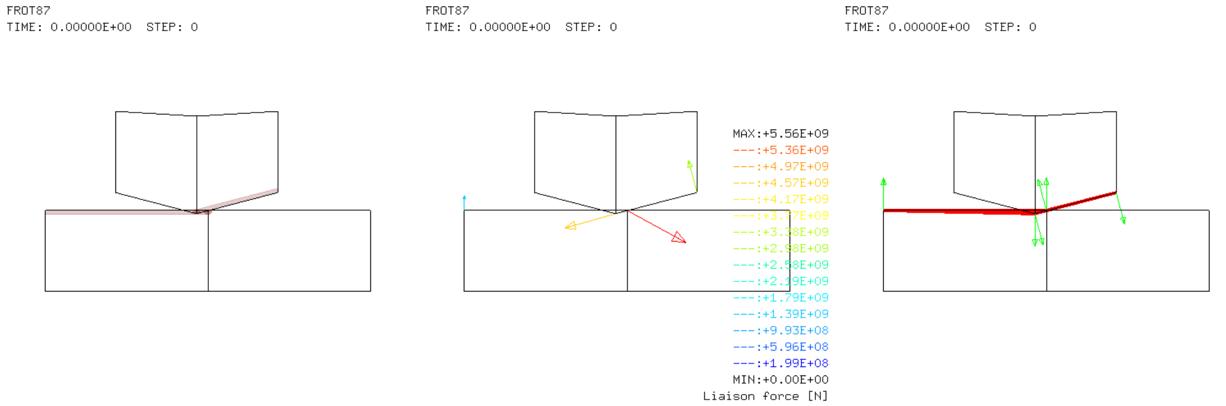
(a) Step 1433 (b) With numbering (c) Step 1433 (detail)

Figure 24: Detail of `GPIN` link forces in test FROT83



(a) Geometry and numbering (b) GPINs

Figure 25: Geometry of simplified test for over-constraining verification

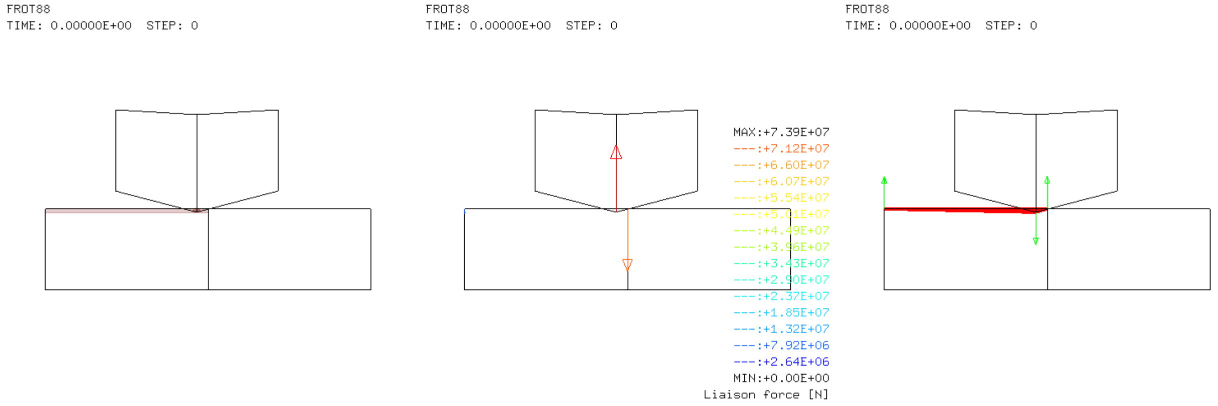


(a) Contacting GPINs (b) Contact forces (c) Contact link joints

Figure 26: Results of test FROT87

Note that the node indexes defining a sliding line must be ordered from one extremity to the other one of the line (so they are not necessarily in growing order), and in such a way that the body is located *to the left* of the line for an observer moving along the line itself.

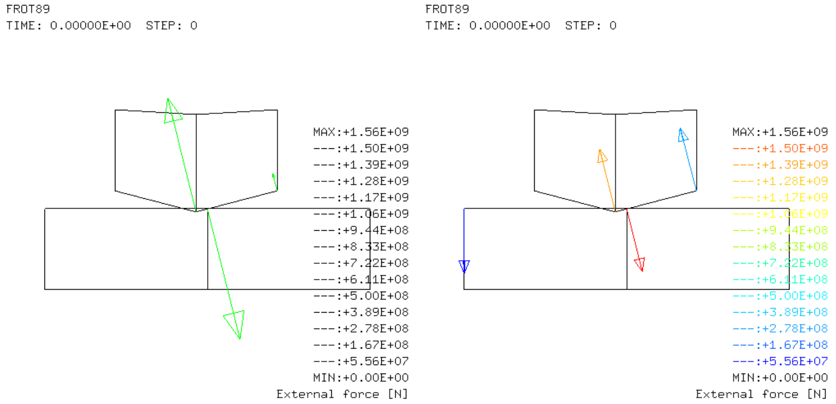
Some results are presented in Figure 28. Neither the contacts nor the link joints can be visualized with the old `LIAI` model. Furthermore, the link forces are not filled up ans are all null. However, the contact forces are added up to external forces for `LIAI`. Since there are no explicitly applied external



(a) Contacting GPINs (b) Contact forces (c) Contact link joints

Figure 27: Results of test FROT88

loads, by visualizing the external forces `FEXT` what we see are in fact the contact forces.



(a) Contact forces (b) Contact force directions

Figure 28: Results of test FROT89

In Figure 28(a) the contact forces are presented as green arrows of length proportional to the force intensity. As it can be seen, the forces in the two penetrating nodes are much larger than in the other nodes and have large horizontal components, like in the solution with `GPIN`. This seems to confirm that over-constraining can lead to weird solutions with the method of Lagrange multipliers.

The Figure 28(b) shows again the contact forces, now by using vectors all of the same length but of color proportional to the intensity. This allows to observe the direction of contact forces at the two extreme nodes, which are much smaller than those on the penetrating nodes and therefore are not visible in the first picture.

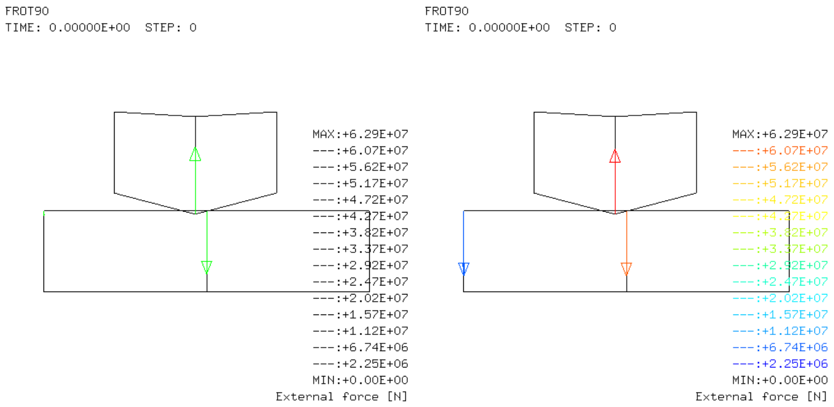
4.4.5 Case FROT90

This test is similar to FROT89 (`GLIS`) but it uses only one (the first one) sliding line (where the plane target is the master) in order to avoid redundancies, for comparison with case FROT88 which used `GPIN` and `MASL`.

The resulting contact forces are presented in Figure 29. There is only one contact and the reactions are vertical, unlike in the previous solution.

4.5 Strategies to eliminate interlocking

Some possible strategies to eliminate or mitigate over-constraining could be as follows:



(a) Contact forces (b) Contact force directions

Figure 29: Results of test FROT90

A Master/slave technique (like with `GLIS`). Asking the user to designate a master and a slave would reduce the number of constraints. In the above example, if 1 is treated as slave and 2 as master, then only the first constraint (63) would be written, thus avoiding the interlocking. This method is relatively easy to implement. However, there are also some drawbacks. First, the symmetry of the contact problem is broken and the solution will depend upon the master/slave choice. Second, there are more spurious penetrations (of master nodes into the slave surface). Finally, it is not evident how to treat the case of self-contact.

B Retain both (all) constraints but use an *average* normal $\hat{\boldsymbol{n}}^*$ (the same for all constraints):

$$\hat{\boldsymbol{n}}^* = \frac{\hat{\boldsymbol{n}}_1 - \hat{\boldsymbol{n}}_2}{||\hat{\boldsymbol{n}}_1 - \hat{\boldsymbol{n}}_2||} \tag{71}$$

One should verify by a practical example whether this technique works or not. Potential drawbacks of this method are firstly that the detection of potentially interlocking constraints is somewhat difficult and computationally expensive, and secondly that there might be more than two related links (any number is possible, in principle).

C Recognize that the (two or more) related constraints are potentially leading to interlocking and replace them by a *single* constraint. In the above example, in place of (63) one would write:

$$\left(\boldsymbol{v}_1' - \boldsymbol{v}_2'\right) \cdot \hat{\boldsymbol{n}}^* = 0 \tag{72}$$

where $\boldsymbol{v}_1'$ and $\boldsymbol{v}_2'$ are the velocities of the projected points P1 and P2 along the lines L1 and L2 and $\hat{\boldsymbol{n}}^*$ is the average normal defined above in case B. With the symbols defined previously, the expression would become:

$$(N_2\boldsymbol{v}_2 + N_3\boldsymbol{v}_3) \cdot \hat{\boldsymbol{n}}^* - (M_1\boldsymbol{v}_1 + M_4\boldsymbol{v}_4) \cdot \hat{\boldsymbol{n}}^* = 0 \tag{73}$$

A drawback of this strategy (as well as of others) is the cost and difficulty of the detection of potentially interlocking constraints.

D Recognize that the (two or more) related constraints are potentially leading to interlocking and retain only one of them (e.g. arbitrarily, the *first* one). This approach has at least two drawbacks: first the cost and difficulty of the detection, as for other methods, and second the fact that the solution will depend upon the particular numbering of nodes and elements chosen in the model.

Strategies B and C seem too complicated (and also too expensive) to implement. They are reminiscent of similar strategies for constraint redundancy elimination that were first tentatively introduced in the classical pinball contact method (`PINB` model), without much success. See reference [15] for details.

An attempt will therefore be made in the following to implement and test strategies A and D.

4.5.1 Detecting potentially interlocking constraints

Most strategies outlined above require the detection of potentially interlocking constraints. A (qualitative) algorithm to achieve this detection could be as follows. A penetration P2-L2 is potentially leading to interlocking if there exists another penetration P1-L1 such that P2 belongs to L1 (i.e., P2 is one of the nodes of L1) and P1 belongs to L2 (i.e., P1 is one of the nodes of L2).

4.5.2 Implementation of a master/slave strategy

An optional master/slave strategy is tentatively implemented in the `GPIN` model. This is activated by the following sub-directive (similar to the `EXCL` sub-directive for the exclusion of body pairs):

```
MASL ( PAIR m s )
```

where `m` (for master) and `s` (for slave) are two different body indexes, each ranging from 1 to the number B of either regular or self-contacting bodies that have been previously declared. As indicated by the parentheses, the `PAIR` sub-keyword can be repeated to define as many master/slave couples as needed. By default, body couples not mentioned in the `MASL` sub-directive act both as a master and as a slave when they come into contact with each other.

A square matrix of integers `GPIN_MASL(:,:)`, where each dimension ranges from 1 to B, is added and initialized to 0. Then the matrix is filled in in such a way that, when checking the contact between a body I and a body J (with $I \neq J$):

- `GPIN_MASL(I,J) = +1` if body I acts as the master and J as the slave.
- `GPIN_MASL(I,J) = -1` if body I acts as the slave and J as the master.
- `GPIN_MASL(I,J) = 0` if both bodies I and J act in turn as master and slave (default behaviour).

Entries of the matrix on the main diagonal (i.e. for $I = J$) are left at 0 for the moment (they might be exploited later to deal with self-contact). So, the matrix `GPIN_MASL` is antisymmetric.

As concerns the generation of GPINs, two strategies appear to be possible. Either build all GPINs as usual, like in the case without master/slave definitions, and then use only the appropriate ones. Or, build GPINs as usual for any body acting as a master with respect to at least another body (slave), but build only a reduced set of GPINs for any body that always acts as a slave with respect to any other body. In 2D, the reduced set means building only P-GPINs and no L-GPINs. In 3D the meaning of reduced set is more complicated and will have to be further explored.

In the 2D case, the only penetration checks are of the types P-P and P-L. Before checking the penetration of a GPIN A into a GPIN B, belonging to body I and J, respectively, we check `GPIN_MASL(I,J)`:

- If `GPIN_MASL(I,J)=0` the two bodies act both as master and as slave with respect to each other, so the check goes on as usual.
- Else (in 2D) A is necessarily a P-GPIN while B can be either a P-GPIN, in which case the check goes on as usual, or B is an L-GPIN, in which case penetration can occur only if body A acts as slave, i.e. when `GPIN_MASL(I,J)=-1`.

In the 3D case, the penetration checks are of types P-P, P-O (where O stays for *other*, i.e. L, T or Q) and L-L. Before checking the penetration of a GPIN A into a GPIN B, belonging to body I and J, respectively, we check `GPIN_MASL(I,J)`:

- If `GPIN_MASL(I,J)=0` the two bodies act both as master and as slave with respect to each other, so the check goes on as usual.
- Else if both A and B are P-GPINs, or both A and B are L-GPINs, the check goes on as usual.
- Else A is a P-GPIN and B is an O-GPIN (L, T or Q), in which case penetration can occur only if body A acts as slave, i.e. when `GPIN_MASL(I,J)=-1`.

4.5.3 Implementation of a constraint redundancy elimination strategy

An attempt is made to implement strategy D of Section 4.5. The algorithm qualitatively described in Section 4.5.1 is used to detect potential over-conditioning. We tentatively include this strategy in the preliminary checks for penetration which, for the 2D case, are described in Section 2.8.1 of the GPIN report [22] (which uses an array C_I).

Assume we have detected a penetration P1-L1 with L1=(P2,P3). The notation in parentheses indicates that L1 has nodes (more precisely, nodal P-GPINs) P2 and P3. Consider another possible penetration PA-LA(PB,PC). We have to skip this penetration check if PA belongs to L1 (i.e. if PA is either P2 or P3) and P1 belongs to LA (i.e. if P1 is either PB or PC).

To this end we introduce a matrix B(N,N) where N is the total number of P-GPINs. This is analogous to the A(:,:) matrix introduced in Section 2.8.2 of [22] for the preliminary checks of L-L penetrations in 3D.

At the beginning of each time step, we set B(:,:)=0. When a P-L penetration PI-LJ(PK,PL) is detected, we set B(I,K)=1, B(I,L)=1, B(K,I)=1 and B(L,I)=1. So the B matrix is symmetric. Then, a penetration check PI-LJ(PK,PL) should be skipped (because potentially leading to interlocking) if B(I,K)==1 or B(I,L)==1.

Note that this strategy is potentially less drastic than the master/slave technique considered in the previous paragraph. In principle, it tries to avoid redundant constraints while leaving both bodies free to act as a master and as a slave at the same time. However, only testing can say if this is effective or not. It is first tentatively implemented only in 2D and it is activated via an option:

```
OPTI GPNS ... RCEL
```

4.6 Solutions with EPX using `GPIN` in master/slave mode

In order to test the master/slave strategy described in a previous paragraph, we perform some simulations, as listed in Table 4.

Case	Mesh	Description	Final time [ms]	Steps	CPU [s]
FROT85	222 CAR4 1 PMAT	No friction, `MASL`	40	2000	15.3
FROT86	222 CAR4 1 PMAT	With friction, `MASL`	40	2000	14.9

Table 4: Calculations for the sliding disk problem in 2D using `GPIN` in master/slave mode.

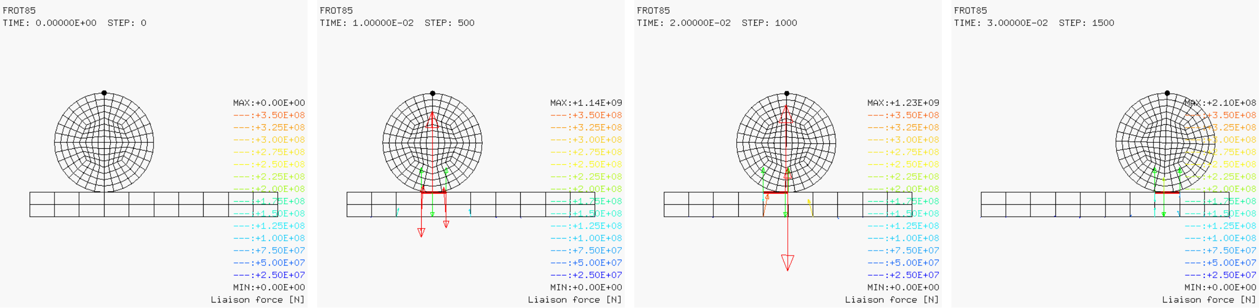
4.6.1 Case FROT85

This test is a repetition of case FROT83 (2D, no friction) but using master/slave mode. We want the target (`base`) to act as a master and the projectile (`circ`) to act as a slave, like in the solutions previously obtained with `GLIS`. Since the target (master) is the second body declared and the projectile (slave) is the first body declared, the constraints directive becomes:

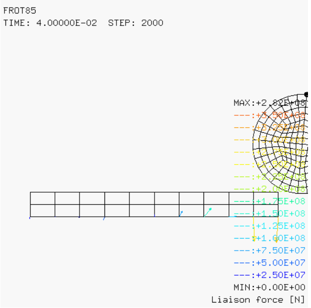
```
LINK COUP
     veri
     BLOQ 12 LECT bas TERM
     GPIN BODY LECT circ TERM
          BODY LECT base TERM
          DIAM 0.05 LECT cir basup TERM
          MASL PAIR 2 1
```

Some results of this simulation are presented in Figures 30 and 31. Figure 32 shows the statistics. The maximum penetration reached is $p_{\max} = 6.016$ mm while the maximum penetration rate is $\dot{p} = 89.12$ m/s.

The solution is relatively smooth and looks correct. The spurious oscillations observed in the final part of case FROT83 are no longer present. Over the entire solution, there is at most one penetration. The (maximum) penetration increases gradually and becomes larger (almost double) than in the solution with bilateral contact (FROT83). This might be normal since here we are imposing less constraints.



(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 30$ ms



(e) $t = 40$ ms

Figure 30: Result of test FROT85

4.6.2 Case FROT86

This test is a repetition of case FROT84 (2D, no friction) but using master/slave mode like in case FROT85.

Some results of this simulation are presented in Figures 33 and 34. Figure 35 shows the statistics. The maximum penetration reached is $p_{\text{max}} = 1.112$ mm while the maximum penetration rate is $\dot{p} = 55.60$ m/s.

Also in this case, the solution is smoother than in the corresponding case with bilateral contact (FROT84), and it looks correct overall.

4.7 Solutions with EPX using `GPIN` and redundant constraints elimination

In order to test the redundant constraints elimination strategy described in a previous paragraph, we perform some simulations, as listed in Table 5.

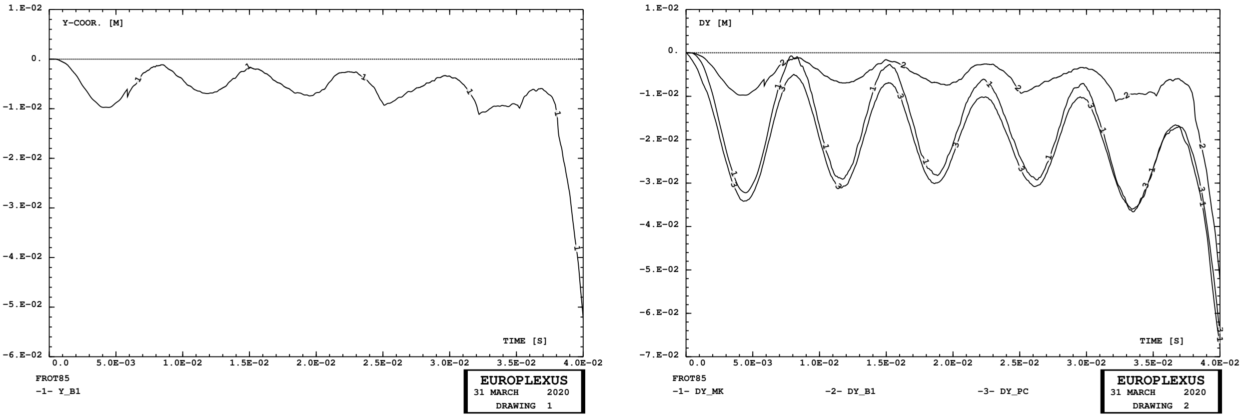
Case	Mesh	Description	Final time [ms]	Steps	CPU [s]
FROT98	222 CAR4 1 PMAT	No friction, `RCEL`	40	2000	14.6
FROT99	222 CAR4 1 PMAT	With friction, `RCEL`	40	2000	14.0

Table 5: Calculations for the 2D sliding disk problem in with `GPIN` and redundant constraints elimination.

4.7.1 Case FROT98

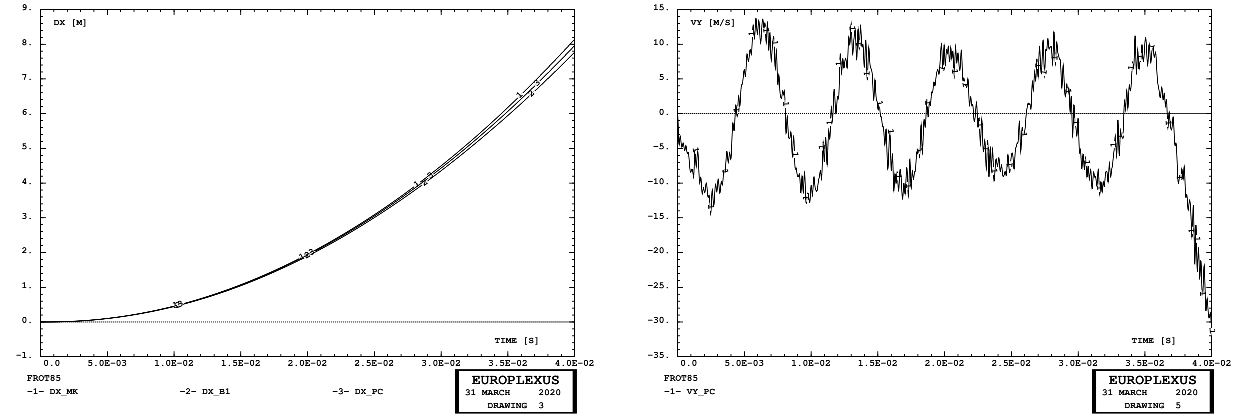
This test is a repetition of case FROT85 (2D, no friction) by using `OPTI GPNS RCEL` to activate redundant constraints elimination during the search for P-L penetrations as described in a previous paragraph.

Remarkably, the result is identical (as far as time curves are concerned) to solution FROT85 (these are not shown for brevity). By comparing the `.gpn` files, we see that in this solution we have one



(a) Vertical coordinate

(b) Vertical displacements



(c) Horizontal displacements

(d) Vertical velocity

Figure 31: Further results of case FROT85

penetration at steps 1902 to 1909 (physical time between 38.04 and 38.18 ms) while in case FROT85 there were no penetrations, and this is the only difference.

4.7.2 Case FROT99

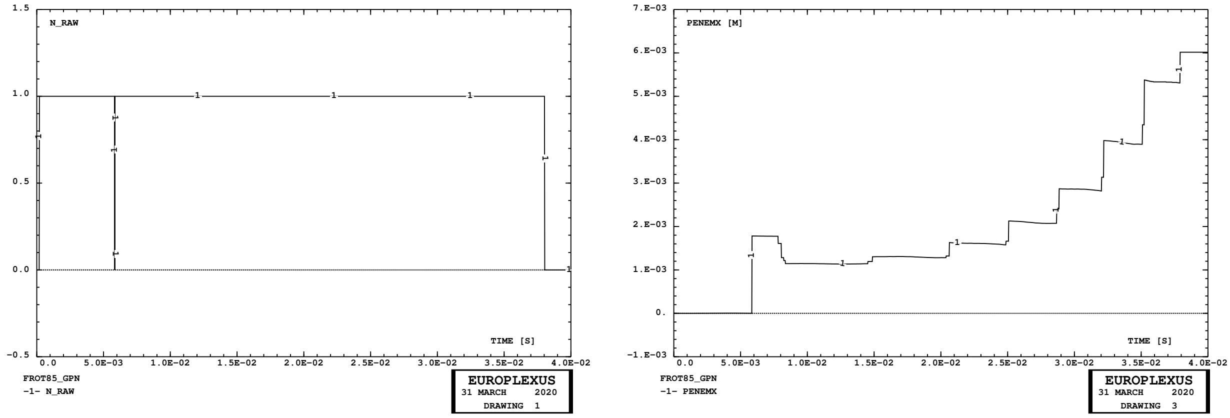
This test is a repetition of case FROT86 (2D, with friction) by using `OPTI GPNS RCEL` to activate redundant constraints elimination during the search for P-L penetrations as described in a previous paragraph.

Remarkably, the result is identical (as far as time curves are concerned) to solution FROT86 (these are not shown for brevity). In addition, also the `.gpn` files are identical.

4.8 Sliding disk problem definition in 3D and original solutions from [25]

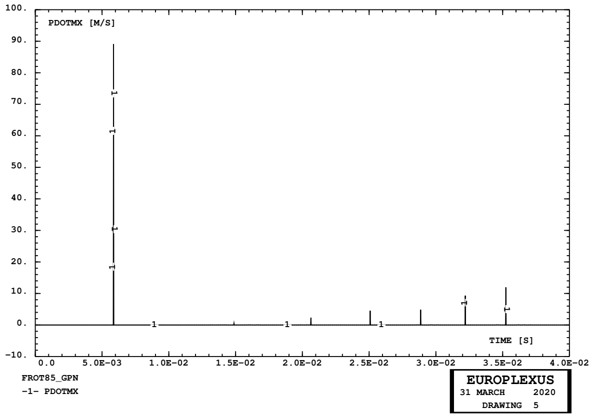
In 3D the problem is set similarly to the 2D case, except for the following details. The disk (projectile) is said (in the text of report [25]) to have a thickness $s = 3$ m, but from the Figures in the same report it appears that the actual thickness is $s = 2$ m (this is the value assumed for the subsequent EPX calculations). The mesh is composed of CUB6 and PRI6 elements. The plane is represented by a parallelepiped of length $L = 10$ m, width $w = 4$ m and height $h = 1$ m (from the Figures, while in the report's text it is erroneously said to have $h = 2$ m). It is meshed exclusively by CUB6 elements ($10 \times 6 \times 2 = 120$ elements), see Figure 36(a).

Like in 2D, the lower base of the target is completely blocked in translation. The forces seem to be applied to the (four) nodes located on the axis of the disk, with each force having a nominal value



(a) Raw penetrations

(b) Max penetration at this step



(c) Max penetration rate at this step

Figure 32: Statistics results for case FROT85

$F_X = -F_Z = 2 \times 10^9$ Pa, with a time variation similar to that of the 2D case but with $t_m = 1.2$ ms, as shown in Figure 36(b).

The calculation is reportedly performed with a constant step $\Delta t = 20$ μs and until a final time $t_f = 40$ ms, which requires 2000 time steps. A first simulation is performed without friction, see the results in Figure 37. In the second simulation friction is assumed, with the same parameters as in the 2D case, see the results in Figure 38.

4.9 Solutions with EPX using `GLIS`

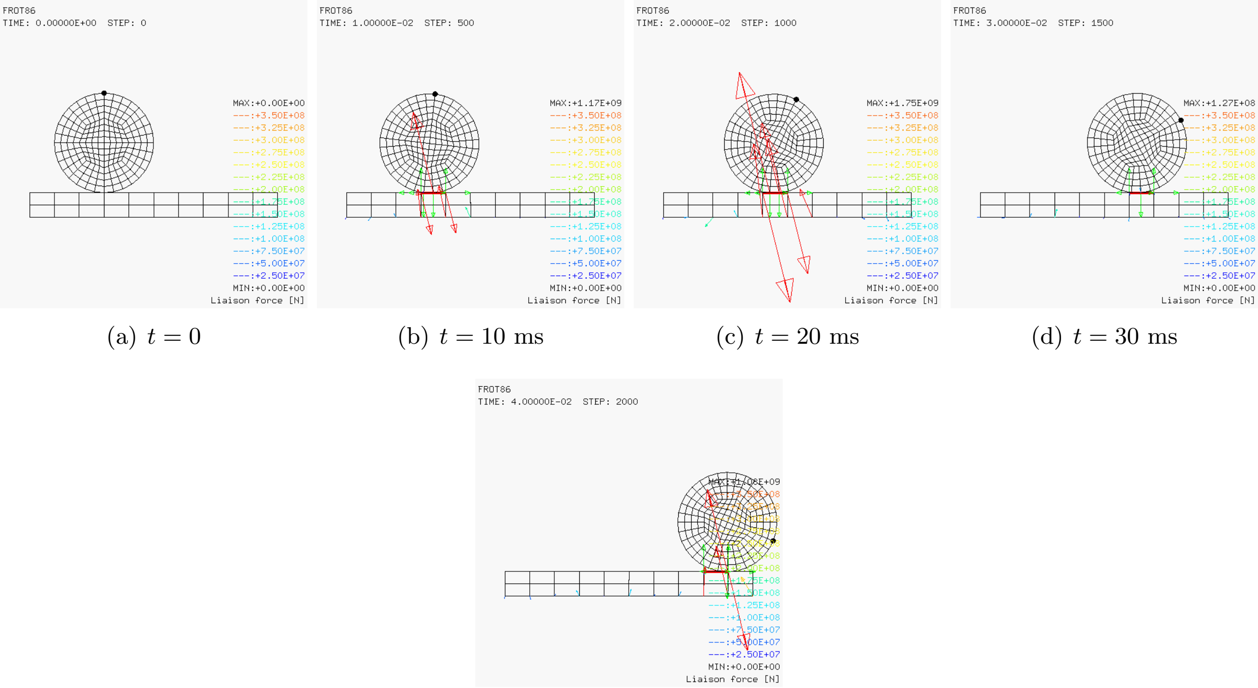
The calculations performed with EPX using `GLIS` are summarized in Table 6.

4.9.1 Case FROT25

This is an attempt to reproduce with EPX the 3D solution without friction. The mesh is shown in Figure 39(a). Note that the disk has been meshed more regularly than in the original solution, by using exclusively hexahedra. The same subdivision of the disk circumference as in the original mesh has been maintained (32 segments).

A zero-mass material point (`PMAT`) is attached to the upper point of the disk in order to appreciate the disk rotation. It appears as a small black dot in the pictures, see Figure 39.

From the result of Figure 39 we observe two things. First, strong hourglassing is developed in the elements of the disk (`CUB6`). The fact that apparently no hourglassing was present in the reference solution may be due to the fact that the reference mesh for the disk contained also some prisms



(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 30$ ms

(e) $t = 40$ ms

Figure 33: Result of test FROT86

Case	Mesh	Description	Final time [ms]	Steps	CPU [s]
FROT25	696 CUB6 1 PMAT	`GLIS`, no friction	24	1200	2.8
FROT26	696 CUB6 1 PMAT	`GLIS`, with friction	24	1200	2.9
FROT27	696 CUB8 1 PMAT	`GLIS`, no friction	24	1200	3.3
FROT28	696 CUB8 1 PMAT	`GLIS`, with friction	24	1200	3.6
FROT29	696 CUB8 1 PMAT	Idem 27, `NORM ELEM`	24	1200	3.2
FROT30	696 CUB8 1 PMAT	Idem 27, raise projectile	24	1200	3.4
FROT58	696 CUB8 1 PMAT	Idem 27, add animation	24	1200	15.8
FROT59	696 CUB8 1 PMAT	Idem 29, add animation	24	1200	15.9
FROT60	696 CUB8 1 PMAT	Idem 59, add `ELIM`	24	1200	16.1
FROT65	696 CUB8 1 PMAT	Idem 60, vertical force 1/4	24	1200	20.5
FROT66	696 CUB8 1 PMAT	Idem 60, force 1/4	40	2000	35.5
FROT67	696 CUB8 1 PMAT	Idem 66, with friction	40	2000	34.8
FROT68	696 CUB8 1 PMAT	Idem 65, symmetric geometry	24	1200	21.6
FROT69	696 CUB8 1 PMAT	Idem 67, 2 `GLIS` surfaces	40	2000	36.2

Table 6: Calculations for the sliding disk problem in 3D using `GLIS`.

(`PRI6`), which might have inhibited the formation of spurious mechanisms.

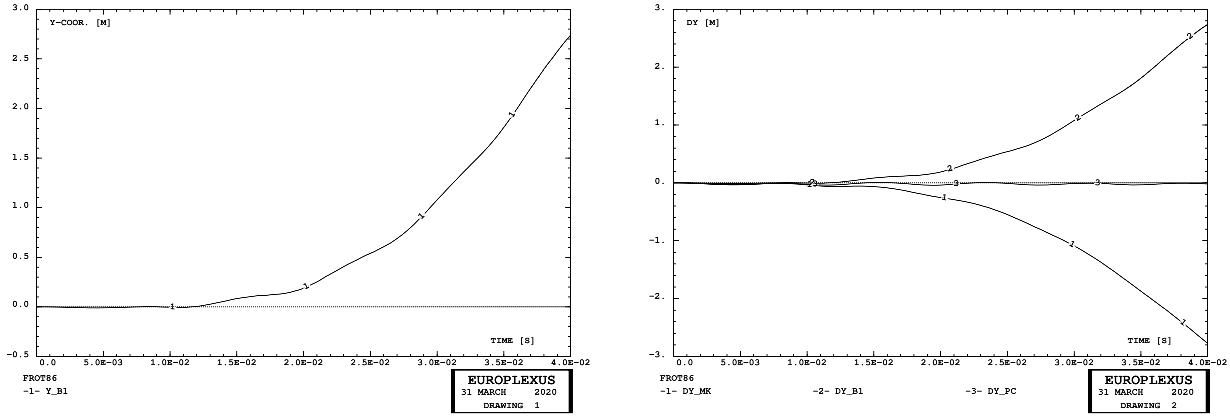
Second, the disk translates much more rapidly than in the reference solution. At 20 ms the disk is already exiting from the plane, while in the reference solution this (apparently) happens only at about 35 ms. In fact, we have limited the EPX simulation only to $t_f = 24$ ms (1200 time steps) instead of the 40 ms (2000 time steps) of the original solution.

This may be due to a difference in the mass of the disk (recall the uncertainty on the disk thickness s mentioned above) or on the applied forces (is the value specified the force at each node or the total force?)

4.9.2 Case FROT26

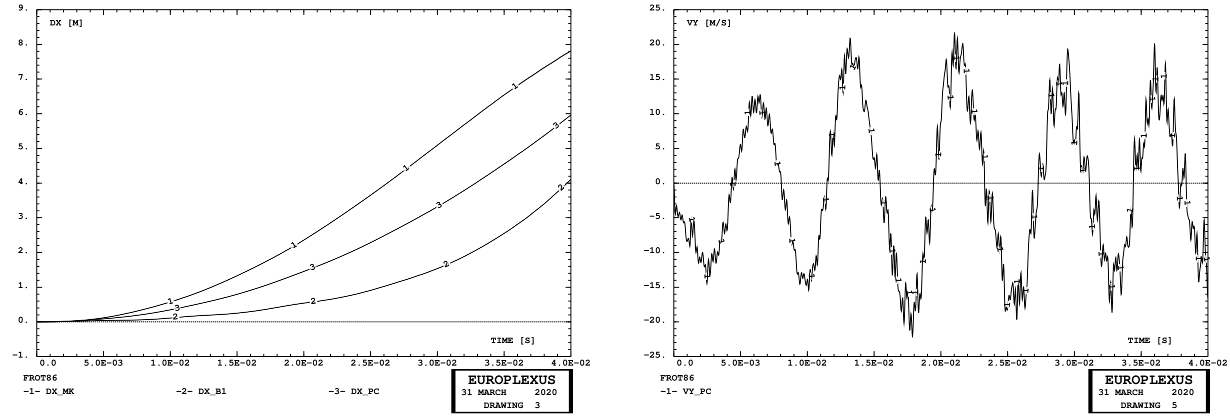
This is a repetition of the previous case by adding the friction.

From the results shown in Figure 40, by ignoring for a moment the strong hourglassing we can see that, like in the previous case, the horizontal translation is much larger than in the reference solution,



(a) Vertical coordinate

(b) Vertical displacements



(c) Horizontal displacements

(d) Vertical velocity

Figure 34: Further results of case FROT86

but also that the disk rotation is larger. Both effects might in principle be due to the use of larger forces than those actually present in the reference.

4.9.3 Cases FROT27 and FROT28

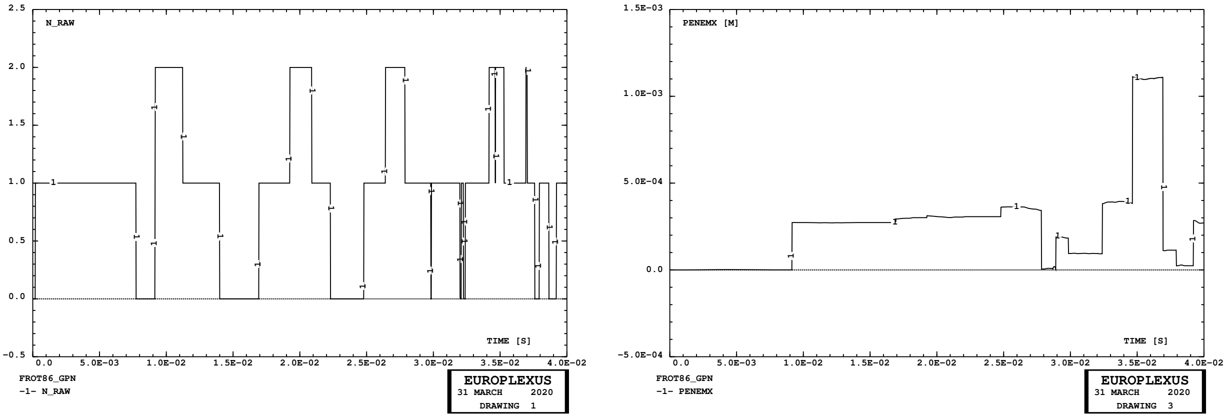
These two cases are repetitions of cases FROT25 (no friction) and FROT26 (friction), respectively, by using the `CUB8` fully-integrated hexahedron element instead of `CUB6`, in an attempt to avoid the hourglassing effects.

The results, presented in Figures 41 and 42 are indeed hourglass-free but, for the rest, they confirm the observations (and the deviations from the reference solutions) already made for cases FROT25 and FROT26.

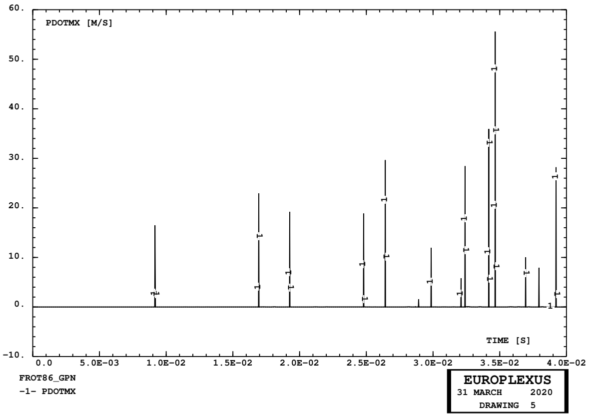
By looking at animations of these results with many more frames than the few presented above, one has the impression that the solution is rather “bouncy” (as already observed in the 2D case) and that the disk (spuriously?) penetrates into the plane quite a bit at certain moments, while getting substantially detached from the plane at other moments.

4.9.4 Case FROT29

This is a repetition of case FROT27 (CUB8, no friction) by activating the `GLIS NORM ELEM` option. The scope is to see whether this may reduce the “bounciness’ of the solution and the spurious penetrations observed.

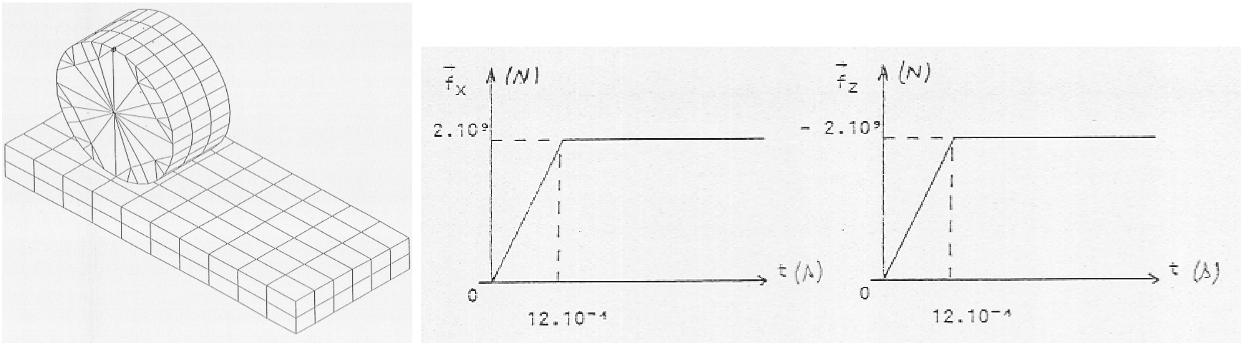


(a) Raw penetrations (b) Max penetration at this step



(c) Max penetration rate at this step

Figure 35: Statistics results for case FROT86



(a) Initial mesh (b) Applied forces

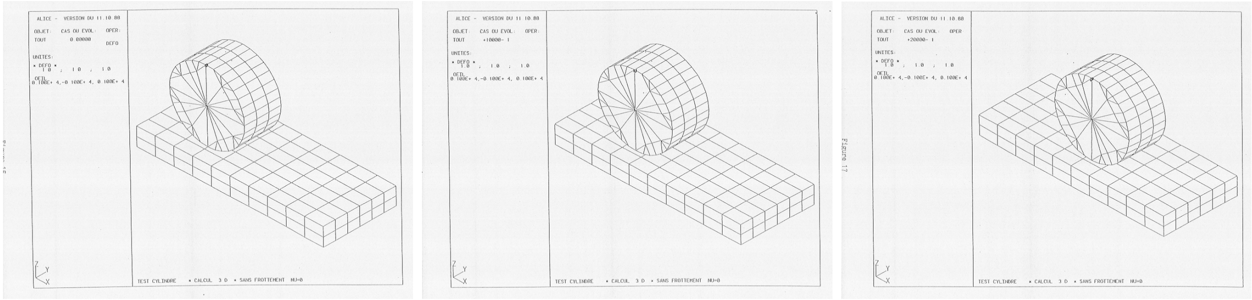
Figure 36: Sliding disk problem in 3D (from [25])

Hovever, the solution is very bad. The projectile rotates strongly in the lateral direction during translation, like if contact would be lost on one part of the (ideal) contact line. See Figure 43.

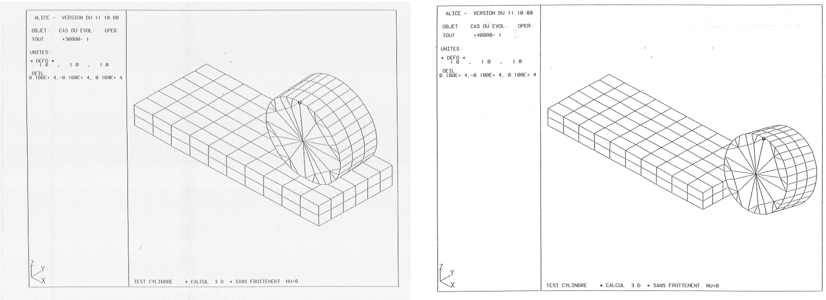
4.9.5 Case FROT30

This is a yet another repetition of case FROT27 (CUB8, no friction) by slightly raising the initial position of the projectile, so as to avoid having contact already in the initial configuration (i.e., at step 0).

Strangely, by raising the projectile 1 mm, there is still contact at step 0 (at least, by looking at the

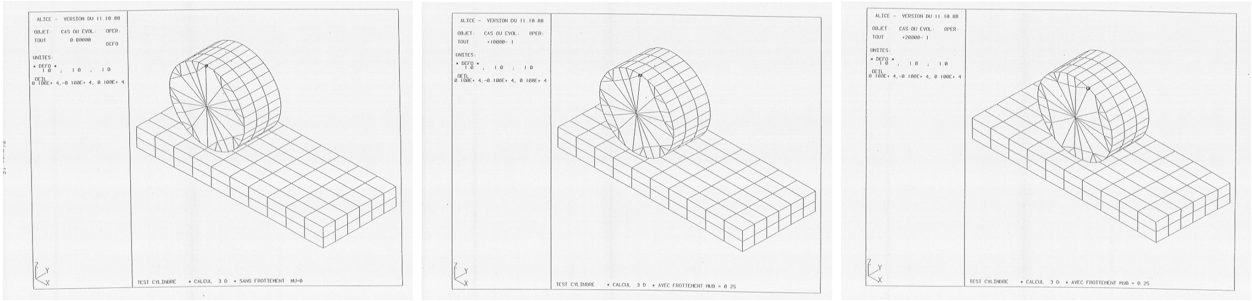


(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms

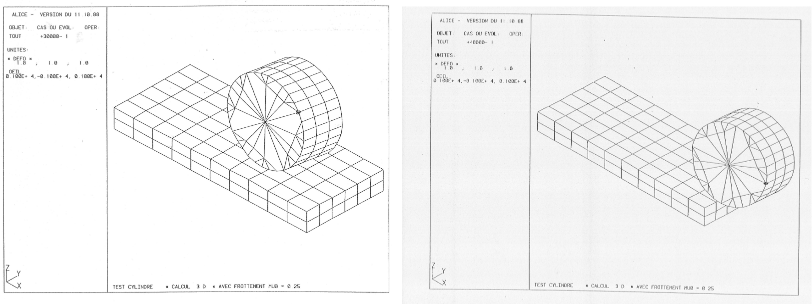


(d) $t = 30$ ms (e) $t = 40$ ms

Figure 37: Result of the sliding disk problem in 3D without friction (from [25])



(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms



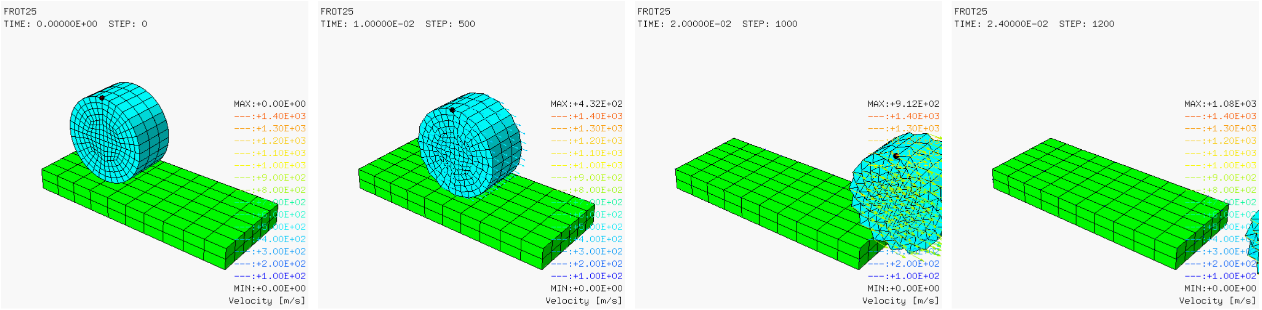
(d) $t = 30$ ms (e) $t = 40$ ms

Figure 38: Result of the sliding disk problem in 3D with friction (from [25])

number of links reported in the file `frot30.lks`. Perhaps this may be due to the fact that contact is "anticipated" in the `GLIS` algorithm, by looking at the (free) position of nodes at the next step? But this seems unlikely since in this problem initial velocities are 0 and also initial external forces are 0.

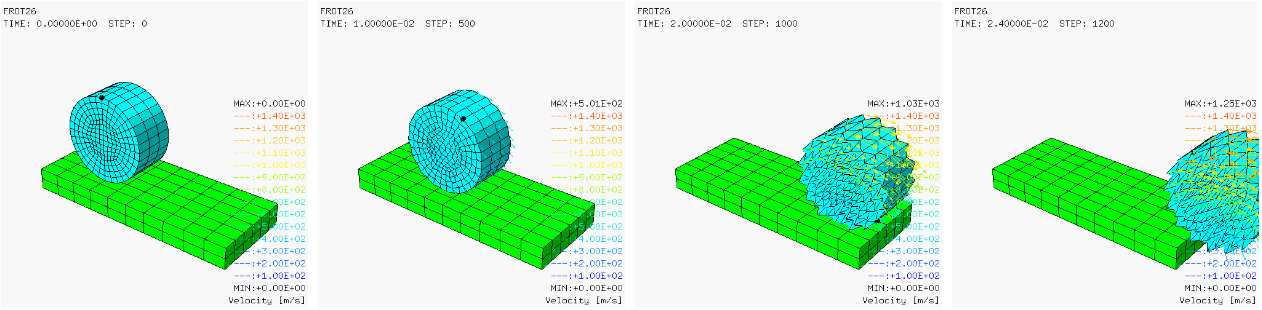
By raising the projectile 10 mm, the first contact occurs at step 40 ($t = 0.8$ ms). The solution is shown in Figure 44.

The solution looks very similar to case FROT27 and the bouncing (spurious penetration) is also equivalent.



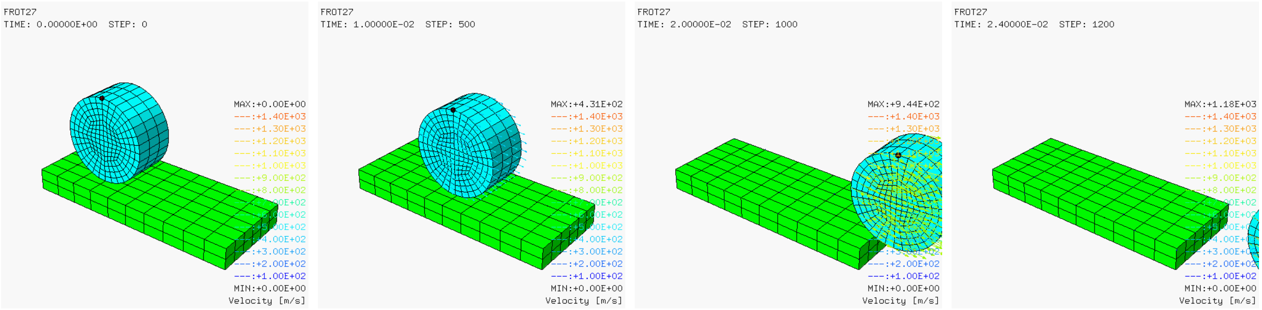
(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 24$ ms

Figure 39: Result of test FROT25



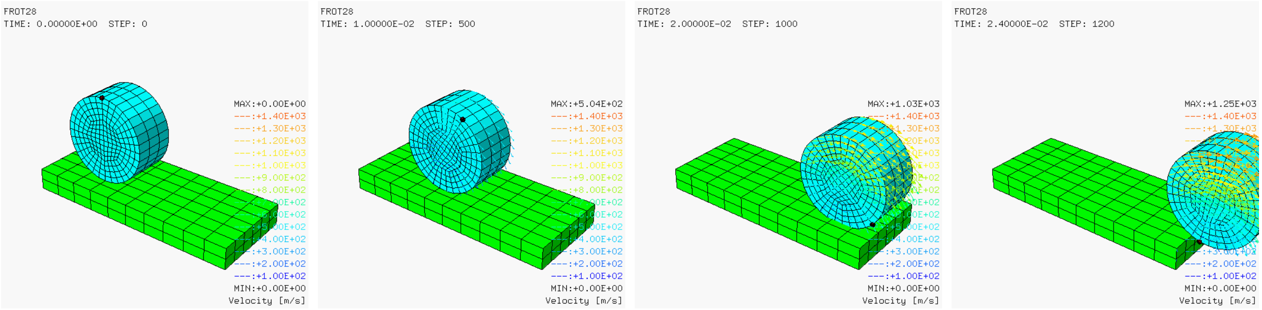
(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 24$ ms

Figure 40: Result of test FROT26



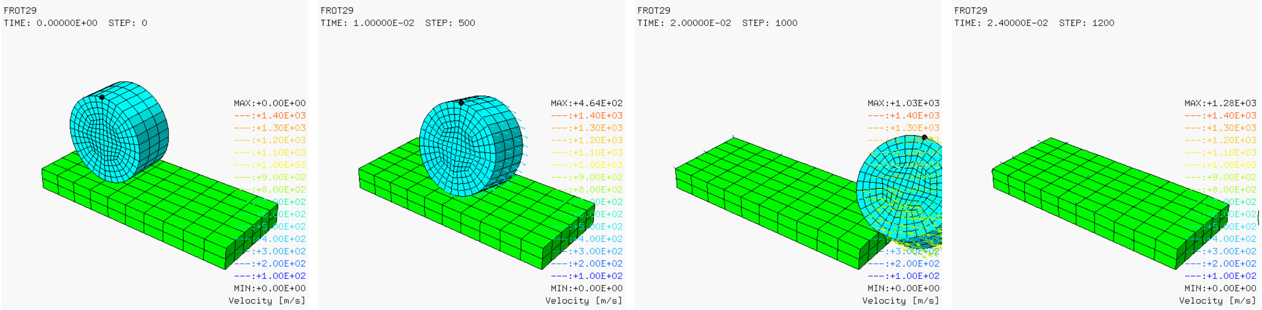
(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 24$ ms

Figure 41: Result of test FROT27



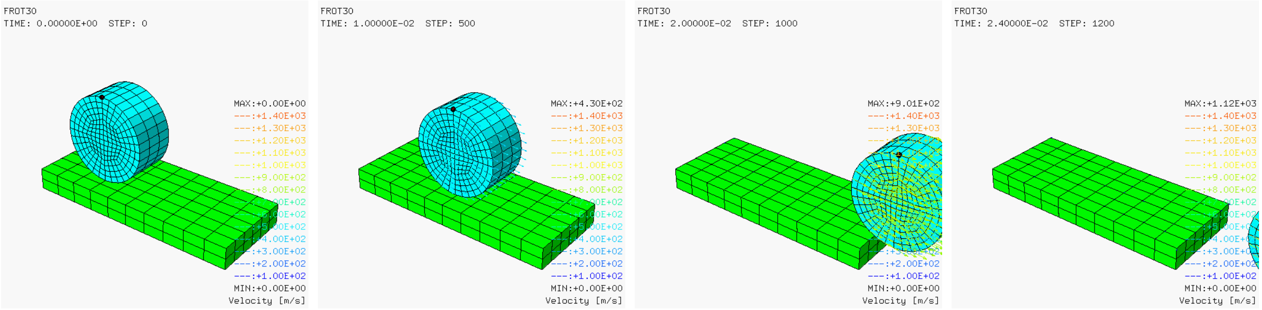
(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 24$ ms

Figure 42: Result of test FROT28



(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 24$ ms

Figure 43: Result of test FROT29



(a) $t = 0$ (b) $t = 10$ ms (c) $t = 20$ ms (d) $t = 24$ ms

Figure 44: Result of test FROT30

4.9.6 Case FROT58

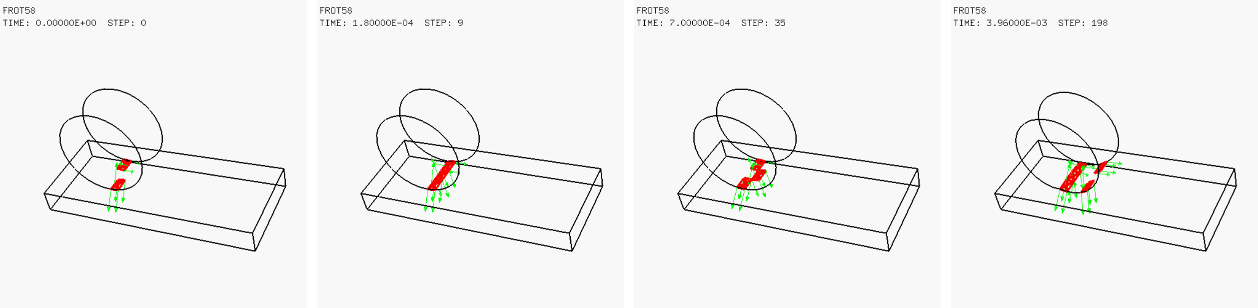
This is a repetition of case FROT27 (CUB8, no friction) by adding an animation (a frame at each step, for a total of 1201 frames) showing the links of type `GLIS` under the form of joints (thick red lines) that connect the linked nodes. This is achieved by the following input commands:

```
OPTI . . .
     LNKS STAT VISU
 . .
SCEN GEOM . . .
          LNKS SHOW GLIS JOIN
```

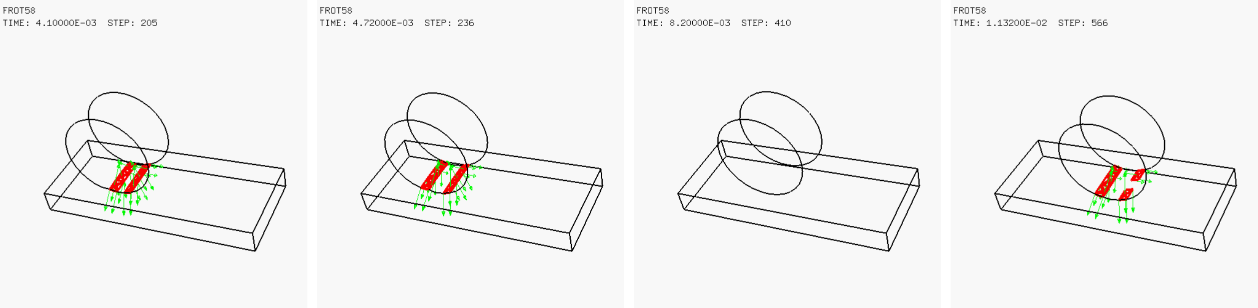
The scope is to inspect the solution in detail and possibly figure out the reasons for the bad behaviour (spurious penetration) observed in this solution. The results are presented in Figure 45 for a few selected frames of the animation. Only the sharp corners (thick black lines) of the structure are drawn, in order to see the “interior”, in particular the contact zone. Only the links of type `GLIS` are shown, while those of type `BLOQ` acting on the base of the target are omitted for clarity.

The green arrows show which of the global degrees of freedom are concerned by links, at each node. The red thick lines connect the linked nodes (one slave node and four master nodes for each link condition). At the initial time there are are only two such links (as confirmed also from the `.lks` file). These appear already slightly un-symmetric, but this might be due to the fact that there is no initial velocity and also the applied force is initially null. The contact seems to become more regular (3 contacts in a row) from $t = 0.18$ ms on.

A slight asymmetry returns at $t = 0.70$ ms, possibly due to the deformation of the bodies. At $t = 3.96$ ms (Figure 45(d)) and again at $t = 4.72$ ms (Figure 45(f)) contact seems to occur along two well-separated lines, indicating that probably the central bottom line of nodes of the disk either is bouncing back (but this is unlikely because the phenomenon has a relatively long duration) or, more probably, has penetrated the target without being detected.



(a) $t = 0$ (b) $t = 0.18$ ms (c) $t = 0.70$ ms (d) $t = 3.96$ ms

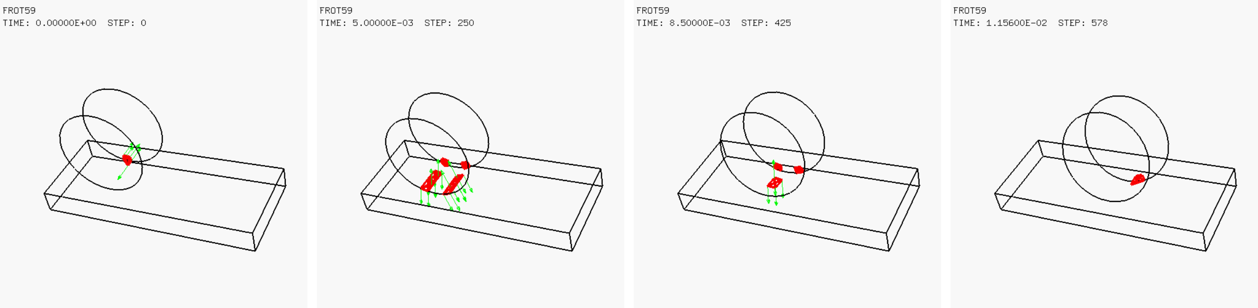


(e) $t = 4.10$ ms (f) $t = 4.72$ ms (g) $t = 8.20$ ms (h) $t = 11.32$ ms

Figure 45: Result (links of type `GLIS`) of test FROT58

4.9.7 Case FROT59

This is a repetition of case FROT29 (CUB8, no friction, `NORM ELEM`) by adding an animation. The resulting `GLIS` links, shown in Figure 46, shed some light on the strange and un-physical solution obtained.



(a) $t = 0$ (b) $t = 5.00$ ms (c) $t = 8.50$ ms (d) $t = 11.56$ ms

Figure 46: Result (links of type `GLIS`) of test FROT59

Already at the initial time we note that there is only one contact and this is of course un-symmetric. Contacts are quite irregular also in the next steps so that the projectile gradually rotates and largely penetrates into the target.

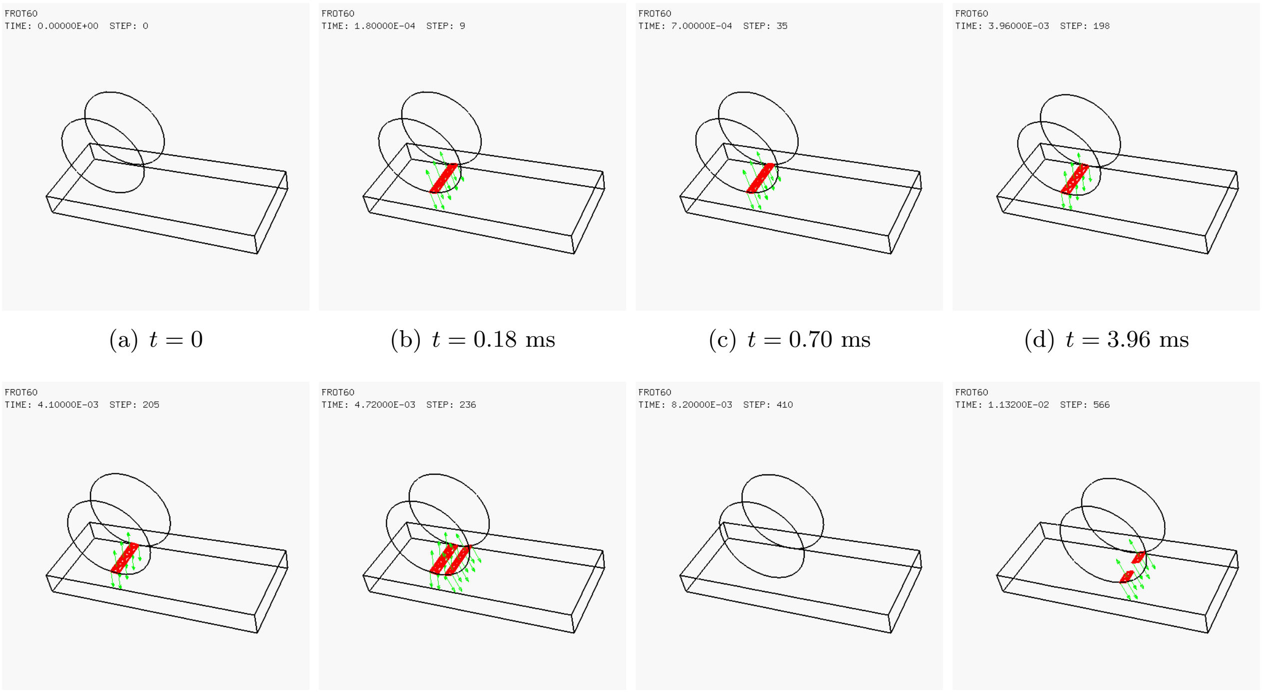
4.9.8 Case FROT60

This is a repetition of case FROT59 (`NORM ELEM`) by adding the `ELIM` optional keyword in the declaration of the sliding surfaces:

```
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 1 ELIM
          MAIT LECT circ TERM
          ESCL LECT base TERM
```

The optional keyword `ELIM` was introduced by F. Bliard in reference [32] in an attempt to solve the issues encountered in `GLIS` when the target body is a continuum formed by just one (unconnected) element or by just one layer of elements.

The resulting links are presented in Figure 47 at the same time instants as for case FROT58 (see Figure 45) for direct comparison. One can see that the contacts are much more regular than in the previous solutions and always symmetric.



(a) $t = 0$ (b) $t = 0.18$ ms (c) $t = 0.70$ ms (d) $t = 3.96$ ms

(e) $t = 4.10$ ms (f) $t = 4.72$ ms (g) $t = 8.20$ ms (h) $t = 11.32$ ms

Figure 47: Result (links of type `GLIS`) of test FROT60

This seems to indicate that the `ELIM` option is highly beneficial and should be recommended whenever the target is a continuum body (`ESCL` keyword), even though consisting of more than one layer of elements (two layers in the present case). Note that it is believed that, when one uses `ELIM`, the option `GLIS NORM ELEM` should be activated as well (like in the present solution FROT60).

4.9.9 Case FROT65

This test is similar to case FROT60 but with the following important differences:

- We apply only the vertical component of the force, in order to study in detail the (elastic) indentation produced by the projectile into the target.
- The total force is reduced to 1/4 of the force applied in the previous solutions (and declared in CEA's report [25]), in an attempt to better match the published solution. This assumes that the value given in the report was the total force and not the force acting on each one of the four nodes along the axis of the disk.
- The value of the force on each node along the axis of the disk is not the same, unlike in the previous solutions. Each of the two "external" nodes (located on the flat faces of the disk) receives 1/6 of the total force, while each of the two internal nodes receives 2/6 of the total force. In this way the applied force is proportional to the mass of each node and better represents an ideal "equally distributed" external load.

The declaration of the applied loads becomes then:

```
COMP NGRO 9 'bloc' LECT base TERM COND Y LT -0.99
            'f1a'  LECT axe TERM COND NEAR POIN 3 2 0
            'f1b'  LECT axe TERM COND NEAR POIN 3 2 2
            'f1'   LECT f1a f1b TERM
            'f2'   LECT axe DIFF f1 TERM
            'b1'   LECT circ TERM COND NEAR POIN 3 0 0
            'b2'   LECT circ TERM COND NEAR POIN 3 0 0.667
            'b3'   LECT circ TERM COND NEAR POIN 3 0 1.333
            'b4'   LECT circ TERM COND NEAR POIN 3 0 2
 . . .
CHAR 1 FACT 2
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
```

Some results are presented in Figure 48. The first picture shows the vertical coordinate of the four "bottom" nodes of the disk. They are initially located at $y = 0$, in contact with the target's upper surface. As it can be seen, the displacement is the same for all four nodes, as expected from the even distribution of the applied load. There are some elastic oscillations and a small rebound ($y > 0$) occurs towards the end of the simulated transient.

The second picture shows the vertical displacement of three nodes, exhibiting oscillations roughly in phase with one another. The two lower curves belong to the marker node (label 1) located at the top of the disk, and to the corresponding point on the disk axis (label 3), where the force is applied. These two curves are very similar as expected by intuition. The third curve (label 2) showing a smaller displacement (as expected) belongs to the node located at the bottom of the disk, which is in contact with the target. This node displaces less than the other two because of the compression of the disk between the axis and the bottom.

The third and final picture shows the horizontal displacement of the three nodes mentioned in the second picture. Ideally, one would expect these displacements to be zero due to the symmetry of the problem (but see also the solution FROT68 in the following and the discussion therein). However, some displacement occurs and this is not the same in all three nodes, indicating also a slight rotation of the disk. It is true that, due to the elastic deformation of the target and of the disk itself (which are not negligible) the contact forces may have some horizontal component, but the solution should be symmetric and therefore such components, if present, should cancel out.

Figure 49 shows the reaction forces at some selected time steps during the transient solution.

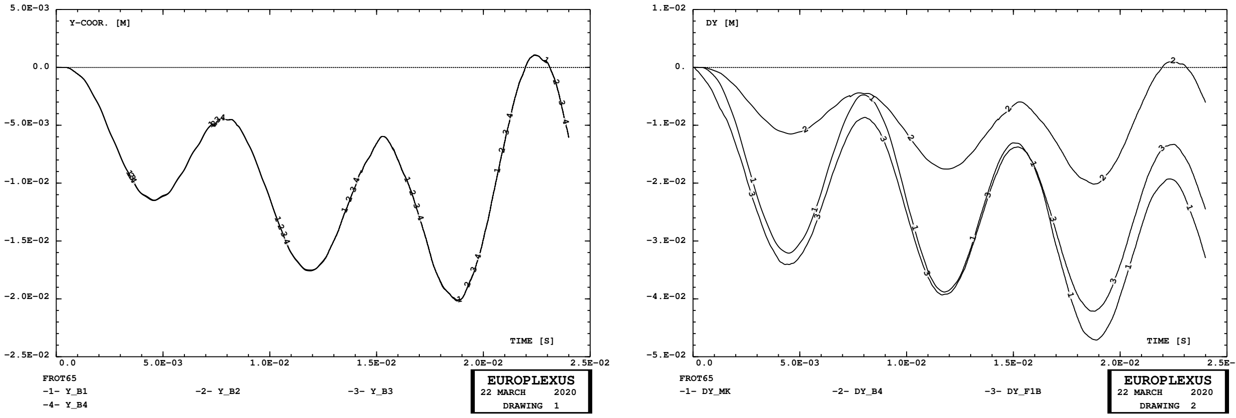
4.9.10 Case FROT66

This case is similar to FROT65 but we add also the horizontal component of the applied force. The loads declaration becomes:

```
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
```

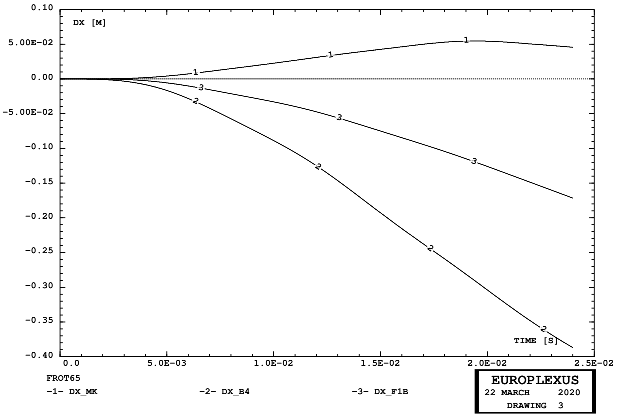
Some results are presented in Figure 50. The first two pictures show the vertical positions and displacements of some reference nodes, presenting a relatively large rebound (larger than in the solution with vertical force only) which produces the observed "bounciness" of the solution. The last picture shows the horizontal displacements. The solution resembles a parabola, which is the correct answer since the horizontal force is constant (except in the relatively short initial part of the transient where it follows a linear ramp). There is a slight spreading of the result for the three monitored nodes, which corresponds to a slight rotation of the projectile. This should not be present if the target would be perfectly rigid, but it is physically plausible when some deformation occurs in the target, as the projectile moves horizontally.

Figure 51 shows the reaction forces at some selected time steps during the transient solution. Note that the final displacement of the projectile (which is about to exit from the plane) is in relatively good agreement with (only slightly behind) the original solution presented in CEA's report [25], thus indicating that the value of the applied force is more correct than in the previous solutions.



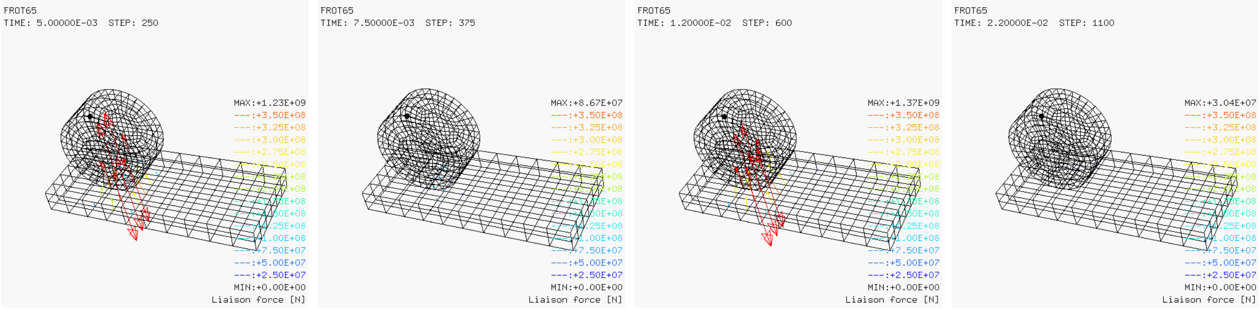
(a) Vertical coordinates

(b) Vertical displacements



(c) Horizontal displacements

Figure 48: Results of case FROT65



(a) $t = 5$ ms (b) $t = 7.5$ ms (c) $t = 12$ ms (d) $t = 22$ ms

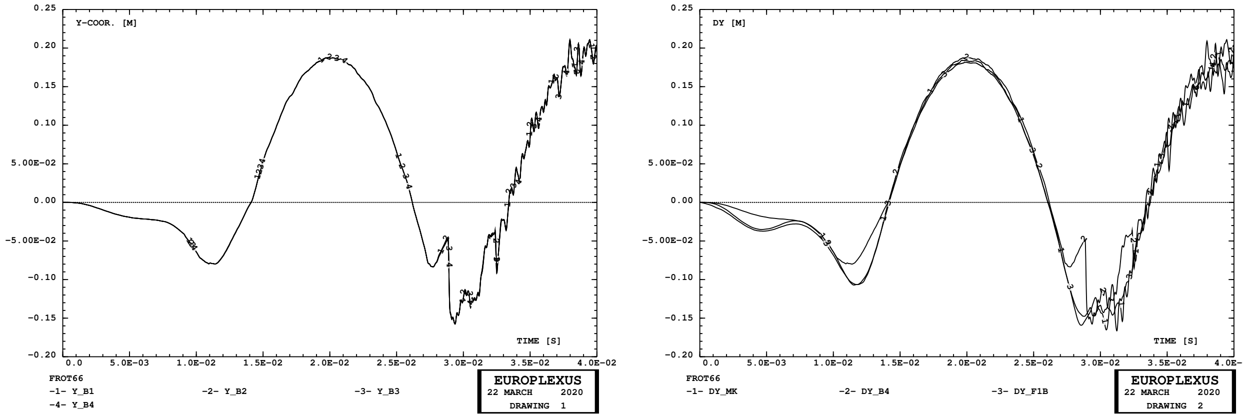
Figure 49: Further results of case FROT65

4.9.11 Case FROT67

This case is similar to FROT66 but we add friction:

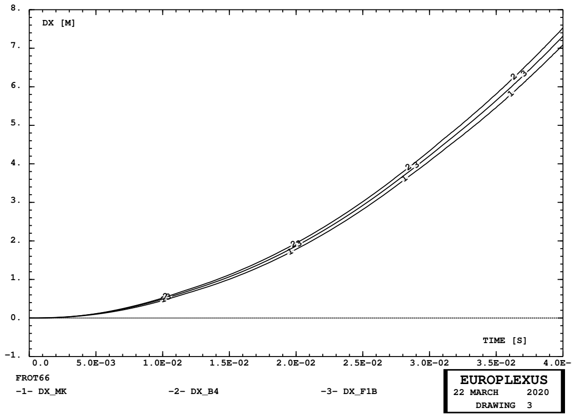
```
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 1 ELIM
          FROT MUST 0.25 MUDY 0.25 GAMM 0.1
          MAIT LECT circ TERM
          ESCL LECT base TERM
```

Some results are presented in Figure 52. The first two pictures show the vertical positions and



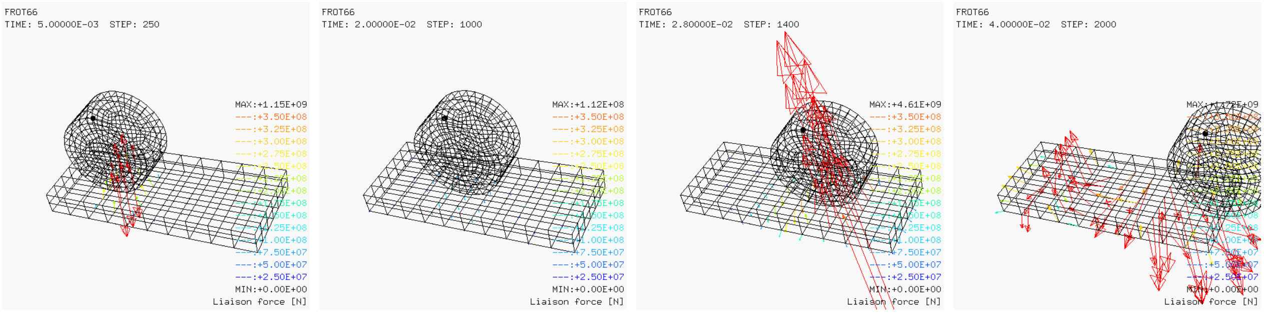
(a) Vertical coordinates

(b) Vertical displacements



(c) Horizontal displacements

Figure 50: Results of case FROT66



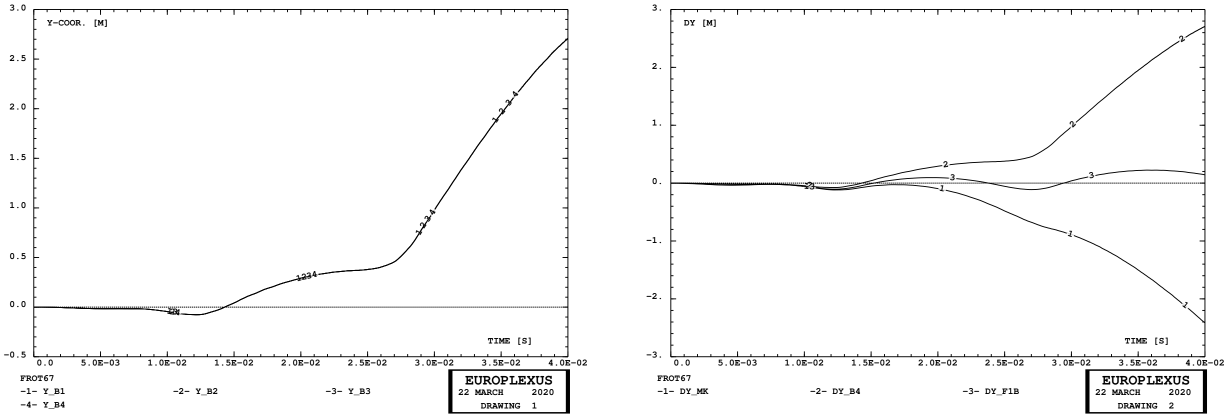
(a) $t = 5$ ms (b) $t = 20$ ms (c) $t = 28$ ms (d) $t = 40$ ms

Figure 51: Further results of case FROT66

displacements of the nodes (initially) at the bottom of the projectile. The apparent large rebound ($Y \gg 0$) in the first picture is due to the fact that the projectile rotates because of friction, so these nodes are no longer the bottom ones as time goes on.

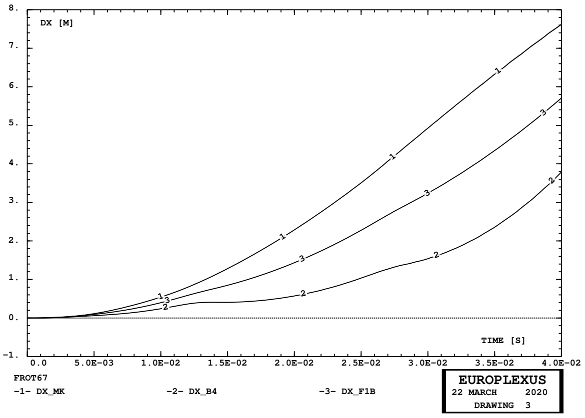
The last picture shows the horizontal displacements. The spreading of the result for the three monitored nodes corresponds to a large rigid-like rotation of the projectile.

Figure 53 shows the reaction forces at some selected time steps during the transient solution. The horizontal component of such forces seems larger than in the previous solution, due to friction. In fact, the final horizontal displacement of the disk is lower than in the friction-less case. Note the position of the marker node (thick black dot) to appreciate the global rotation.



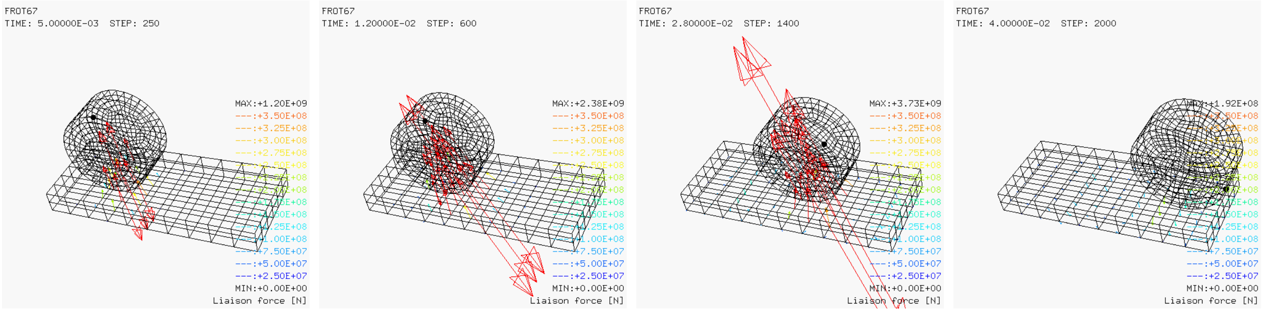
(a) Vertical coordinates

(b) Vertical displacements



(c) Horizontal displacements

Figure 52: Results of case FROT67



(a) $t = 5$ ms (b) $t = 12$ ms (c) $t = 28$ ms (d) $t = 40$ ms

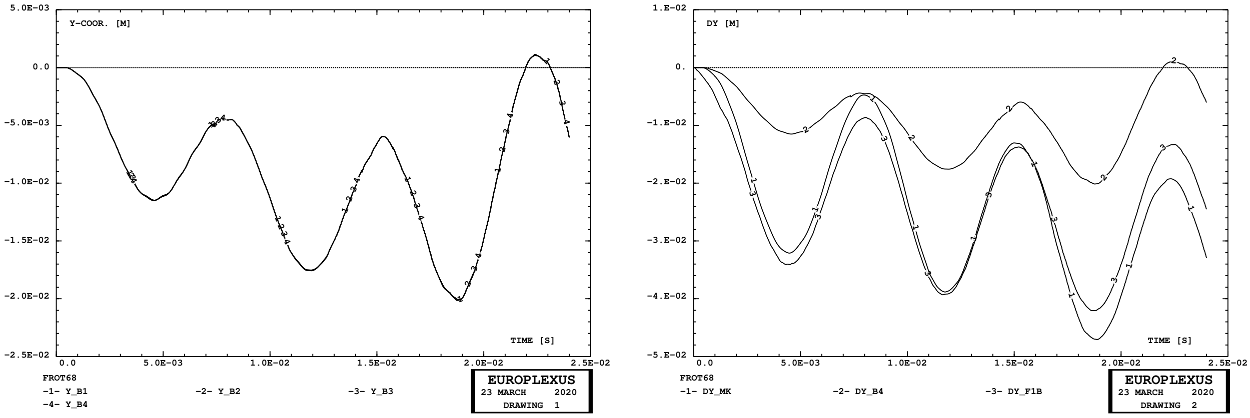
Figure 53: Further results of case FROT67

4.9.12 Case FROT68

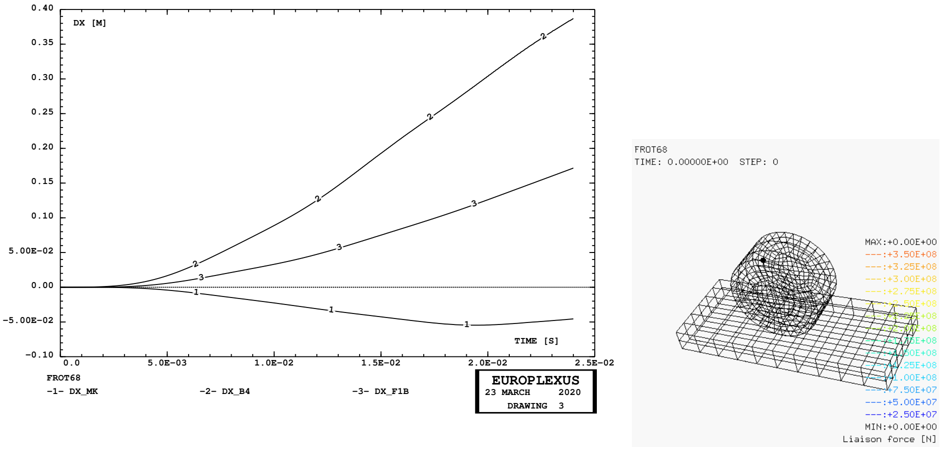
This case is similar to FROT65 (only vertical force) but the problem is rendered (ideally) perfectly symmetric in the geometry by placing the disk at the centre of the plane as shown in Figure 54(d) and not near one extremity, in the hope that the spurious horizontal displacements of the projectile are avoided or greatly reduced.

Note in fact that, if the disk is offset with respect to the meshed part of the plane like in all previous solutions, the model is not perfectly symmetric geometrically. Due to the deformation of the plane, elastic waves are generated in the plane itself and their reflection at the meshed plane boundaries is un-symmetric.

The results are presented in Figure 54. The vertical coordinates and displacements are practically identical to those of case FROT65. The horizontal displacements are, unfortunately, as large as in solution FROT65, strangely enough with the opposite sign. It seems therefore that the asymmetry of the solution comes from the contact model and not from the geometry.



(a) Vertical coordinates (b) Vertical displacements



(c) Horizontal displacements (d) Initial configuration

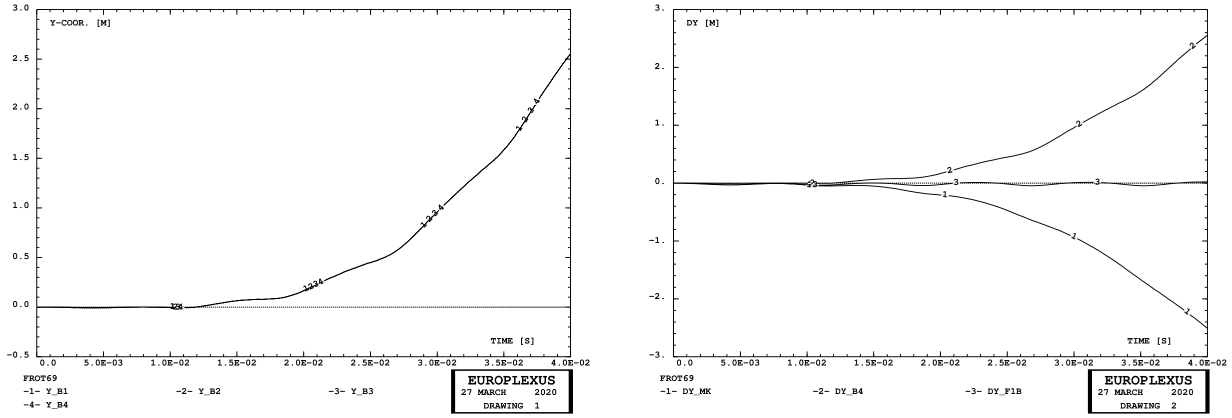
Figure 54: Results of case FROT68

4.9.13 Case FROT69

This case is similar to FROT67 but uses two `GLIS` sliding surfaces instead of one, whereby the roles of master and slave are swapped in the second surface:

```
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 2 ELIM
          FROT MUST 0.25 MUDY 0.25 GAMM 0.1
          MAIT LECT circ TERM
          ESCL LECT base TERM
            ELIM
          FROT MUST 0.25 MUDY 0.25 GAMM 0.1
          MAIT LECT base TERM
          ESCL LECT circ TERM
```

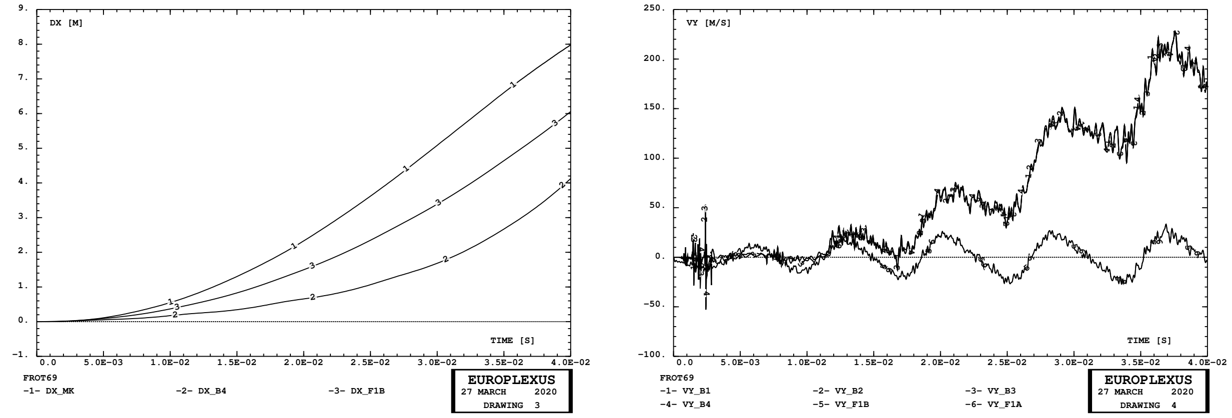
This is in order to obtain a solution more similar to that with `GPIN`, where both bodies are treated as master and as slave at the same time. Some results are presented in Figures 55 (vertical coordinates and displacements), 56 (horizontal displacements and vertical velocities) and 57 (vertical velocities and all velocities). The solution is indeed slightly different from that of case FROT67.



(a) Vertical coordinates

(b) Vertical displacements

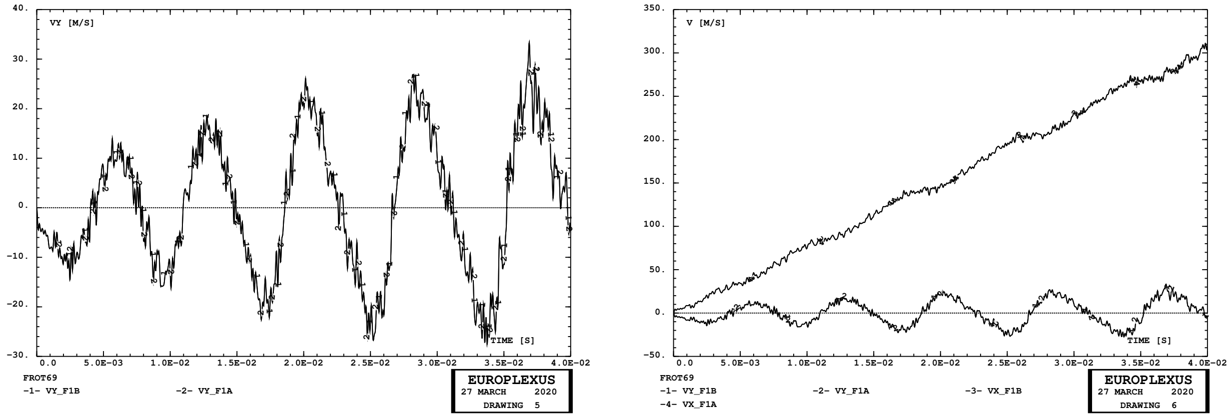
Figure 55: Results of case FROT69



(a) Vertical coordinates

(b) Vertical displacements

Figure 56: Further results of case FROT69



(a) Vertical coordinates

(b) Vertical displacements

Figure 57: Further results of case FROT69

4.10 Reference solutions with `GLIS`

The three solutions (FROT65, FROT66 and FROT67) will be assumed as a reference for the validation of the following solutions using GPINs.

4.11 Solutions with EPX using `GPIN`

The calculations performed with EPX using `GPIN` are summarized in Table 7.

Case	Mesh	Description	Final time [ms]	Steps	CPU [s]
FROT70	696 CUB8 1 PMAT	`GPIN`, vertical force	24	1200	18.2
FROT71	696 CUB8 1 PMAT	Idem 70, add horizontal force	40	2000	30.3
FROT72	696 CUB8 1 PMAT	Idem 71, add friction	40	2000	30.3
FROT73	696 CUB8 1 PMAT	Idem 70, symmetric geometry	24	1200	17.8
FROT74	696 CUB8 1 PMAT	Idem 72, `REB1 REBC`	40	2000	30.7
FROT75	696 CUB8 1 PMAT	Idem 72, `REB2`	[25.5]	[1275]	[19.7]
FROT76	696 CUB8 1 PMAT	Idem 75, larger `DIAM`	40	2000	30.7

Table 7: Calculations for the sliding disk problem in 3D using `GPIN`.

4.11.1 Case FROT70

This solution is similar to case FROT65, where only the vertical component of the force is applied (without friction), but uses the `GPIN` model for contact. The relevant input directives are:

```
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
 . . .
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY LECT circ TERM
          BODY LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
CHAR 1 FACT 2
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
 . . .
OPTI . . .
     GPNS REB1
     LNKS STAT VISU
```

Generalized pinballs with a diameter of 0.05 m are placed in the cylindrical surface of the disk and in the upper surface of the plane target, as shown in Figure 58. The *a priori* rebound method (`REB1`) is used, although this is redundant since it is the default rebound method for the `GPIN` model at the moment of this writing.

Some results are presented in Figure 59. The first picture shows the vertical coordinate of the four “bottom” nodes of the disk. As it can be seen, the displacement is *not* the same for all four nodes, unlike in case FROT65 that uses `GLIS`.

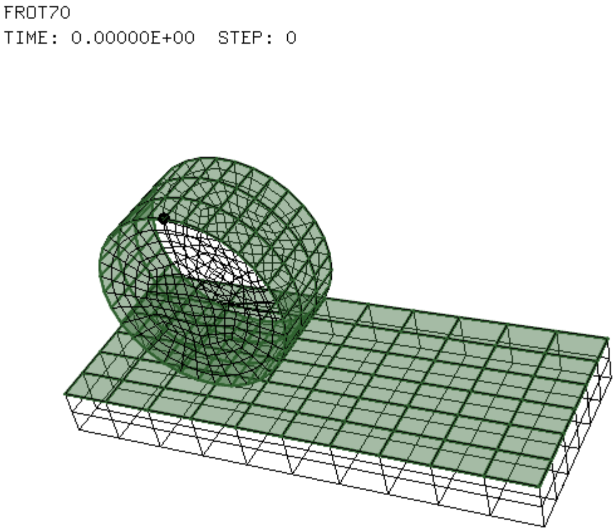
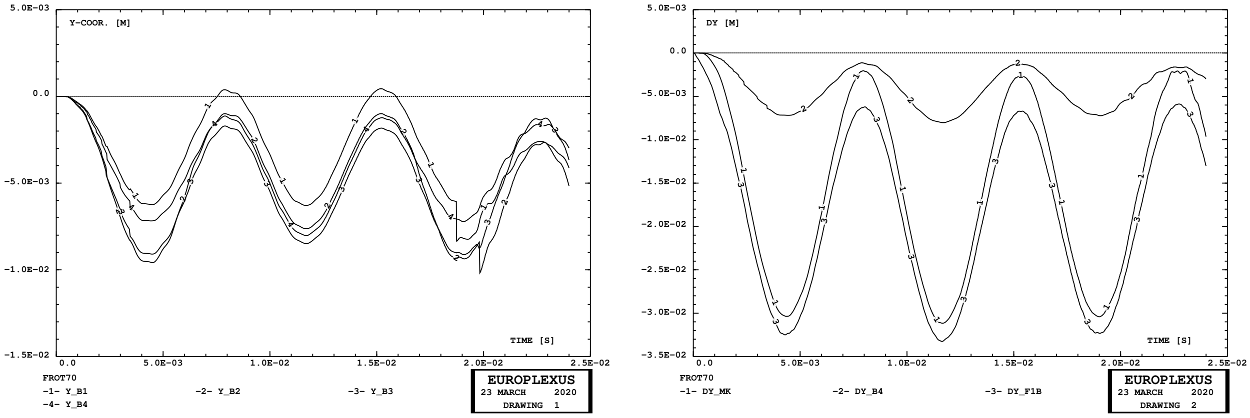
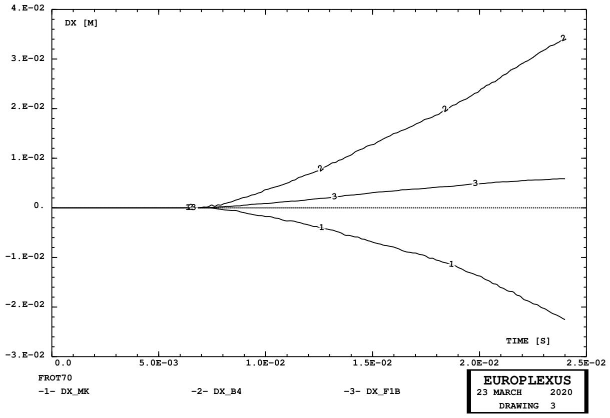


Figure 58: GPINs in case FROT70



(a) Vertical coordinates

(b) Vertical displacements

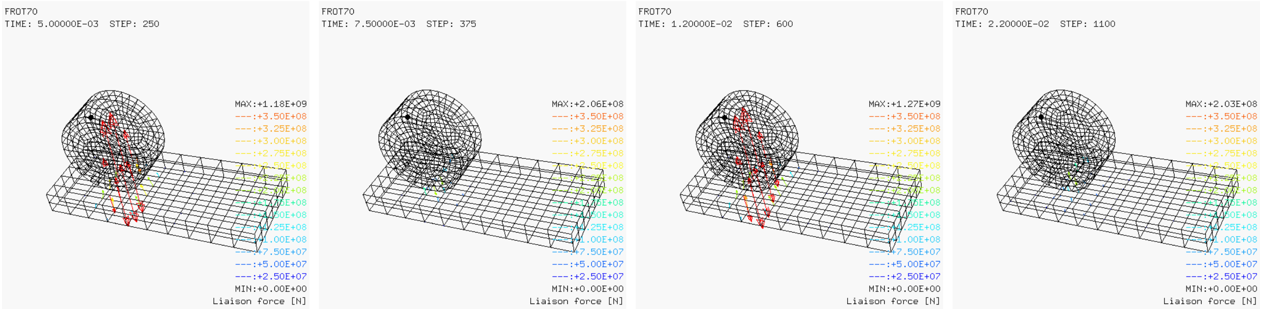


(c) Horizontal displacements

Figure 59: Results of case FROT70

The third and final picture shows the horizontal displacements. These are not negligible, but one order of magnitude less than in case FROT65.

Figure 60 shows the reaction forces at some selected time steps during the transient solution.



(a) $t = 5$ ms (b) $t = 7.5$ ms (c) $t = 12$ ms (d) $t = 22$ ms

Figure 60: Further results of case FROT70

4.11.2 Case FROT71

This solution is similar to case FROT70 but we add also the horizontal component of the applied load (without friction). Thus it is the `GPIN` equivalent to solution FROT66 that used `GLIS`.

Some results are presented in Figure 61. The solution is quite smooth until $t = 32$ ms, but then relatively strong oscillations start to appear as the disk approaches and finally drops past the right edge of the plane. Overall, the solution is quantitavely very different from that using `GLIS`, case FROT66, cfr. Figure 50.

4.11.3 Case FROT72

This solution is similar to case FROT71 but we add friction like in case FROT67 with `GLIS`.

Some results are presented in Figure 62. The simulation seems to start correctly but it becomes completely weird after about 5 ms. This is confirmed by Figure 63 wgere one can see that at a certain moment the disk penetrates into the target, apparently gets “locked” in it and eventually even moves back, although the horizontal load continues to push forward.

4.11.4 Case FROT73

This solution is similar to case FROT70 (only vertical component of the applied load) but the disk is initially placed at the centre of the plane, like in solution FROT68 with `GLIS`.

The results are presented in Figure 64 and are quite similar to case FROT70, unfortunately also as concerns the (spurious) horizontal displacements shown in the third picture. Interestingly, the spurious displacements are also similar to those obtained with `GLIS` in solution FROT68, and this despite the fact that the two contact models used are very different.

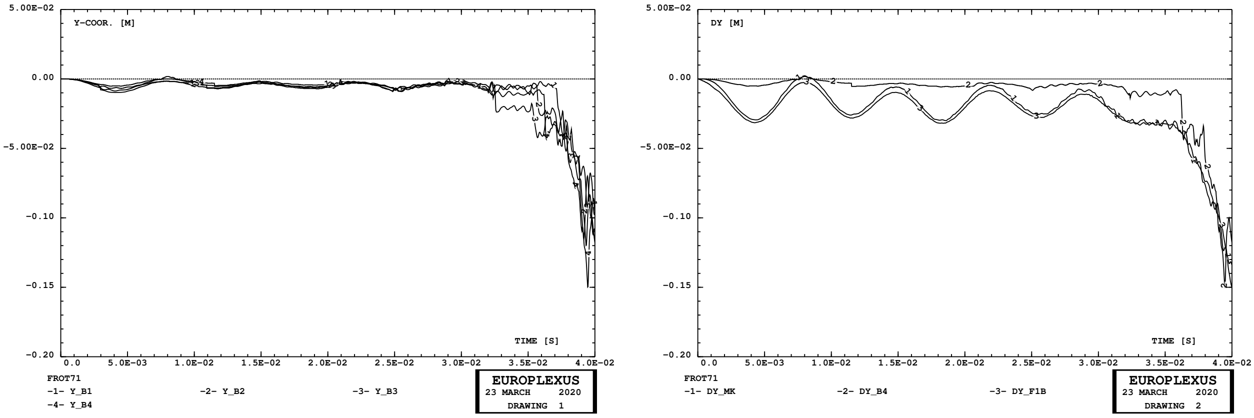
4.11.5 Case FROT74

Focus is placed on the strange “locking” mechanism observed in solution FROT72. This could be due to the rebound algorithm. Therefore, we test the other rebound algorithms available. In the present simulation we add the `REBC` keyword so as to obtain a combined pre-post rebound algorithm.

The results are presented in Figures 65 and 66. The solution is still very bad and only marginally better than case FROT72.

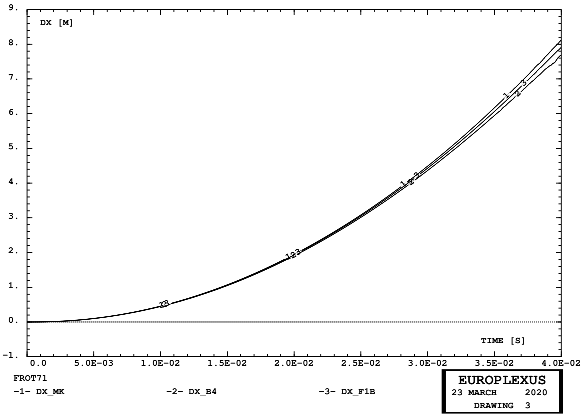
4.11.6 Case FROT75

This test is similar to case FROT74 but uses the *a posteriori* rebound strategy (`REB2`). The calculation failed (“élément croisé”) at step 1276 and $t = 25.5$ ms. Results until that point are shown in Figures 67 and 68.



(a) Vertical coordinates

(b) Vertical displacements



(c) Horizontal displacements

Figure 61: Results of case FROT71

This failure is particularly disappointing because in theory the *a posteriori* rebound algorithm of `GPIN` is relatively similar to the strategy used in the `GLIS` model, which works on this problem.

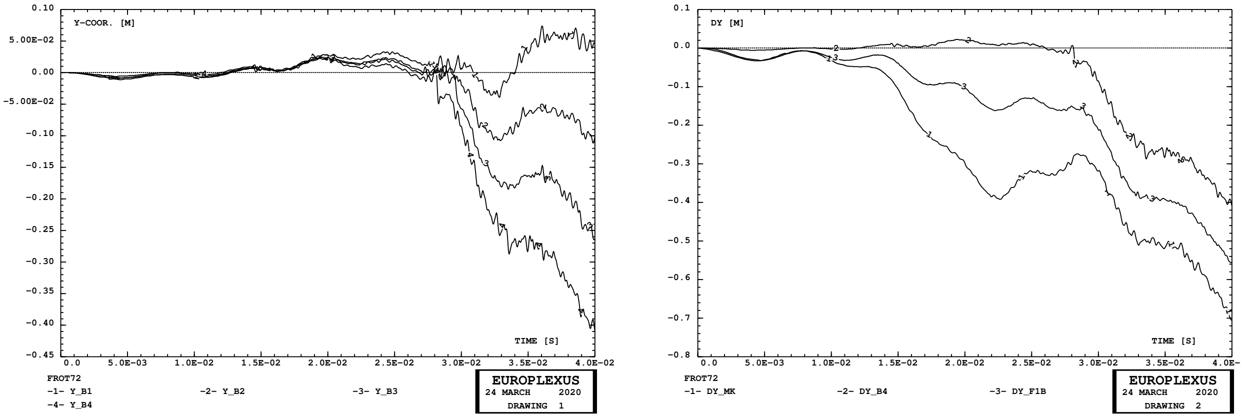
One important difference between the `GPIN` and the `GLIS` models (which will be worth investigating in future tests) is that in `GPIN` both contacting bodies are treated as master and as slave at the same time, thus leading to a larger number of constraints (and to more likely interlocking?) than in the `GLIS` model, which requires the user to define one body as master and the other as slave.

4.11.7 Case FROT76

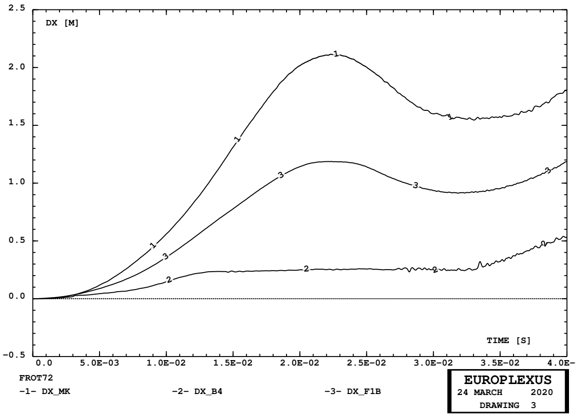
Close inspection of the results and in particular of the interlocking phenomena observed so far raises the question of the size (thickness) of the `GPIN` domains. In theory, the Lagrange multipliers method should ensure that, once contact is established (through a certain penetration), the penetration cannot increase but can only decrease.

But it is also true that, if a penetrating entity (typically, a “slave” node) “overtakes” the master domain in one step, contact will be lost thus potentially destroying the solution. So we repeat the previous case FROT75 by taking (much) thicker domains: instead of uniform domain diameter of 0.05 m everywhere we take a domain diameter of 0.2 m on the cylinder, and of 0.40 m on the plane, as shown in Figure 69.

The calculation runs successfully until the final time $t_f = 40$ ms and the results are presented in Figures 70 and 71. The solution is quite similar (though clearly not identical) to the one obtained with `GLIS`, see test FROT67, Figures 52 and 53 and even more similar to solution FROT69 that uese two `GLIS` surfaces, see Figures 55, 56 and 57. The reaction forces are different (at the particular time

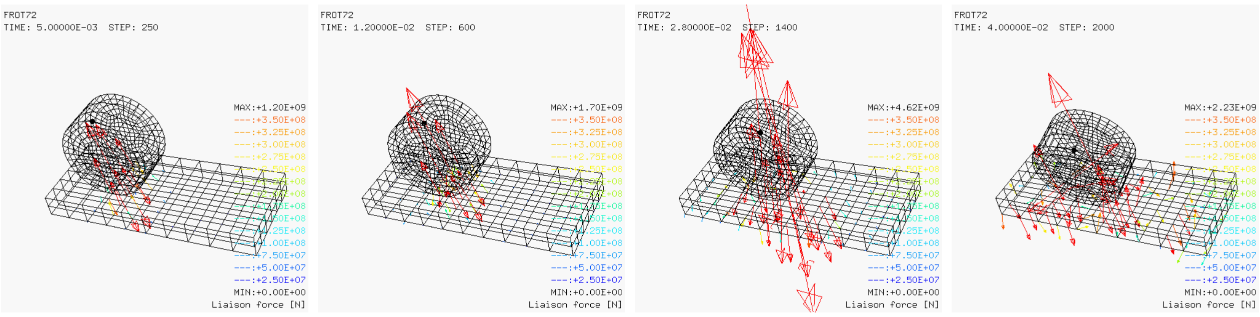


(a) Vertical coordinates (b) Vertical displacements



(c) Horizontal displacements

Figure 62: Results of case FROT72



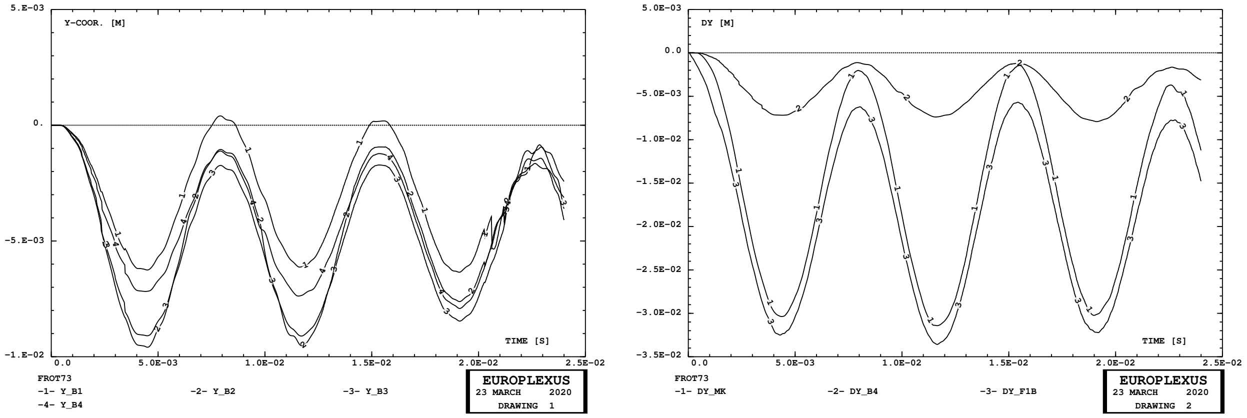
(a) $t = 5$ ms (b) $t = 12$ ms (c) $t = 28$ ms (d) $t = 40$ ms

Figure 63: Further results of case FROT72

instants chosen) but the overall final position and final rigid rotation of the disk are quite similar.

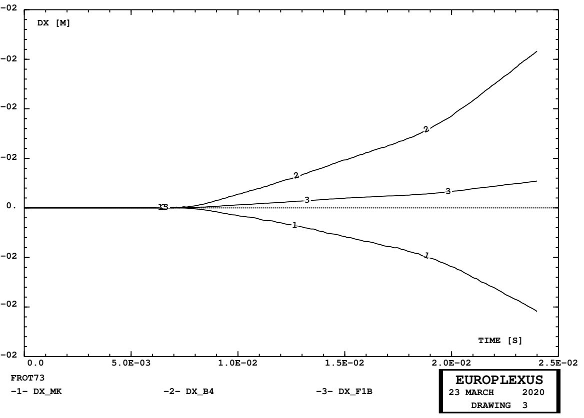
Encouraging as it is, this result is not satisfying because it is unclear why such thick domains are needed in order to avoid the loss of contact. After all, in this problem the vertical force is applied quite gradually during the first part of the solution, so the vertical component of the projectile velocity should remain small and it is difficult to understand how a penetrating node could overtake a master domain in just one step.

To clarify this point, some additional results are inspected. Figure 72 (first picture) shows the vertical velocity of the nodes at the two extremities of the disk axis. Note that values of the order of 50 m/s are reached, which is relatively high. True, this is the absolute velocity while as far as penetration



(a) Vertical coordinates

(b) Vertical displacements



(c) Horizontal displacements

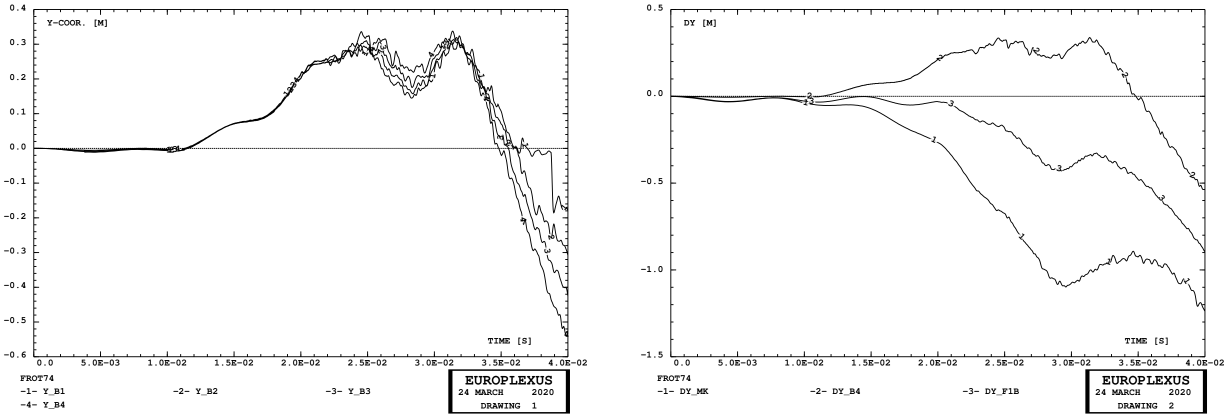
Figure 64: Results of case FROT73

is concerned, the important parameter is the *relative* velocity between a penetrating entity (typically a node of the disk) and a GPIN domain (typically a face of the target), but this is hard to visualize since the contact changes in a highly dynamic way. The second picture shows the horizontal velocities compared with the vertical ones. The horizontal component grows linearly as expected, under the constant applied force, and reaches a value of about 300 m/s at the end of the simulation.

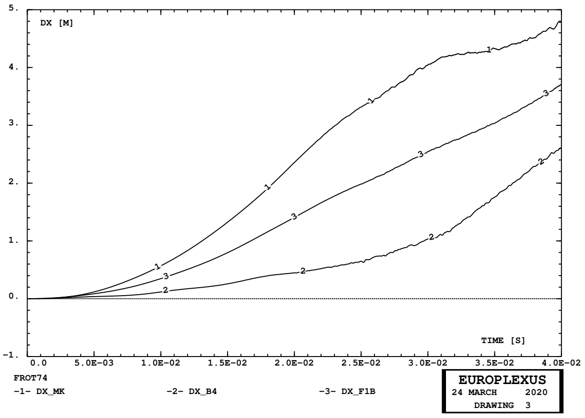
Finally, some interesting information can be obtained by activating the option `OPTI GPNS STAT`. This causes the writing on a file with the extension `.gpn` of a line at each times step containing the step number `STEP`, the time `T`, the number of raw penetrations `N_RAW`, the number of penetrations remaining after the *a priori* rebound `N_REBO`, the maximum penetration `PENEMX` during the current step, the maximum penetration `PENEMAX` so far, the maximum penetration rate `PDOTMX` during the current step, and the maximum penetration rate `PDOTMAX` so far.

A utility program `gpn2plot.exe` is written, which allows to read the `.gpn` file and to produce a corresponding `_gpn.pun` file in the EPX “punch” format. The punch file can then be read by EPX to produce the drawings of the various quantities, as shown in the next Figures for the current test case.

Figure 73 shows the number of penetrations at each step, reaching a maximum of about 17. Figure 74 shows the maximum penetration at each step and so far. Note that a value as high as 4.7 cm is reached at some point of the simulation. Figure 75 shows the maximum penetration rate at each step and so far. Note that a value as high as 440 m/s is reached at some point of the simulation.

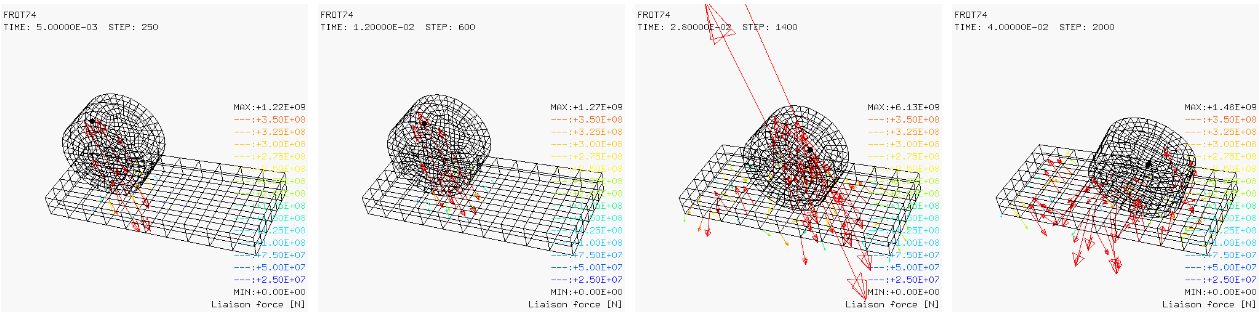


(a) Vertical coordinates (b) Vertical displacements



(c) Horizontal displacements

Figure 65: Results of case FROT74



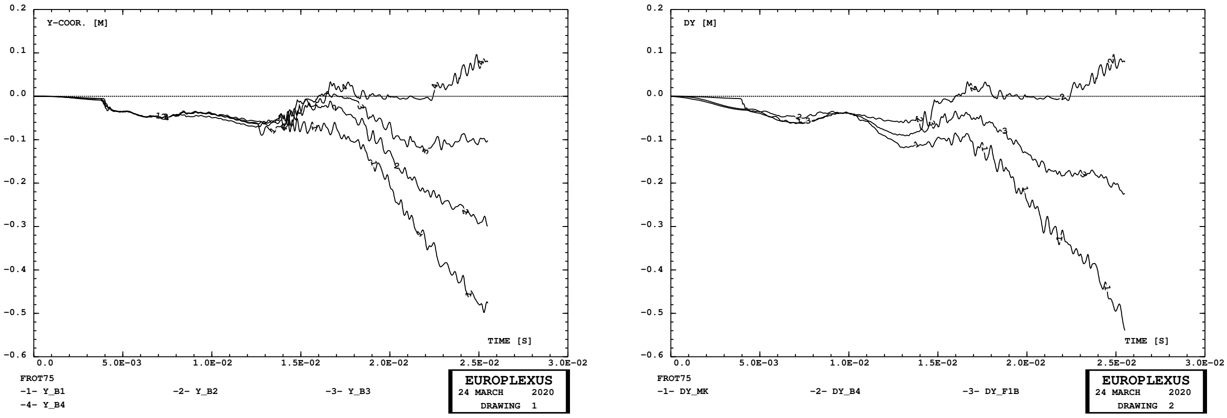
(a) $t = 5$ ms (b) $t = 12$ ms (c) $t = 28$ ms (d) $t = 40$ ms

Figure 66: Further results of case FROT74

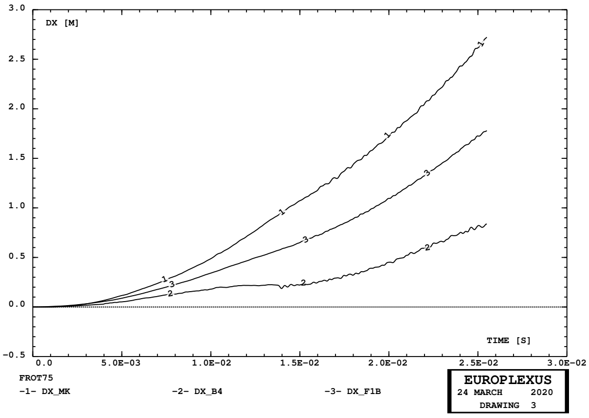
4.11.8 Cases FROT77 and FROT78

The result shown in Figure 74 for the previous simulation seems to indicate that the value of `DIAM` (5 cm) used in the simulations prior to case FROT76 is indeed insufficient to avoid the spurious loss of contact. In fact, since the GPINs used in this test are single-sided, the diameter should be at least twice the maximum penetration, that is $2 \times 4.7 = 9.4$ cm.

Therefore, in simulation FROT77 we tentatively set `DIAM` to 10 cm everywhere. However, the result is very bad, as shown in Figure 76(a). The next solution, FROT78, uses a diameter of 20 cm everywhere and, once again, the solution is bad, see Figure 76(b), while we know from case FROT76

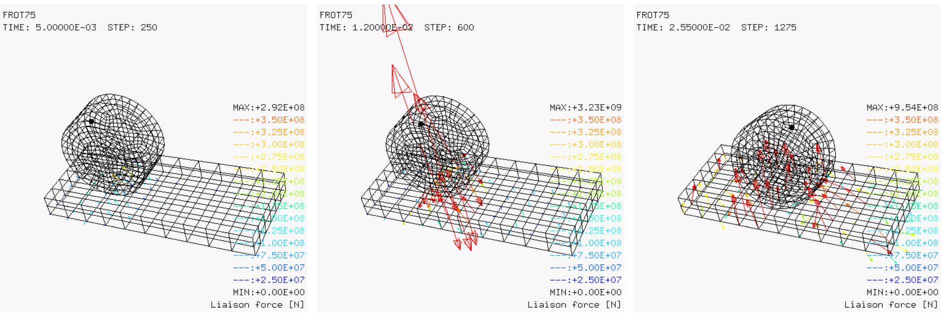


(a) Vertical coordinates (b) Vertical displacements



(c) Horizontal displacements

Figure 67: Results of case FROT75



(a) $t = 5$ ms (b) $t = 12$ ms (c) $t = 25.5$ ms

Figure 68: Further results of case FROT75

that by taking a diameter of 40 cm for the GPINs in the target gives the correct solution.

A slight dependence of the solution upon the chosen GPIN diameter can perhaps be expected, but not as much as shown here. Moreover, the maximum penetration reported from the `.gpn` file is 5 cm (the limit) in case FROT77 and 10 cm (again, the limit) in case FROT78. This does not seem to make much sense.

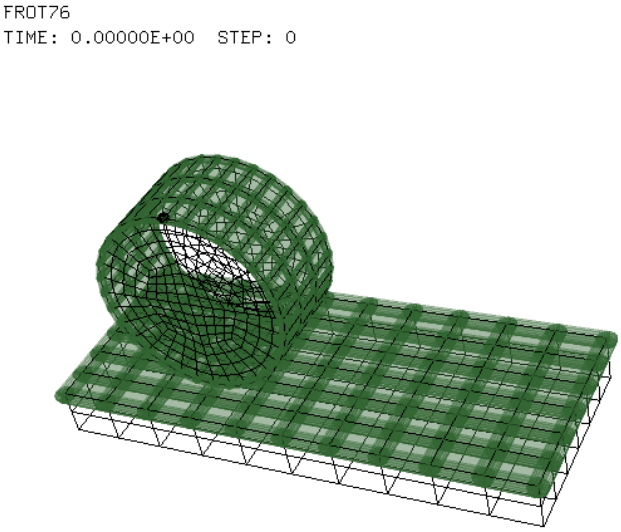
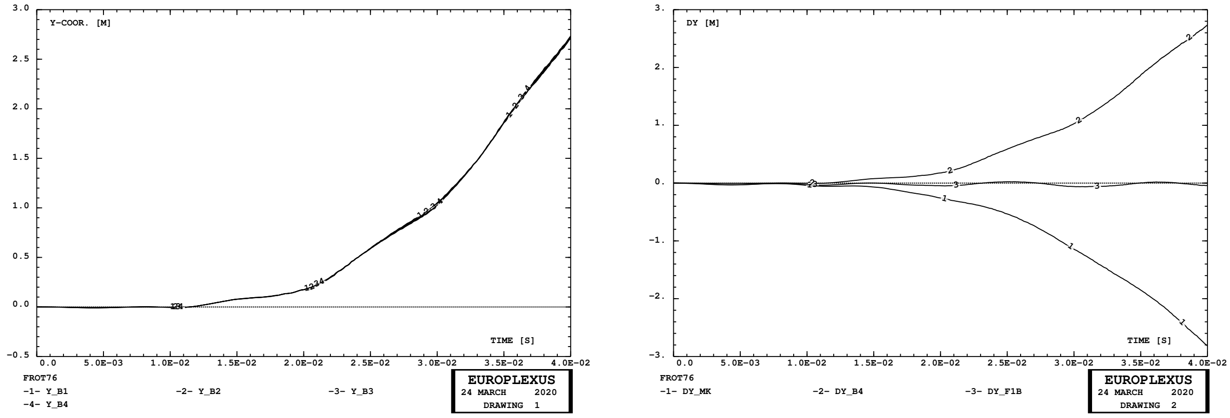
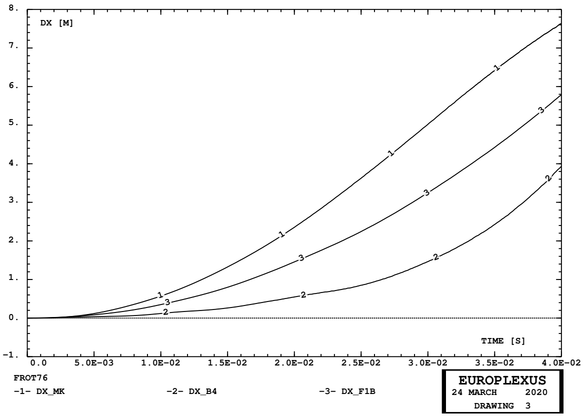


Figure 69: GPINs in case FROT76



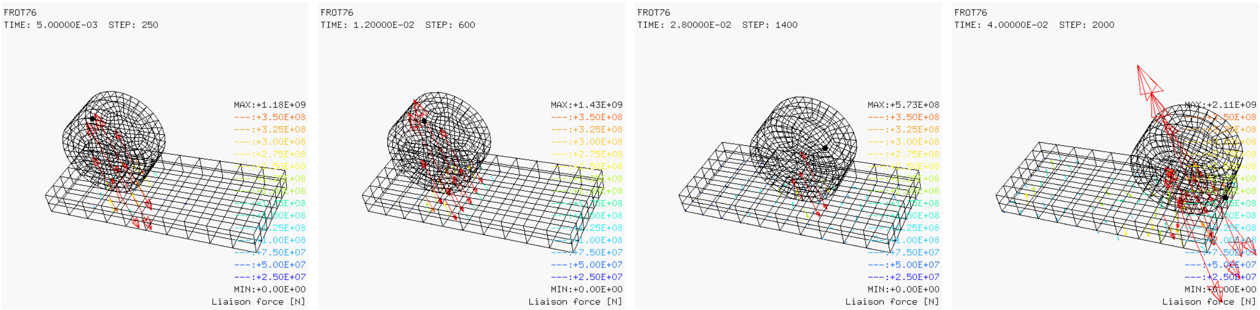
(a) Vertical coordinates

(b) Vertical displacements



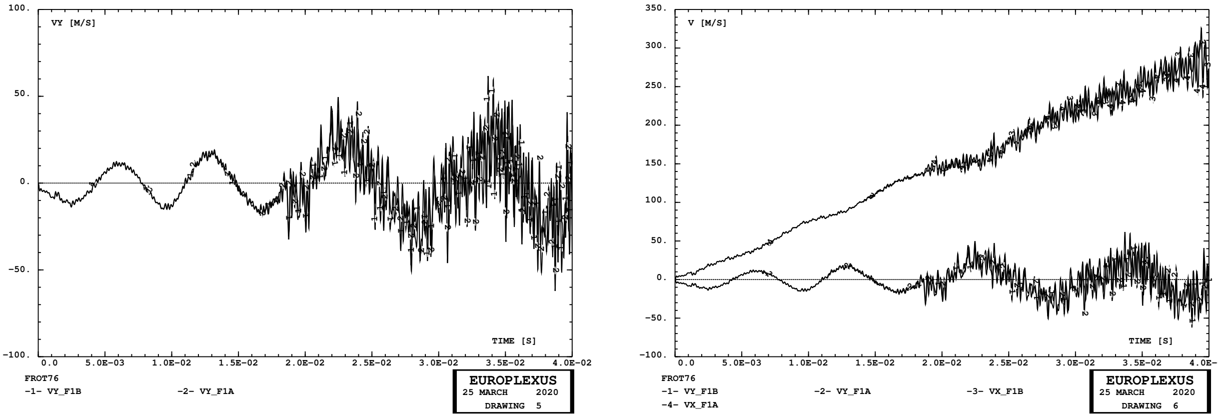
(c) Horizontal displacements

Figure 70: Results of case FROT76



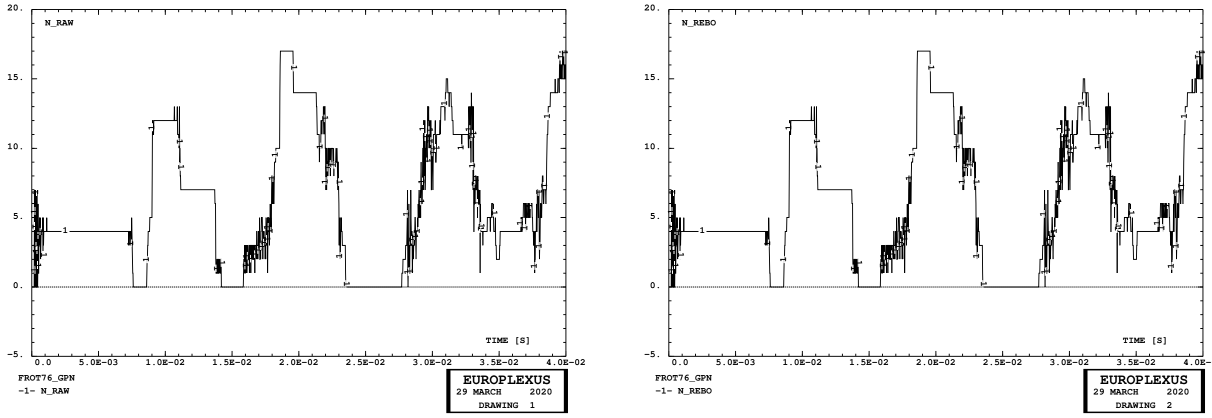
(a) $t = 5$ ms (b) $t = 12$ ms (c) $t = 28$ ms (d) $t = 40$ ms

Figure 71: Further results of case FROT76



(a) Vertical velocities (b) Vertical and horizontal velocities

Figure 72: Velocities in case FROT76

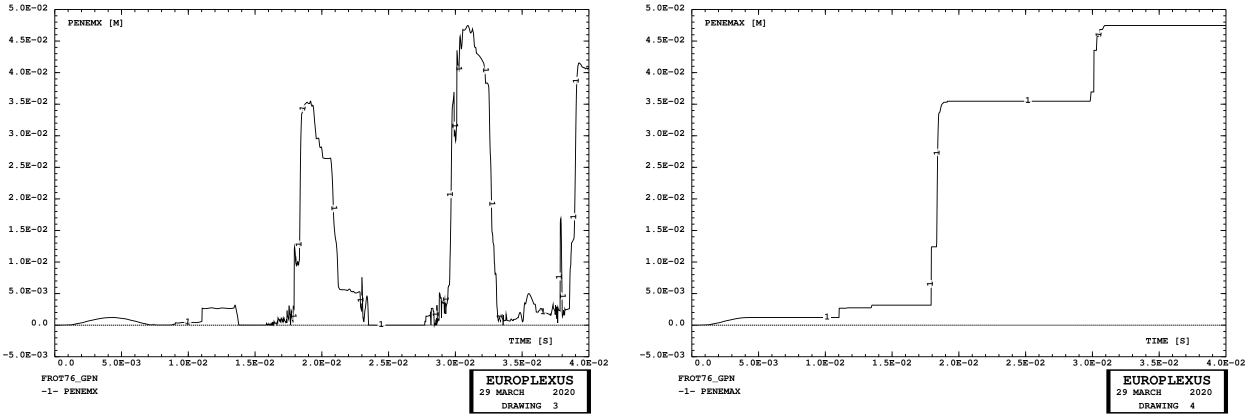


(a) Raw penetrations (b) Penetrations after rebound

Figure 73: Number of penetrations in case FROT76

4.12 Solutions with EPX using `GPIN` in master/slave mode

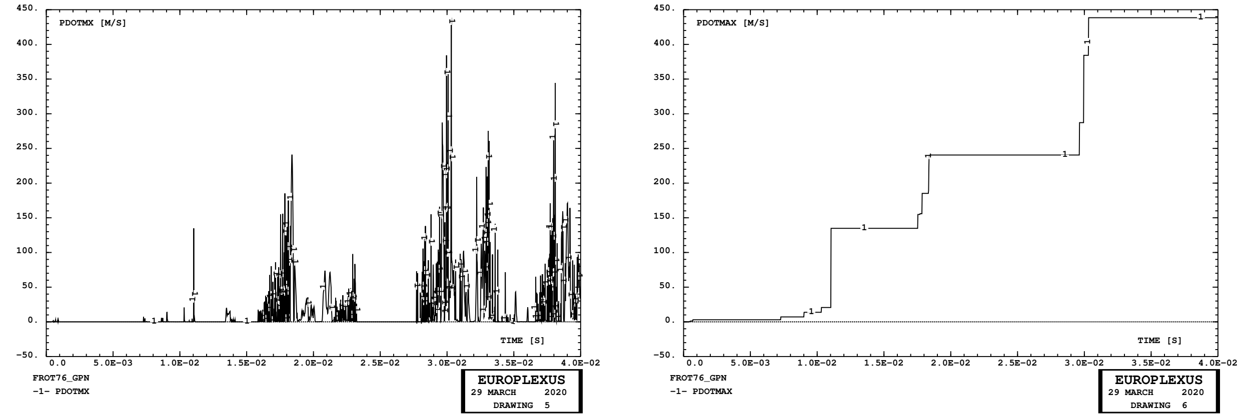
In order to test the master/slave strategy described in a previous paragraph, we perform some simulations, as listed in Table 8.



(a) Max penetration at this step

(b) Max penetration so far

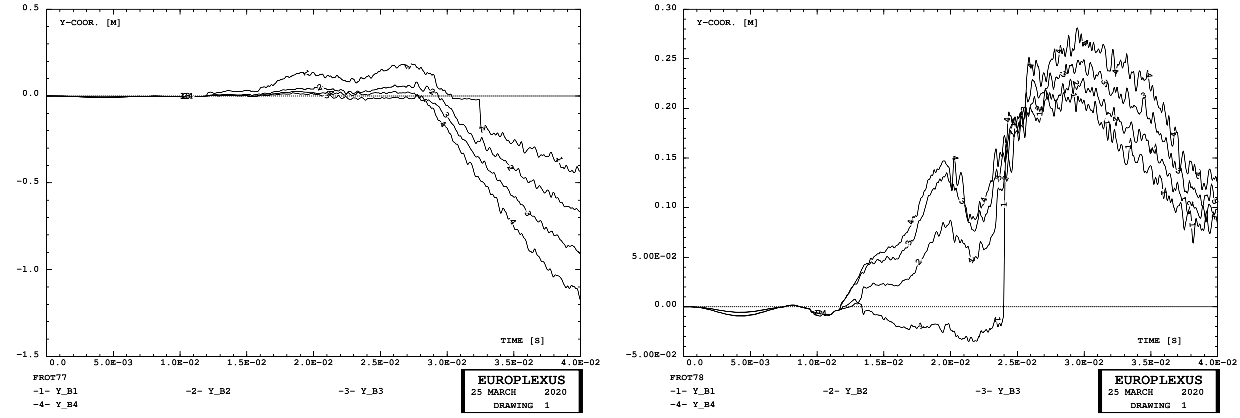
Figure 74: Maximum penetration in case FROT76



(a) Max penetration rate at this step

(b) Max penetration rate so far

Figure 75: Maximum penetration rate in case FROT76



(a) Vertical coordinates in case FROT77

(b) Vertical coordinates in case FROT78

Figure 76: Results of cases FROT77 and FROT78

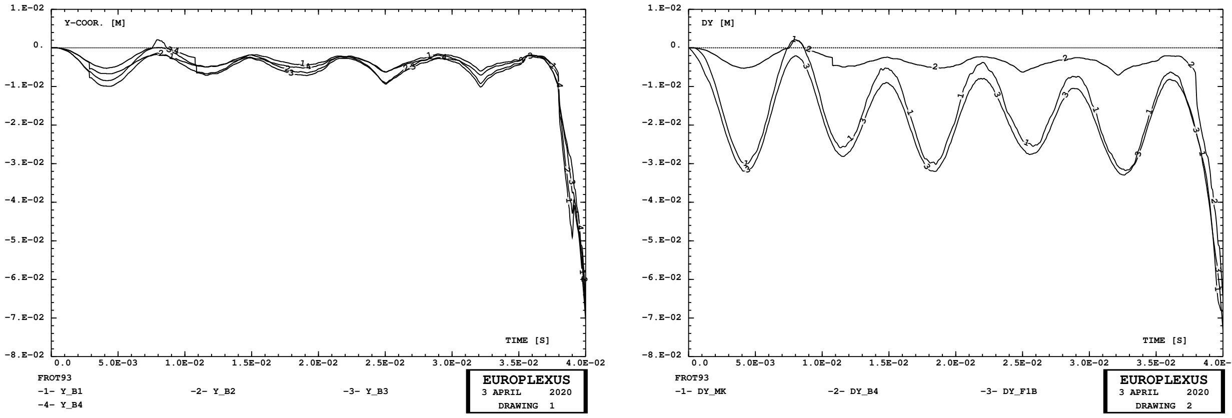
Case	Mesh	Description	Final time [ms]	Steps	CPU [s]
FROT93	696 CUB8 1 PMAT	No friction, `MASL`	40	2000	38.2
FROT94	696 CUB6 1 PMAT	With friction, `MASL`	40	2000	38.2
FROT95	696 CUB6 1 PMAT	With friction, `REBC`, `MASL`	(26.3)	(1315)	(25.0)
FROT96	696 CUB6 1 PMAT	With friction, `REB2`, `MASL`	(36.0)	(1802)	(34.7)
FROT97	696 CUB6 1 PMAT	Idem 96, different `DIAM`	40	2000	39.2

Table 8: Calculations for the sliding disk problem in 3D using `GPIN` in master/slave mode.

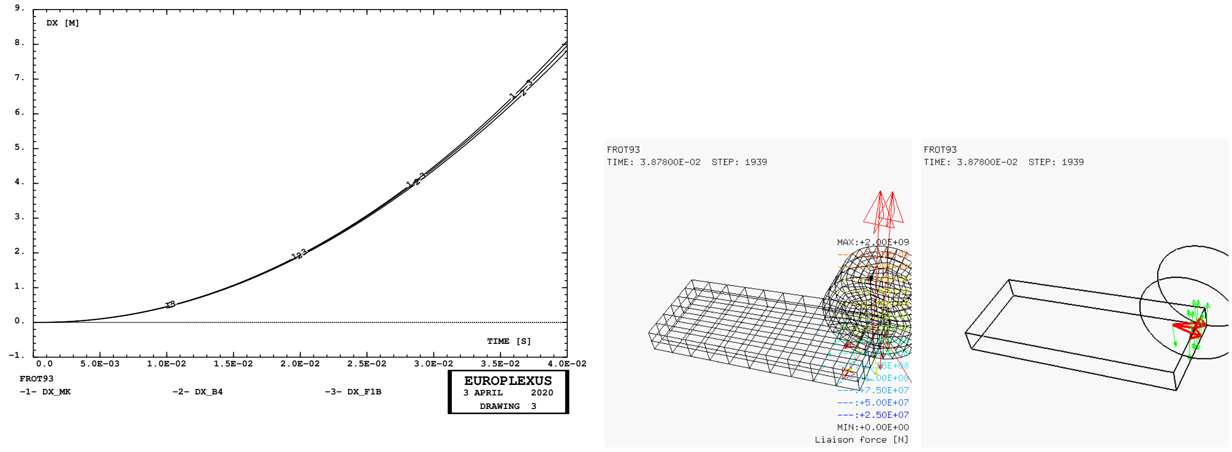
4.12.1 Case FROT93

This test is a repetition of case FROT71 (3D, no friction, `REB1`) but we add the `MASL 2 1` sub-directive of `GPIN` in order to activate the master-slave paradigm. With these settings, the plane behaves as a master and the disk behaves as a slave.

Some results are presented in Figure 77. The solution, shown in Figure 77, is only marginally better than that of case FROT71, cfr. Figure 61.



(a) Vertical coordinates (b) Vertical displacements



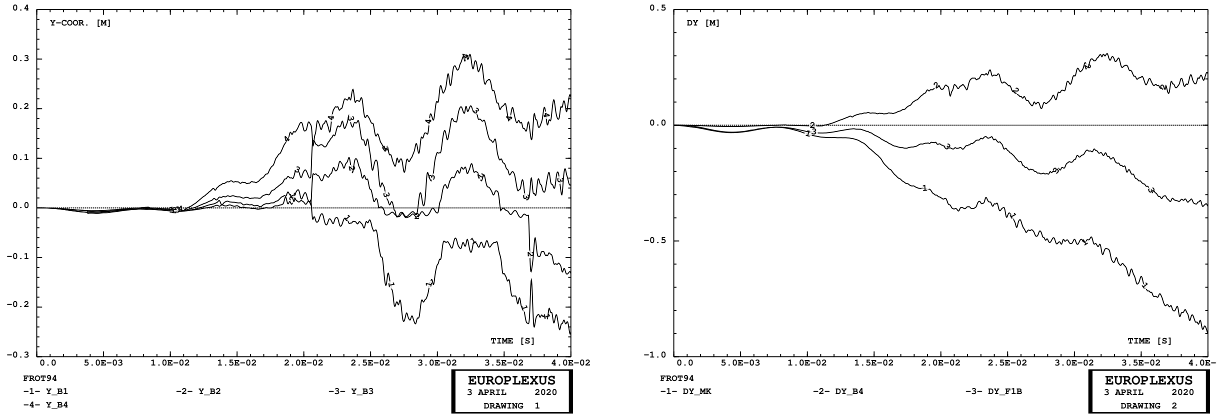
(c) Horizontal displacements (d) Link forces at step 1939 (e) Link joints at step 1939

Figure 77: Results of case FROT93

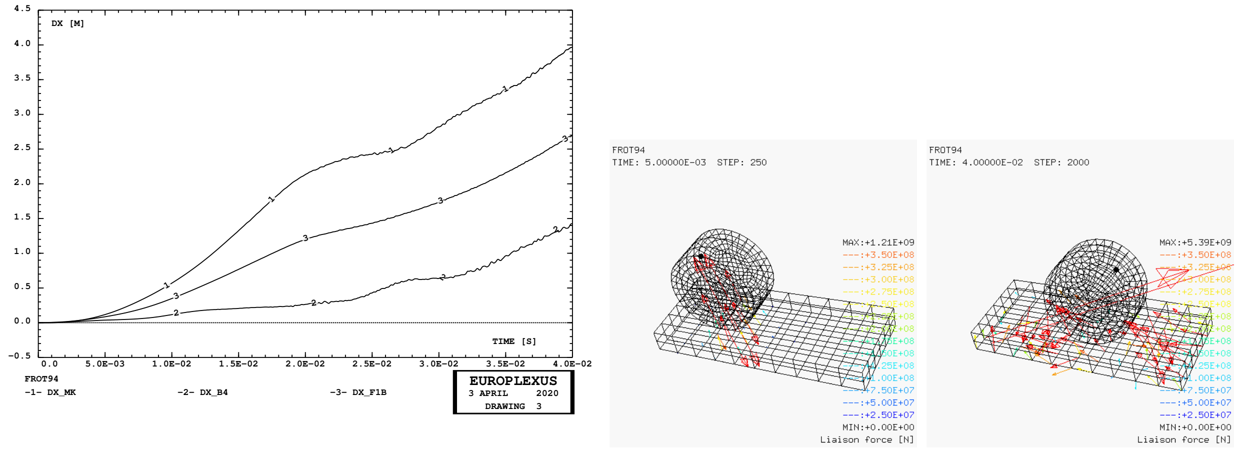
Some important oscillations persist as the disk approaches and finally leaves the end of the plane, like if the disk would somewhat stick to the right edge of the plane. For example, at step 1939 the link forces look a bit weird and have important horizontal components, as shown in Figure 77(d).

4.12.2 Case FROT94

This test is a repetition of case FROT72 (3D, friction, `REB1`) but we add the `MASL 2 1` sub-directive of `GPIN` in order to activate the master-slave paradigm. The results are very bad and not better than those of case FROT72, as shown in Figure 78.



(a) Vertical coordinates (b) Vertical displacements



(c) Horizontal displacements (d) $t = 5$ ms (e) $t = 40$ ms

Figure 78: Results of case FROT94

4.12.3 Cases FROT95 and FROT96

These tests are a repetitions of cases FROT74 (3D, friction, `REB1 REBC`) and FROT75 (3D, friction, `REB2`), respectively, but we add the `MASL 2 1` sub-directive. The first calculation crashes (élément croisé) at 26.3 ms and the second calculation crashes at 34.7 ms. The solutions until then are very bad and are not shown for brevity.

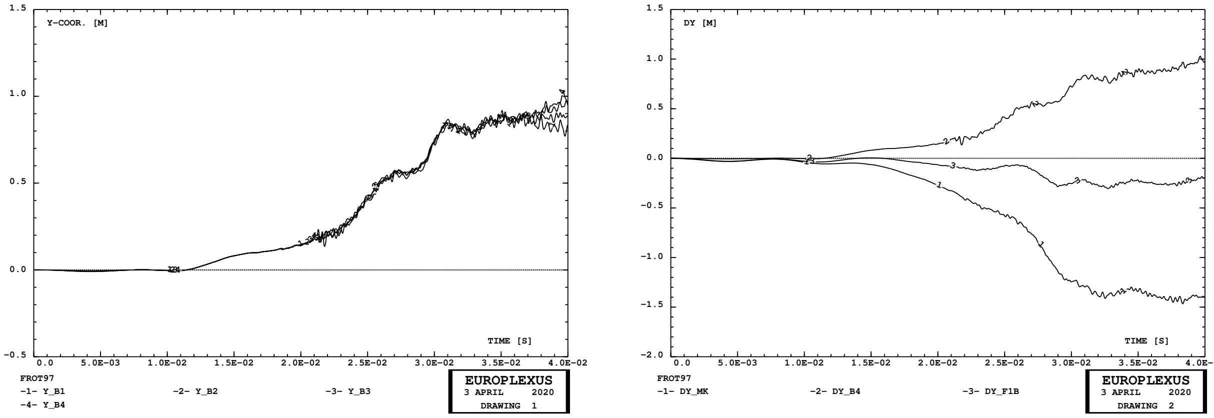
4.12.4 Cases FROT97

This test is a repetition of case FROT76 (3D, friction, `REB2`, GPIN diameter 0.2 in the disk and 0.4 in the plane) but we add the `MASL 2 1` sub-directive.

A bit surprisingly, the solution is much worse than that of case FROT76 (which was not too bad), see Figures 79 and 80.

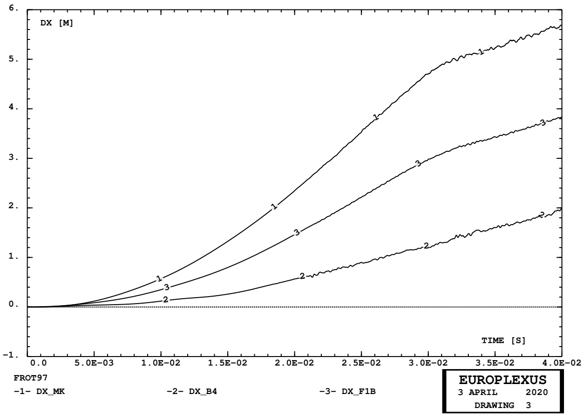
4.13 Solutions with EPX using `GPIN` and redundant constraints elimination

To be completed ...



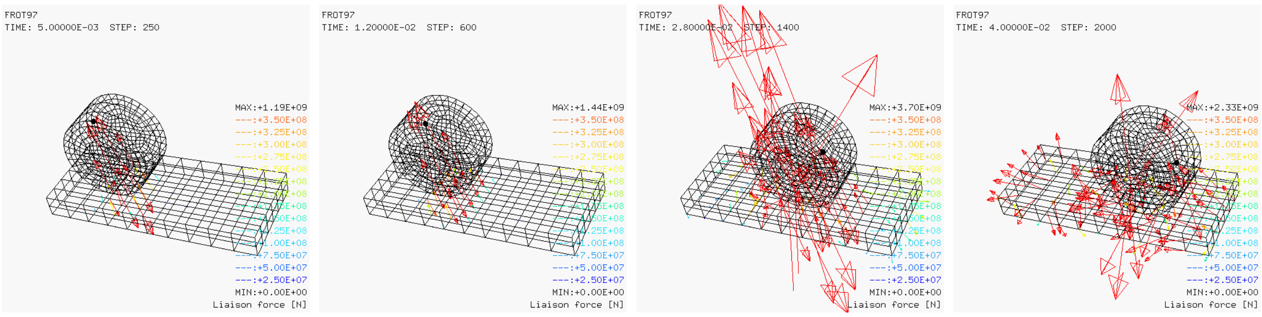
(a) Vertical coordinates

(b) Vertical displacements



(c) Horizontal displacements

Figure 79: Results of case FROT97



(a) $t = 5$ ms

(b) $t = 12$ ms

(c) $t = 28$ ms

(d) $t = 40$ ms

Figure 80: Further results of case FROT97

4.14 Further strategies to eliminate interlocking

We consider some further strategies to eliminate or reduce the risk of interlocking (over-constraining), especially in 3D applications.

In 2D, the possible GPIN types are P-GPINs and L-GPINs. The latter are always face GPINs and are associated either with faces of continuum elements or with bar/beam/shell elements. All these GPINs must be always created and the possible penetrations to be checked are P-P and P-L penetrations.

In 3D, the possible types of GPIN are P-GPINs, T-GPINs, Q-GPINs and L-GPINs. The latter are either *face* L-GPINs, associated with bars and beams, or *corner* GPINs (also denoted C-GPINs), associated with corners of continuum or plate/shell elements. The possible penetrations to be checked are P-P, P-T, P-Q, P-L and L-L penetrations.

It seems clear that in 3D the P, T, Q-GPINs and the face L-GPINs must always be created. However, as concerns the corner L-GPINs (i.e. the C-GPINs), one might perhaps avoid or limit their creation, in an attempt to reduce the number of possible inter-penetrations and thus alleviate the problem of over-constraining. To this end, new optional keywords are introduced in the GPIN input syntax:

```
"GPIN" $[ "PENA" <"SFAC" sfac> ]$
           ( $[ "BODY" ; "SELF" ]$
               < $[ "NOCG" ; "SHCG" < "ANGL" angl > < "ABS" > ]$ >
               < "FROT" "MUST" must "MUDY" mudy "GAMM" gamm >
             /LECT/ )
           ( "DIAM" diam /LECT/ )
        < "MASL" ("PAIR" m s) >
        < "EXCL" ("PAIR" n1 n2) >
```

The meaning of the new keywords, which are borrowed from the visualization of sharp corners in the EPX embedded OpenGL graphical module, is as follows:

`NOCG` Do not create corner GPINs (C-GPINs) for this body. This has only effect in 3D. By default, C-GPINs are cerated in 3D for all corners (both sharp and not sharp) of continuum bodies and for all corners (both sharp and not sharp) and all free edges of plate/shell bodies.

`SHCG` Create corner GPINs (C-GPINs) only at *sharp* corners and at free edges for this body. For the definition of sharp corners see the description of the `ANGL` keyword below. This has only effect in 3D.

`ANGL` Sets the minimum angle α_0 (between two 3D faces with a common side) beyond which the side is considered to be a sharp corner. By default, this angle is 60 degrees. Let n_1 and n_2 be unit normals to the two faces. Then the scalar product $n_1 \cdot n_2 = \cos\alpha$ is equal to the cosine of α, the angle between the normals (which is also the angle between the faces). Thus the corner is sharp if $\cos\alpha < \cos 60°$, i.e. when $\alpha < 60°$.

`ABS` Consider the absolute value of the above scalar product instead of the signed value. This has the following effect: when two faces have a common side and opposite (or nearly opposite) normals, the side is *not* considered sharp (while by default it would be). This option may be useful in the presence of complex 3D shell structures, because it is not always easy (and sometimes even impossible) to orient them consistently. With this option many “spurious” sharp corners disappear. Thus with this option the rule becomes: the corner is sharp when $|\alpha| < 60°$.

In order to test the C-GPIN limitation strategy described above, we perform some simulations, as listed in Table 9.

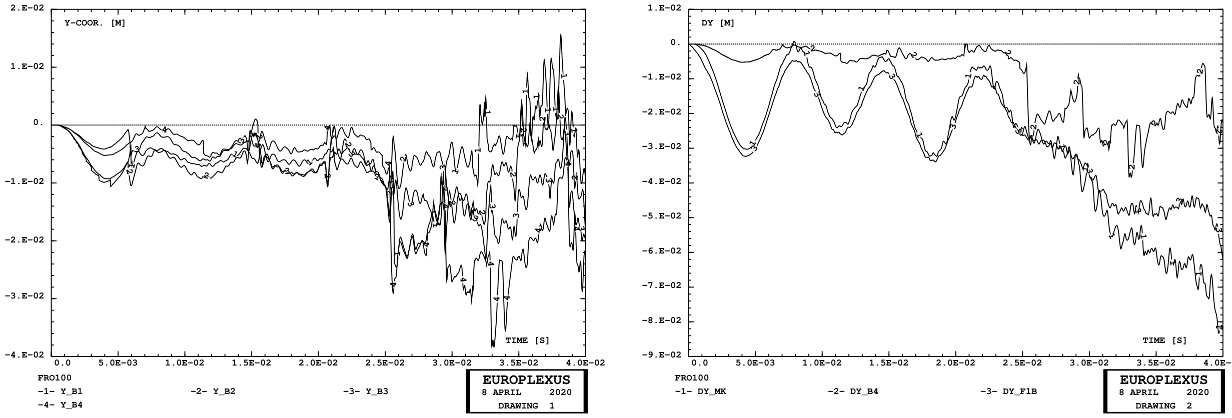
Case	Mesh	Description	Final time [ms]	Steps	CPU [s]
FRO100	696 CUB8 1 PMAT	No friction, `NOCG`	40	2000	28.9
FRO101	696 CUB6 1 PMAT	With friction, `NOCG`	40	2000	28.8
FRO102	696 CUB8 1 PMAT	No friction, `NOCG`, `MASL`	40	2000	35.7
FRO103	696 CUB6 1 PMAT	With friction, `NOCG`, `MASL`	40	2000	28.3

Table 9: Calculations for the sliding disk problem in 3D using `GPIN` with C-GPIN limitation.

4.14.1 Case FRO100

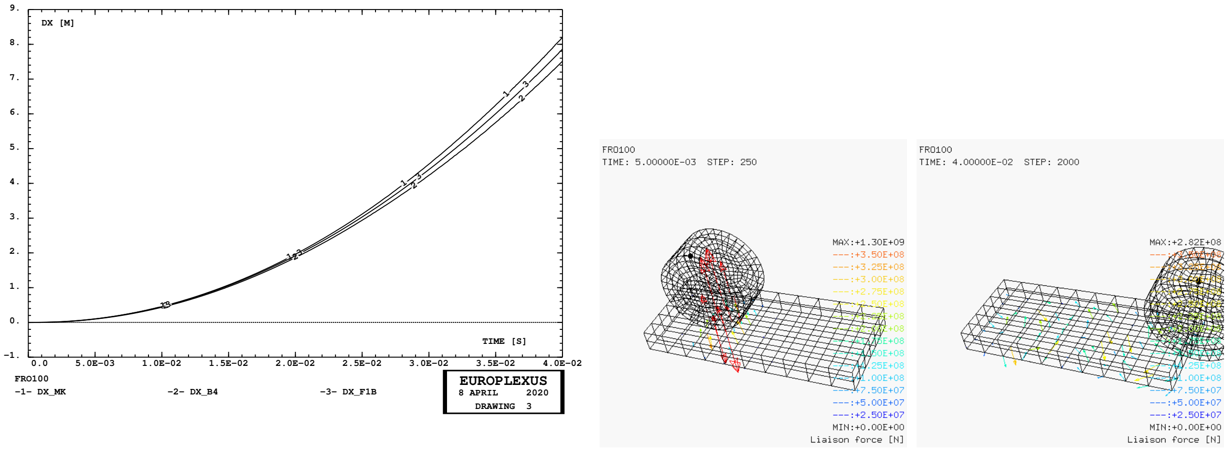
This test is a repetition of case FROT71 (3D, no friction) by adding the `NOCG` keyword to both bodies. As a consequence, no C-GPINs are built for this test. The only possible penetrations are P-P and P-Q, but no L-L (i.e. C-C) penetrations will take place. The situation is therefore similar to what would happen by using the `GLIS` model, but with the difference that in the case of `GLIS` one body would behave as a master and the other as a slave while here both bodies act both as master and as slave (reciprocal penetration).

Some results of this simulation are presented in Figure 81. The solution is relatively smooth until about 20 ms but then some oscillations appear, although less than in case FROT71.



(a) Vertical coordinates

(b) Vertical displacements



(c) Horizontal displacements

(d) $t = 5$ ms

(e) $t = 40$ ms

Figure 81: Results of case FRO100

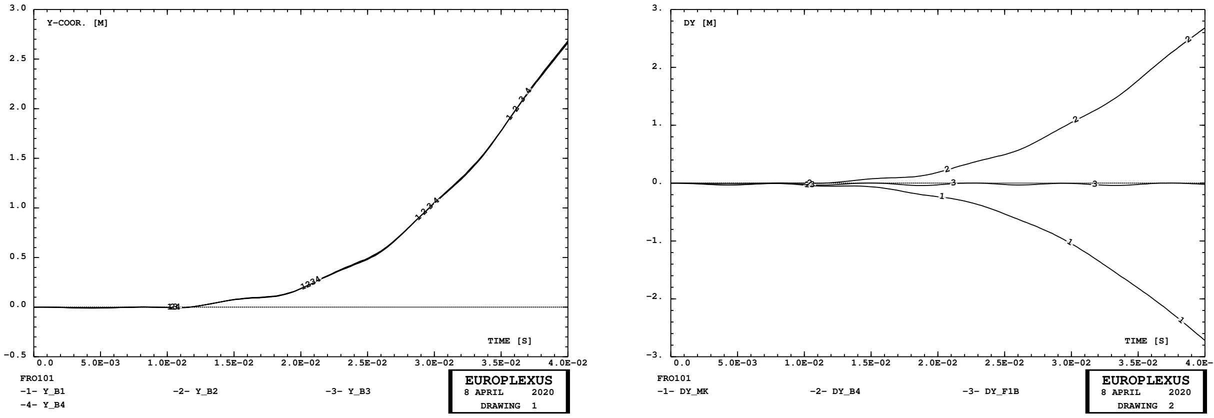
4.14.2 Case FRO101

This test is a repetition of case FROT72 (3D, with friction) by adding the `NOCG` keyword to both bodies. Some results of this simulation are presented in Figure 82. The solution is relatively smooth until the end and much better than case FROT72.

4.14.3 Case FRO102

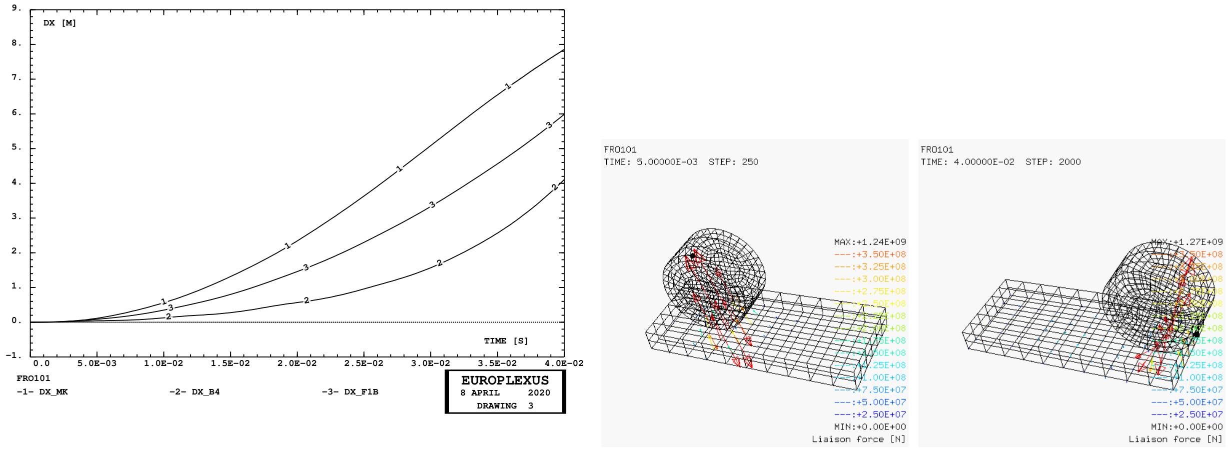
This test is a repetition of case FRO100 (3D, no friction, `NOCG`) by adding also the `MASL PAIR 2 1` sub-directive which activates master/slave behaviour (here with the plane as master and the disk as slave). In this way the `GPIN` contact model becomes very close to the `GLIS` model.

Some results of this simulation are presented in Figure 83. The solution is relatively smooth until the end and much better than case FRO100.



(a) Vertical coordinates

(b) Vertical displacements



(c) Horizontal displacements

(d) $t = 5$ ms

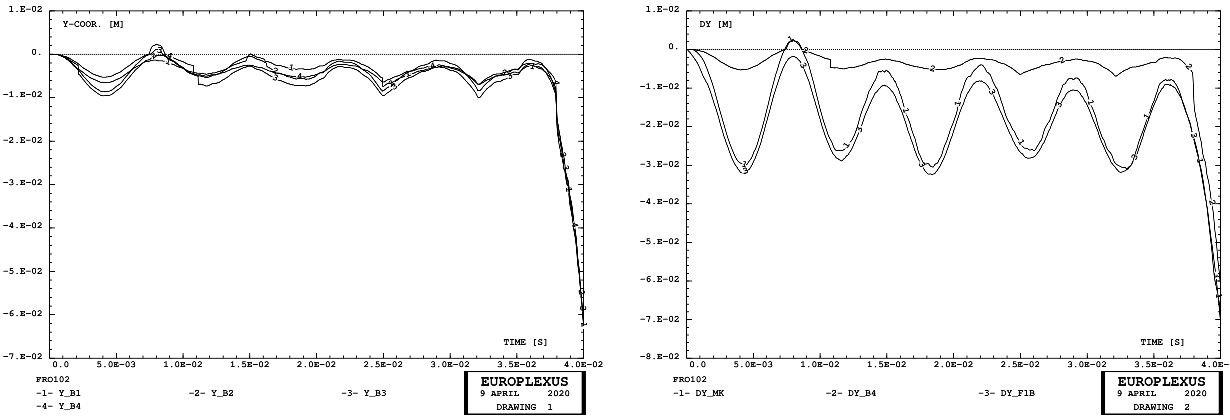
(e) $t = 40$ ms

Figure 82: Results of case FRO101

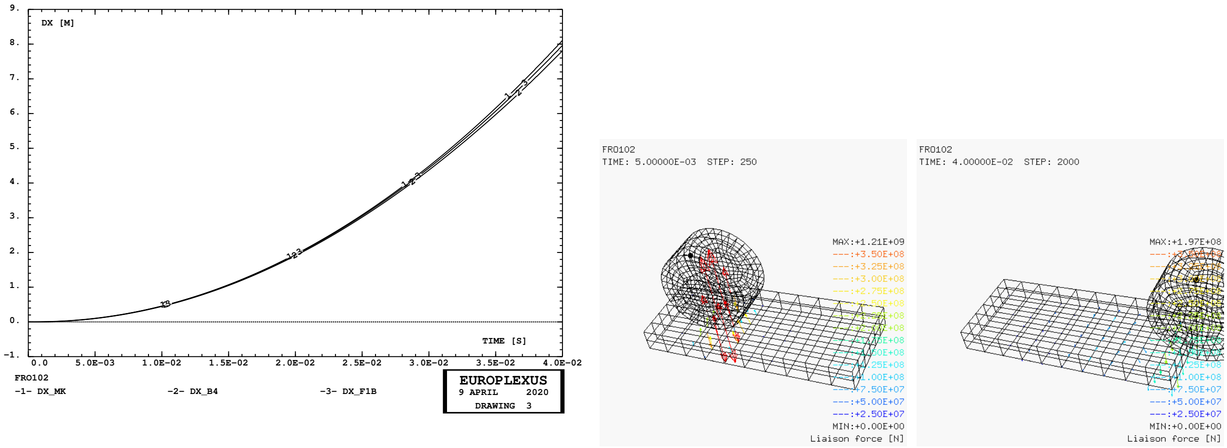
4.14.4 Case FRO103

This test is a repetition of case FRO101 (3D, with friction, `NOCG`) by adding also the `MASL PAIR 2 1` sub-directive which activates master/slave behaviour (here with the plane as master and the disk as slave). In this way the `GPIN` contact model becomes very close to the `GLIS` model.

Some results of this simulation are presented in Figure 84. The solution is relatively smooth until the end and similar to case FRO101.

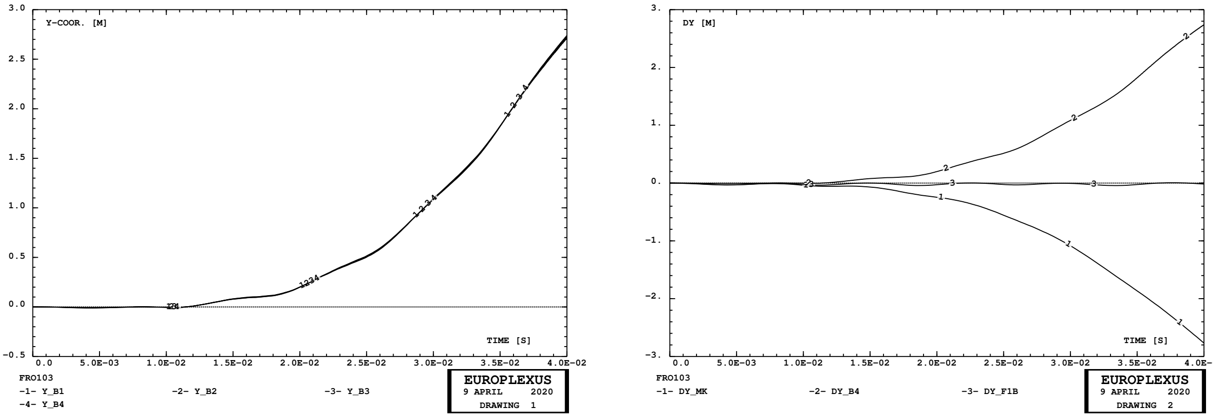


(a) Vertical coordinates (b) Vertical displacements

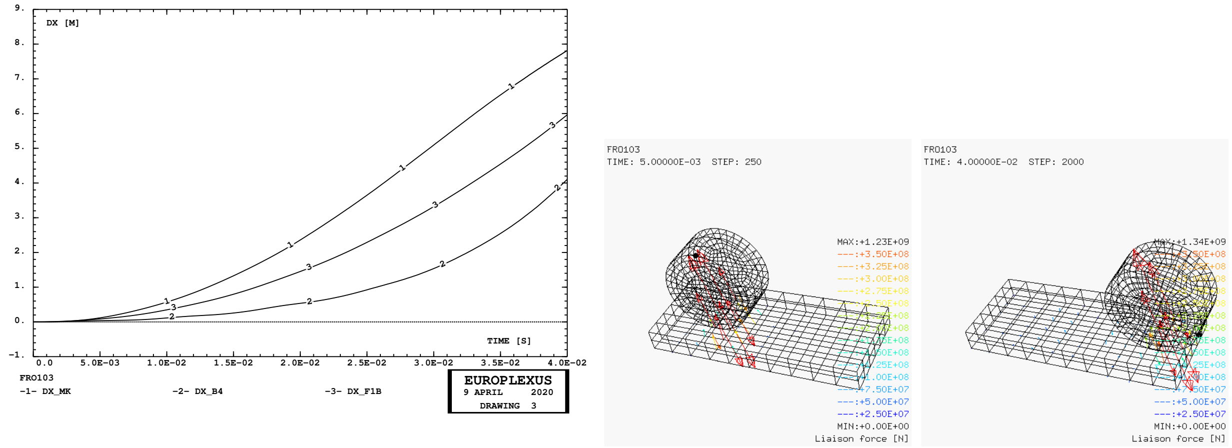


(c) Horizontal displacements (d) $t = 5$ ms (e) $t = 40$ ms

Figure 83: Results of case FRO102



(a) Vertical coordinates (b) Vertical displacements



(c) Horizontal displacements (d) $t = 5$ ms (e) $t = 40$ ms

Figure 84: Results of case FRO103

5 Numerical examples using penalty-based contact models

We perform a set of basic numerical examples in order to check the implementation of friction in the penalty-based contact models.

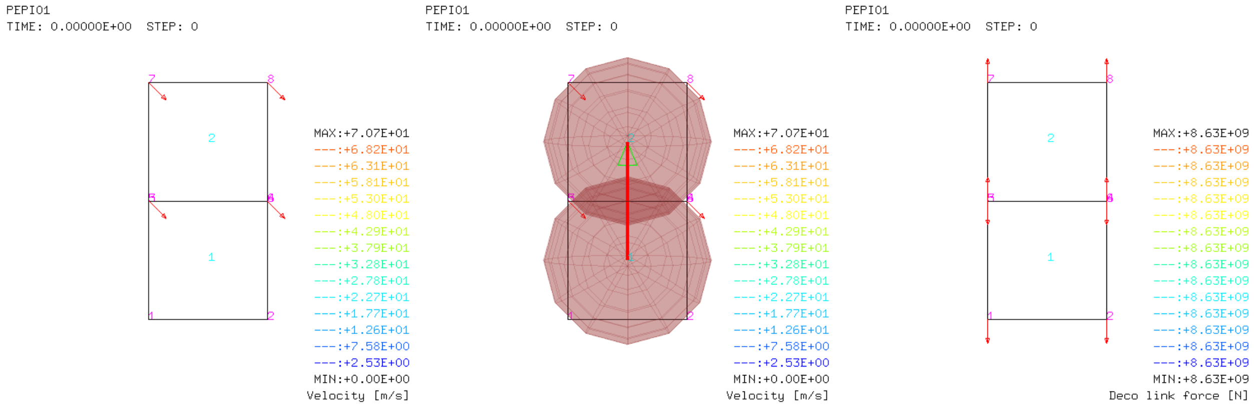
5.1 Solutions with EPX using `PINB`

We start by considering the pinball contact model `PINB`. The solutions are carried out for 2000 time steps, until $t_f = 40$ ms.

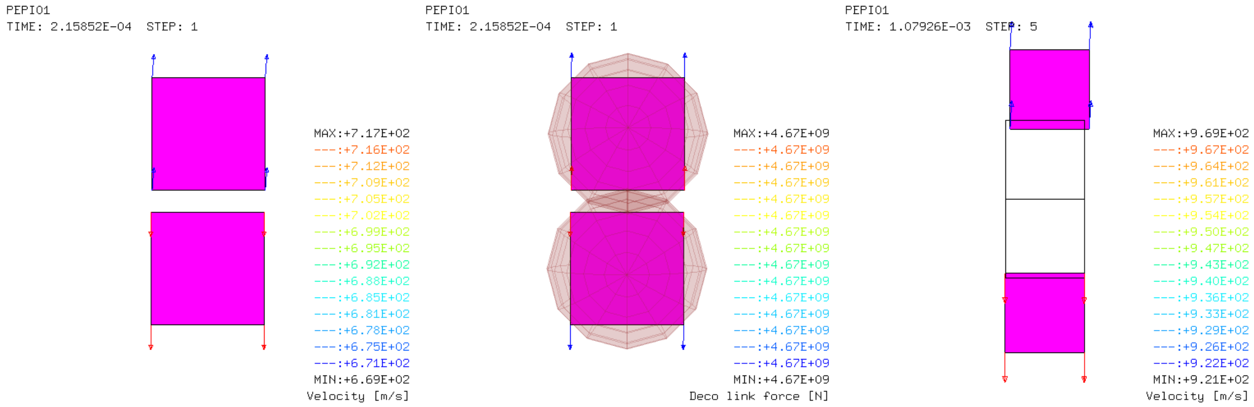
5.1.1 Case PEPI01

This test considers two quadrilateral elements in 2D impacting each other, see Figure 85(a). The first element is initially stationary while the second one has an initial velocity directed towards the first element, with both a normal and a tangential component. The two elements are initially adjacent so that interpenetration of the corresponding pinballs occurs already in the initial configuration.

This example considers no friction and is used as a reference solution for the subsequent case with friction. Some results are presented in Figure 85. Note that the contact forces are directed along the normal (vertical in this case) since there is no friction. The impacted body assumes therefore a purely vertical velocity.



(a) Initial velocities (b) Initial contact (c) Initial contact forces



(d) Step 1 velocities (e) Step 1 contact forces (f) Step 5 velocities and initial configuration

Figure 85: Test case PEPI01

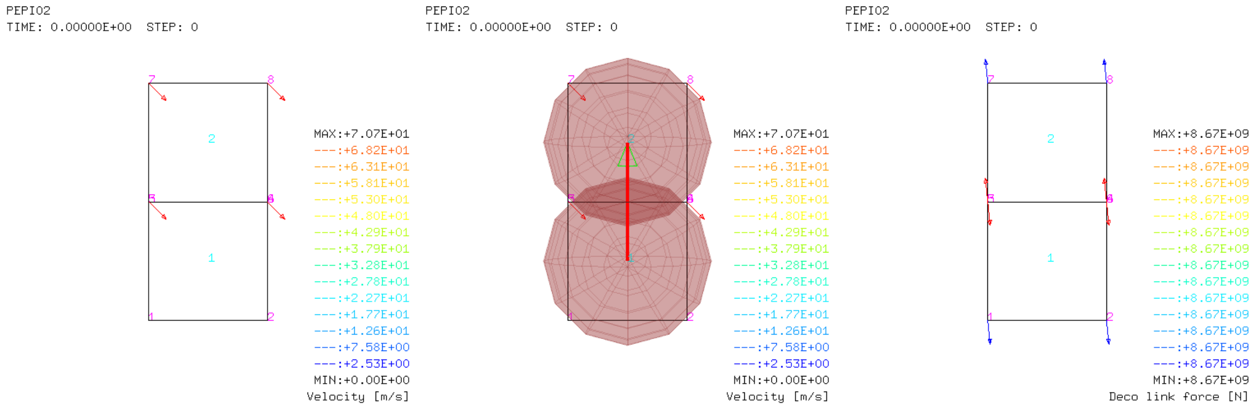
5.1.2 Case PEPI02

This test is a repetition of the previous test `PEPI01` by activating friction. The declaration of pinball contacts in the EPX input file becomes:

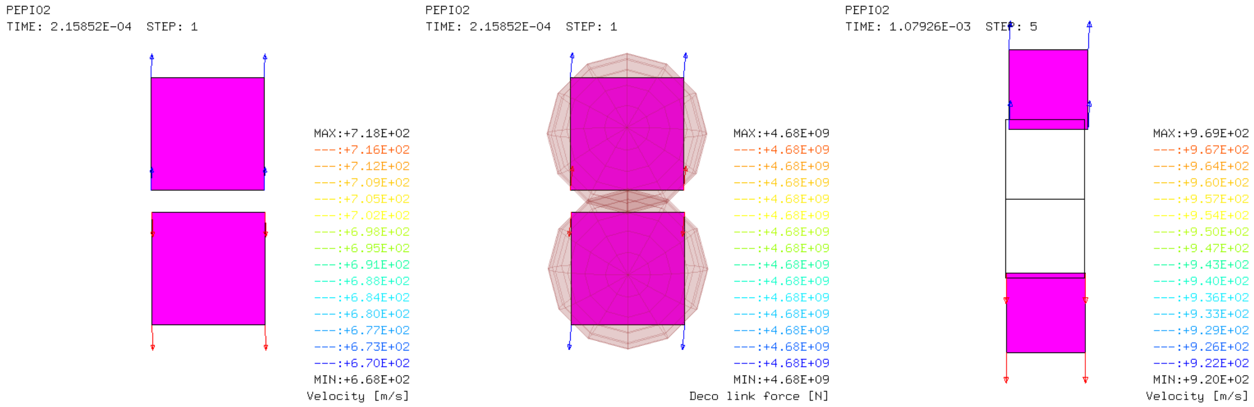
```
LINK DECO
     PINB PENA SFAC 1.0
          BODY FROT MUST 0.3 MUDY 0.1 GAMM 2.3025
          LECT 1 TERM
          BODY FROT MUST 0.3 MUDY 0.1 GAMM 2.3025
          LECT 2 TERM
```

where the FROT sub-directive has been added.

Results of this test case are presented in Figure 86. Small tangential (horizontal) components of the contact forces arise at step 0 due to the friction. Consequently, the impacted body assumes a (small) horizontal velocity component.



(a) Initial velocities (b) Initial contact (c) Initial contact forces



(d) Step 1 velocities (e) Step 1 contact forces (f) Step 5 velocities and initial configuration

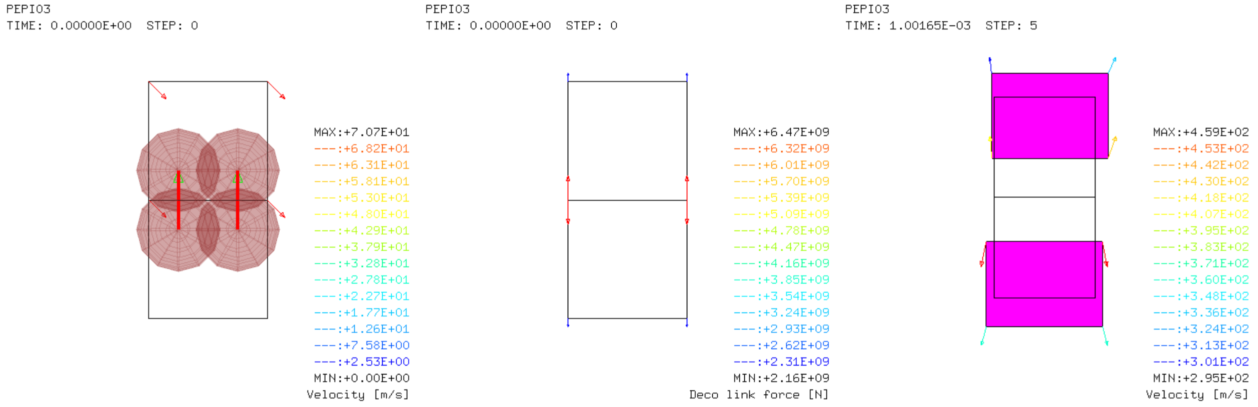
Figure 86: Test case PEPI02

5.1.3 Cases PEPI03 and PEPI04

These tests are repetitions of the previous tests `PEPI01` and `PEPI02`, respectively, by using hierarchic pinballs at level 1 (`MLEV 1`). In case `PEPI03`, for example, the declaration of pinball contacts in the EPX input file becomes:

```
LINK DECO
     PINB PENA SFAC 1.0
          BODY MLEV 1 LECT 1 TERM
          BODY MLEV 1 LECT 2 TERM
```

Results of test `PEPI03` (without friction) are presented in Figure 87. The contact forces are directed along the normal, like in case `PEPI01`, but are no longer equally distributed on the nodes of the impacting elements. Contact forces on the nodes belonging to the impacting element sides are

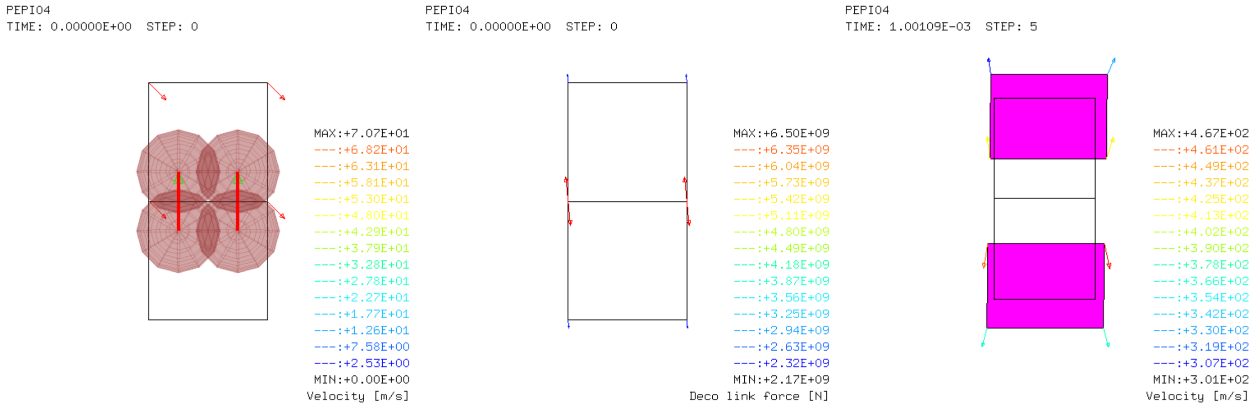


(a) Initial velocities (b) Initial contact forces (c) Step 5 velocities and initial configuration

Figure 87: Test case PEPI03

larger than those on the opposite sides (as one would expect). Consequently, the elements deform under the contact forces, as shown in the last picture (while this was not the case in test `PEP001`).

Results of test `PEPI04` (with friction) are presented in Figure 88. Small tangential (horizontal) components of the contact forces arise at step 0 due to the friction. Consequently, the impacted body assumes a (small) horizontal velocity component.



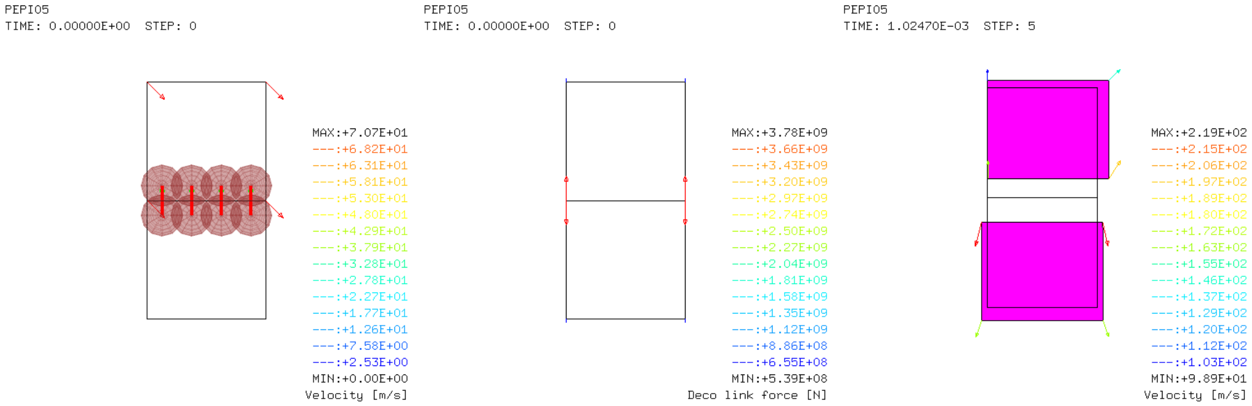
(a) Initial velocities (b) Initial contact forces (c) Step 5 velocities and initial configuration

Figure 88: Test case PEPI04

5.1.4 Cases PEPI05 and PEPI06

These tests are repetitions of the previous tests `PEPI01` and `PEPI02`, respectively, by using hierarchic pinballs at level 2 (`MLEV 2`).

Results of test `PEPI05` (without friction) are presented in Figure 89. Results of test `PEPI06` (with friction) are presented in Figure 90.

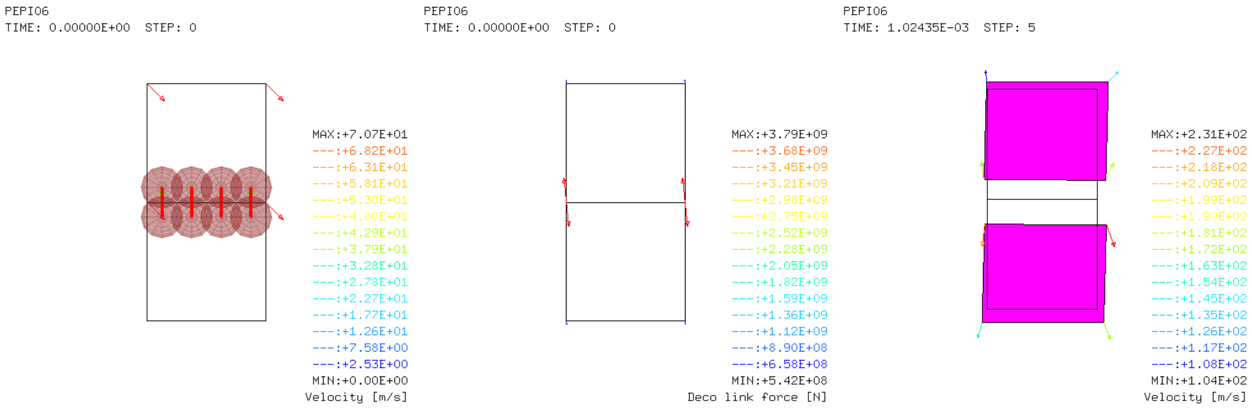


(a) Initial velocities

(b) Initial contact forces

(c) Step 5 velocities and initial configuration

Figure 89: Test case PEPI05



(a) Initial velocities

(b) Initial contact forces

(c) Step 5 velocities and initial configuration

Figure 90: Test case PEPI06

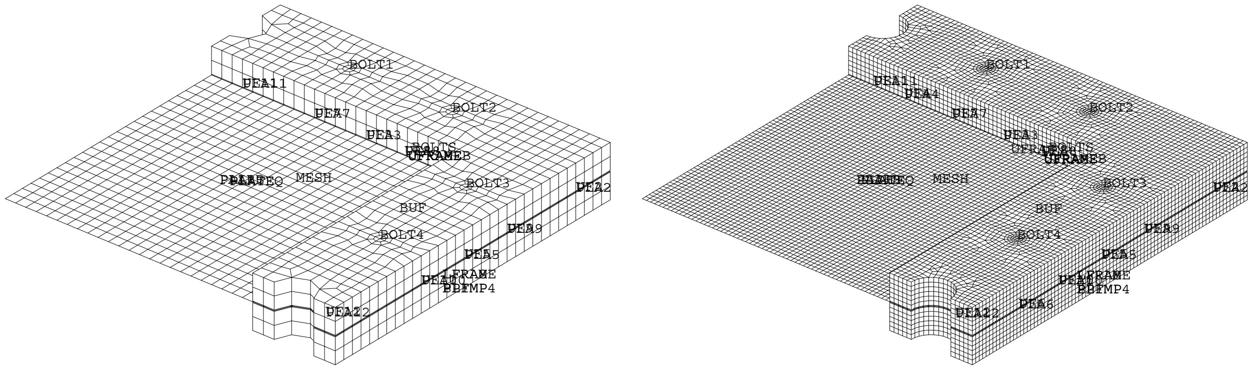
6 Application - Blast-loaded clamped plate

We now consider a more realistic application. A thin square plate is clamped in a frame and loaded by a blast pressure. Experimental results of this problem are available from NTNU Trondheim. The test has been already simulated by EPX using the `GLIS` contact model, see reference [20]. Further verifications of the `GLIS` technique have been shown in [33].

Here the problem is first solved by the `GLIS` model with various discretizations in order to obtain a reference solution. Then, solutions with pinballs are attempted.

6.1 Reference solutions with GLIS

Two solutions are obtained with `GLIS`, the first one with a coarse (base) mesh and adaptivity, see Figure 91(a), the second one with a much finer constant mesh and no adaptivity, see Figure 91(b). In both cases, only one fourth of the experimental setup is modelled thanks to symmetries.



(a) Coarse mesh

(b) Fine mesh

Figure 91: Meshes for the clamped plate problem

6.1.1 Case OSTO03

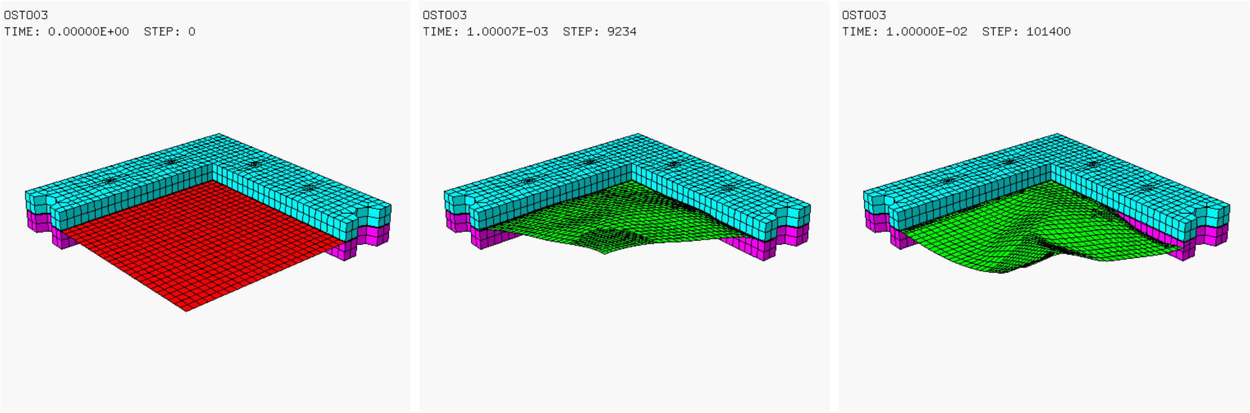
This simulation took 101 400 time steps and 6071 s (1.7 hours) of CPU time to reach the chosen final time of 10 ms. Adaptive mesh refinement in the plate up to a level of 3 (`MAXL 3`) is specified in the plate region, following a threshold-based criterion:

```
ADAP THRS ECRO 11 TMIN 0.05 TMAX 0.25 MAXL 3
          LECT pinade TERM
```

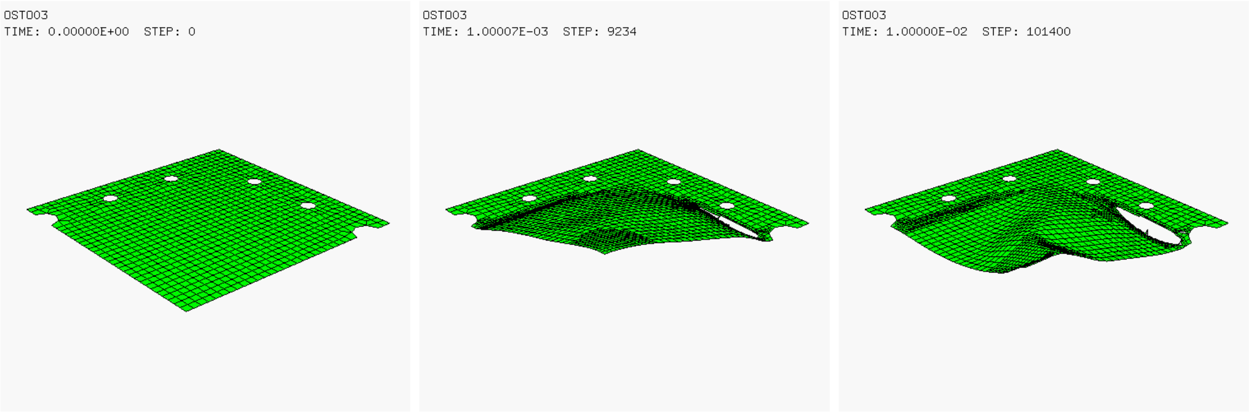
Contact is specified to occur along two sliding surfaces (`GLIS` method). The first one uses the lower frame elements as master surfaces and the nodes of the plate as slaves. The second one uses the upper frame and bolts elements as master surfaces and the plate nodes as slaves. In both cases a gap of 0.4 mm is specified and friction is applied. The slave plate nodes are all nodes of the plate (both in the exposed and in the clamped area of the plate), but excluding the nodes located along the two symmetry planes.

```
LINK COUP SPLT NONE SOLV PARD
     BLOQ 123 LECT bloc TERM
     BLOQ 1 LECT symx TERM
     BLOQ 2 LECT symy TERM
     BLOQ 56 LECT symxP TERM
     BLOQ 46 LECT symyP TERM
     GLIS 2 FROT MUST 0.1 MUDY 0.1 GAMM 0 ! Contact surface #1
            PGAP 0.4E-3
            MAIT LECT lframe TERM
            PESC LECT plateN  TERM
            FROT MUST 0.1 MUDY 0.1 GAMM 0 ! Contact surface #2
            PGAP 0.4E-3
            MAIT LECT uframeb TERM
            PESC LECT plateN   TERM
```

Some results are shown in Figure 92. Some mesh refinement occurs at the plate centre and near the perimeter of the exposed plate area (close to the clampings). Some tearing (element erosion) takes place, but the solution is not perfectly symmetric.



(a) $t = 0$ (b) $t = 1$ ms (c) $t = 10$ ms



(d) $t = 0$ (e) $t = 1$ ms (f) $t = 10$ ms

Figure 92: Some results of test case OSTO03 (top row: assembled model, bottom row: plate only)

6.1.2 Case OSTO04

This simulation took 129 673 time steps and 81 125 s (22.6 hours) of CPU time to reach the chosen final time of 10 ms. The mesh is constant and no adaptive refinement is applied. Element erosion in the plate is activated, but does not actually take place in the simulation.

Some results are shown in Figure 93. No tearing (element erosion) takes place, and the solution looks symmetric, as one would expect.

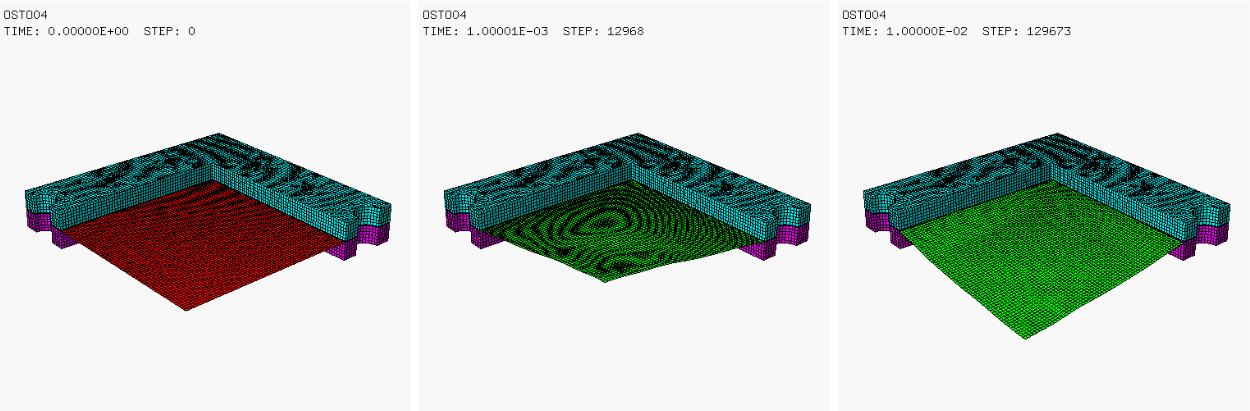
6.2 Solutions with PINB

We now consider solutions with the pinball contact method.

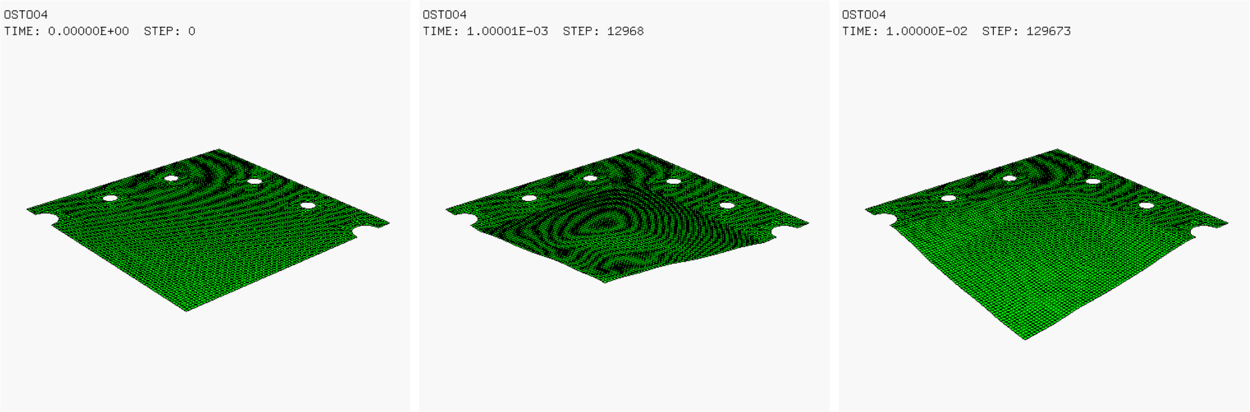
6.2.1 Case OSTO05

This case is similar to test `OSTO03` presented above (coarse base mesh and adaptivity) but an attempt is made to replace the `GLIS` contact model with the pinball (`PINB`) model. A penalty-based (decoupled) pinball method is tentatively adopted. The boundary conditions read:

```
OPTI PINS ASN
LINK COUP SPLT NONE SOLV PARD
     BLOQ 123 LECT bloc TERM
     BLOQ 1 LECT symx TERM
```



(a) $t = 0$ (b) $t = 1$ ms (c) $t = 10$ ms



(d) $t = 0$ (e) $t = 1$ ms (f) $t = 10$ ms

Figure 93: Some results of test case OSTO04 (top row: assembled model, bottom row: plate only)

```
     BLOQ 2 LECT symy TERM
     BLOQ 56 LECT symxP TERM
     BLOQ 46 LECT symyP TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY FROT MUST 0.1 MUDY 0.1 GAMM 0 MLEV 3
          LECT lframe TERM
          BODY FROT MUST 0.1 MUDY 0.1 GAMM 0 MLEV 3
          LECT uframeb TERM
          BODY FROT MUST 0.1 MUDY 0.1 GAMM 0 DIAM 0.0008
          LECT nplate TERM
```

The Assembled Surface Normals method (`ASN`) is activated in an attempt to obtain more regular pinball contact normals than by the default method. Note that the option (`OPTI PINS ASN`) must be activated *before* specifying the pinballs, in order to be effective.

The same contacting surfaces are tentatively assumed as in the previous simulations with the `GLIS` model. That is, three pinball-endowed bodies are defined. The first two are the lower frame (`lframe`) and the upper frame with bolts (`uframeb`), respectively. The third body is the plate. However, placing pinballs directly in the plate (shell) elements would be un-practical in this case, since these elements are relatively big.

Thus a trick is used in order to associate pinballs with the plate *nodes* rather than with the plate elements. This is not possible directly in EPX at the moment since pinballs are always associated with elements (not with nodes). The trick consists in adding fake material point elements (`PMAT`) attached to the plate nodes and then using these elements as geometrical supports for the pinballs. In Cast3m we define an object `nplate` by the command

```
nplate = chan POI1 plate;
```

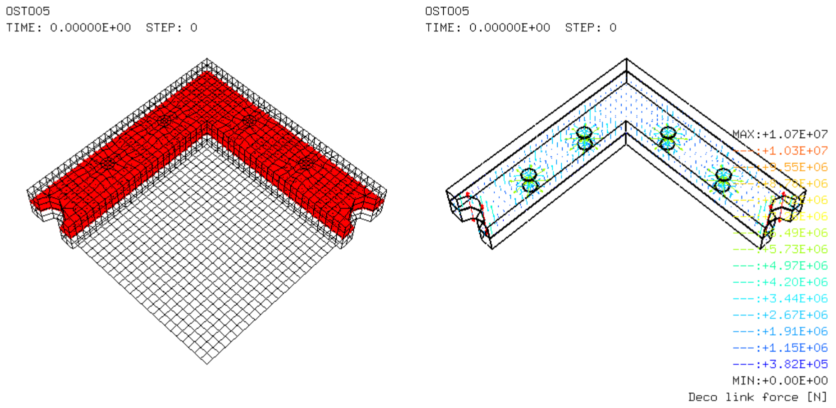
Next, in EPX we associate `PMAT` elements with the `nplate` object. A thickness (diameter) of 0.8 mm, equal to the physical thickness of the plate, is assigned to these material points by the `COMP EPAI` directive.. The points are associated with a mass (`MASS`) material having zero density, so that no mass is actually added to the structure. However, the Young's modulus and Poisson's coefficient are set equal to those of the plate material. These values are used by EPX in order to compute the bulk modulus, which is used in evaluating the penalty contact forces.

```
MATE . . . (skipped data)
     MASS 0.0 YOUN 70.0E9 NU 0.3
          LECT nplate TERM
```

Finally, pinballs with a diameter of of 0.8 mm, equal to the physical thickness of the plate, are associated to the material points, as already shown above. These pinballs will keep a constant diameter in time, in an attempt to represent the actual physical thickness of the plate.

The pinballs embedded in the clamping frame are hierarchically refined (`MLEV`) in an attempt to obtain a decent geometrical representation of the clamping setup. Keep in mind that an exact representation is impossible with spherical pinballs, so this is a tough problem for the pinball method, which is designed for impact problems, and not so much for (smooth) contact problems like the present one.

A refinement level of 2 (`MLEV 2`) was initially tentatively assumed. However, the resulting pinballs in the clampings are still too big. Many penetrations occur in the initial setup, as shown by the red segments in Figure 94(a). Very large (non-physical) penalty contact forces arise as shown in Figure 94(b) and huge velocities (≈ 3000 m/s) appear already at step 1, so that after some steps the solution becomes unstable.



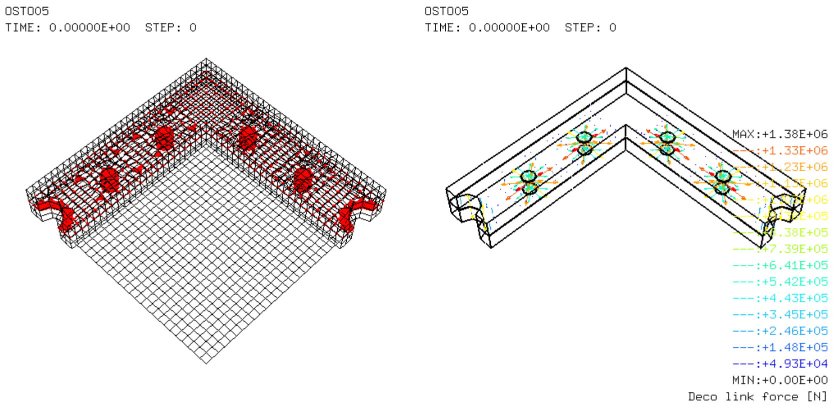
(a) Initial penetrations (b) Initial penalty forces

Figure 94: Initial contacts in case OSTO05 with `MLEV 2`

This is not too surprising, if one considers the radically different approach used by the penalty in comparison to the Lagrange Multipliers approach. In the LM method, if an initial penetration occurs but no relative velocities are present (like in this case), then no contact forces are generated because the penetration does not tend to increase. However, in the penalty method the pinballs act as springs, so that an initial penetration generates (possibly huge) contact (repulsive) forces, even in the absence of a relative velocity.

By increasing the refinement level to `MLEV 3`, thus obtaining twice smaller sub-pinballs in the clampings, many initial inter-penetrations (especially between the bolts and clamp, less between the clamps and the plate) still remain but a numerical solution can be obtained. The initial contacts and penalty forces are shown in Figure 95. The penalty forces are one order of magnitude smaller than with `MLEV 2`.

Some elements start failing already at the initial steps and complete tearing of the explosed plate area along the clamped perimeter gradually occurs. A macro fragment is separated. Also some internal bolt elements fail.

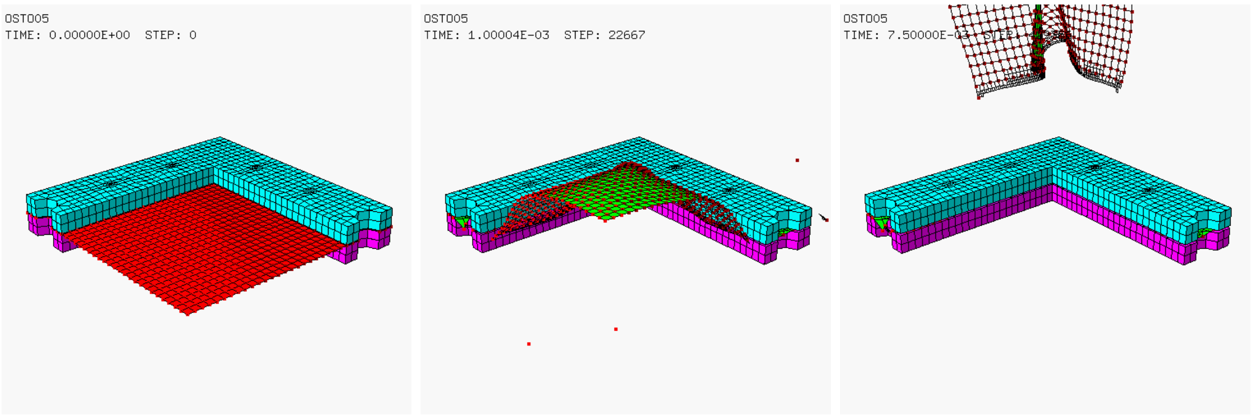


(a) Initial penetrations (b) Initial penalty forces

Figure 95: Initial contacts in case OSTO05 with `MLEV 3`

The calculation is intentionally stopped after 441 125 time steps, at $t = 7.6$ ms, having consumed 83 756 s (23.3 hours) of CPU time. The estimated CPU time for 10 ms would then be about 31 hours.

Some results are shown in Figure 96.



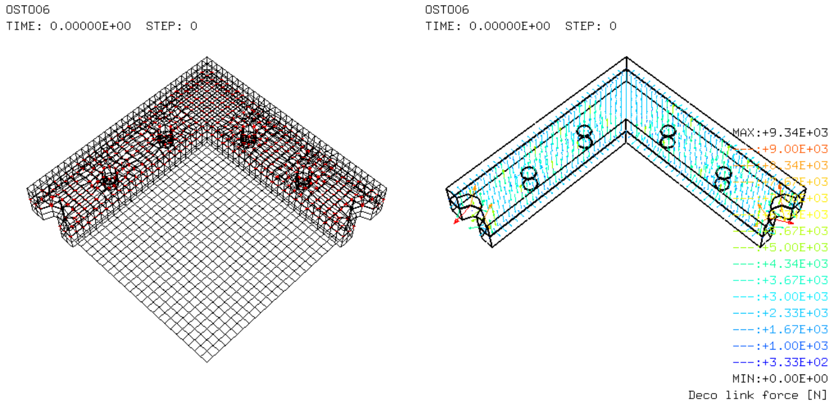
(a) $t = 0$ (b) $t = 1$ ms (c) $t = 7.6$ ms

(d) $t = 0$ (e) $t = 1$ ms (f) $t = 7.6$ ms

Figure 96: Some results of test case OSTO05 (top row: assembled model, bottom row: plate only)

6.2.2 Case OSTO06

This calculation is similar to `OSTO05` but uses a refinement `MLEV 4` in the clamping pinballs. No interpenetration occurs in the bolts at the initial time. The initial contacts and penalty forces are shown in Figure 97. The penalty forces are much more regular than in the previous case and two orders of magnitude smaller than with `MLEV 3`.



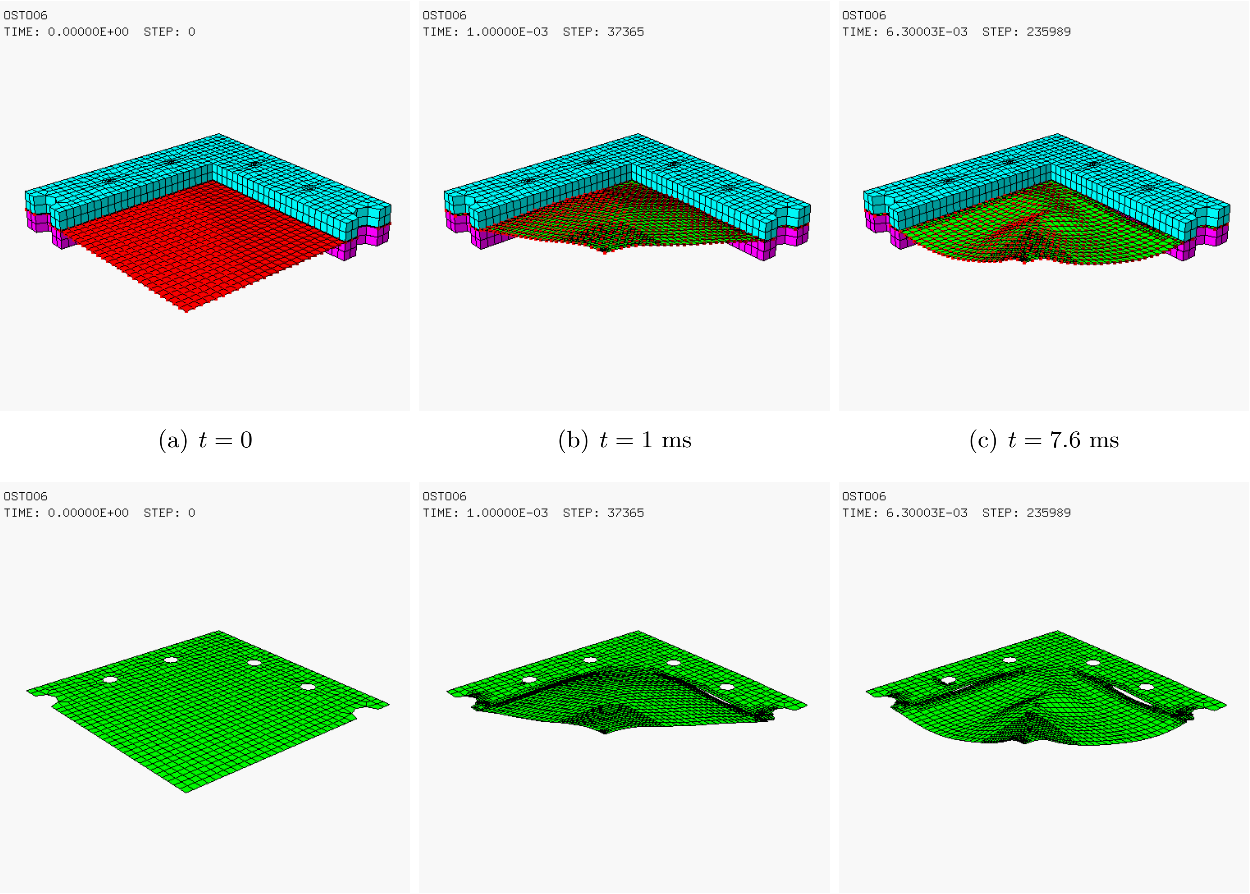
(a) Initial penetrations (b) Initial penalty forces

Figure 97: Initial contacts in case OSTO05 with `MLEV 3`

The calculation is intentionally stopped after 239 926 time steps, at $t = 6.4$ ms, having consumed 80 716 s (22.4 hours) of CPU time. The estimated CPU time for 10 ms would then be about 35 hours.

Some results are shown in Figure 98. The four solutions obtained so far are compared in Figure 99 in terms of the vertical displacement of the plate centre. The closest result to the real behaviour of the plate observed in the experiments is the one obtained with `GLIS` and the fine mesh (case `OSTO04`, cyan curve). The coarse mesh solution with `PINB` and `MLEV 3` (case `OSTO05`, green curve) is unrealistic, showing the importance of choosing the size of the pinballs. The other two solutions, obtained with the coarse mesh plus adaptivity and either `GLIS` (case `OSTO03`, black curve) or `PINB` with `MLEV 4` (case `OSTO06`, red curve) are quite similar (although different from the real behaviour).

This seems to indicate that the `PINB` hierarchic model with penalty may lead to results comparable to those obtained with `GLIS`, provided the size of the pinballs is properly chosen. However, a comparison of CPU times indicates that the `GLIS` model is much faster than (hierarchic) pinballs in these applications.



(a) $t = 0$ (b) $t = 1$ ms (c) $t = 7.6$ ms

(d) $t = 0$ (e) $t = 1$ ms (f) $t = 7.6$ ms

Figure 98: Some results of test case OSTO06 (top row: assembled model, bottom row: plate only)

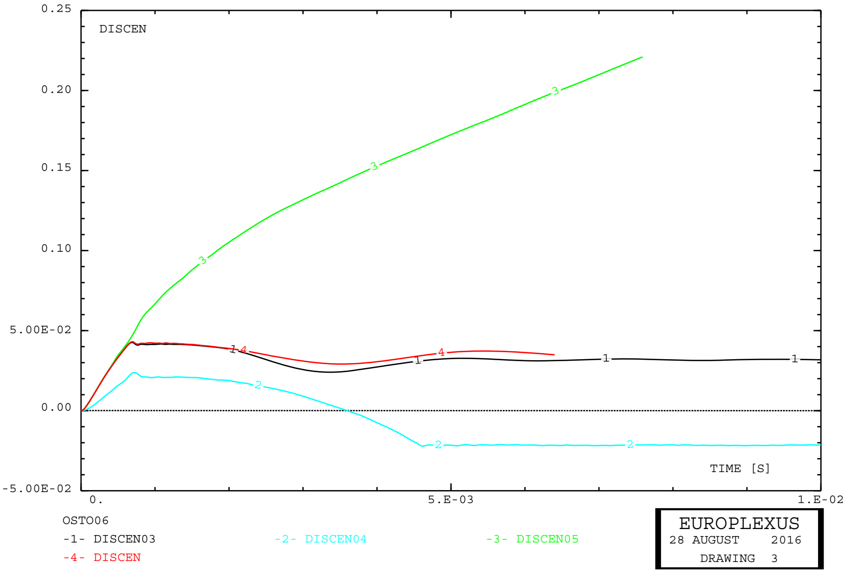


Figure 99: Comparison of plate center displacements in cases OSTO03 to OSTO06

7 Application - Shock-tube loaded plate

We now consider another application. A thin plate is mounted in a shock tube device and is loaded by a pressure wave. The exposed area of the plate has a square shape. Experimental results of this problem are available from NTNU Trondheim. Here the problem is first solved by the `GLIS` model in order to obtain a reference solution. Then, solutions with pinballs are obtained.

7.1 Reference solutions with GLIS

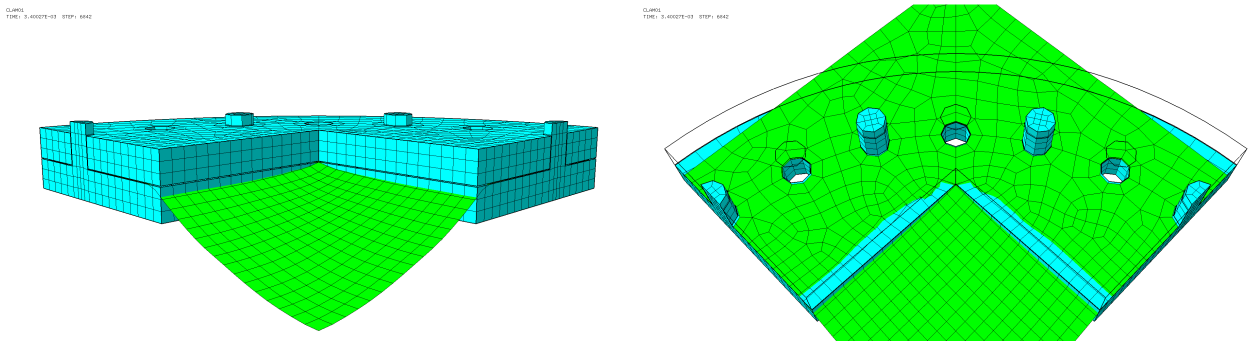
The `GLIS`-based solutions use a mesh of the test section, including both the (full) plate and the (full) clamping frame, in the form of a K-file produced at NTNU.

7.1.1 Case CLAM01

This test uses two `GLIS` sliding surfaces for the contact: the first one between the lower frame and bolts as master, and the plate nodes as slave, the second one between the upper frame as master and the plate nodes as slave. The final time is provisionally set to 3.4 ms. The plate is loaded by the experimentally recorded applied pressure, uniformly applied to the entire exposed area of the plate (a square of 0.3×0.3 m^2). The input directives for the boundary conditions read:

```
LINK COUP
     BLOQ 123 LECT NSET 1 TERM
     GLIS 2 FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #1
            PGAP 0.4E-3
            MAIT LECT PART 1 TERM
            PESC LECT PART 2  TERM
            FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #2
            PGAP 0.4E-3
            MAIT LECT PART 4 TERM
            PESC LECT PART 2 TERM
```

The calculation reaches the final time of 3.4 ms in 6842 time steps and 1600 s (27 minutes) of CPU time. However, as shown in Figure 100 which depicts only 1/4 of the model for clarity, the contact conditions are not completely respected. A narrow zone of the plate in the vicinity of the clamping frame border penetrates into the lower frame.



(a) Complete assembly

(b) Lower frame and plate

Figure 100: Plate deformation at 3.4 ms in case CLAM01 (1/4 of model shown)

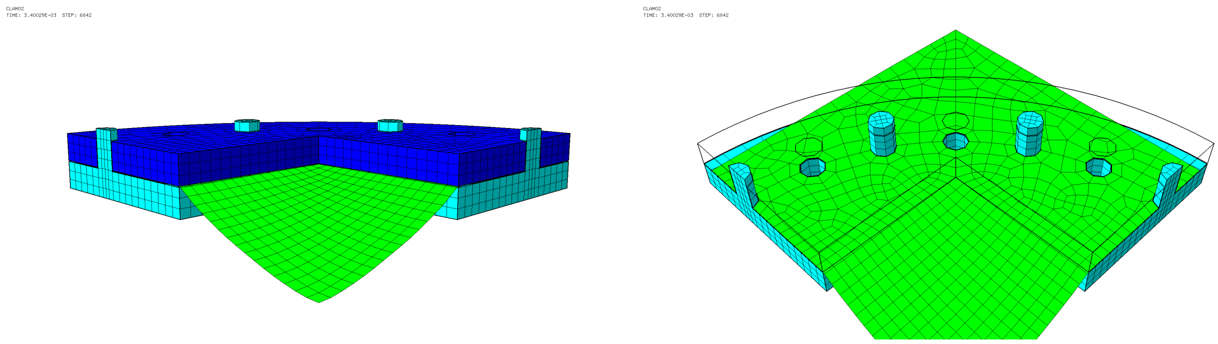
7.1.2 Case CLAM02

In order to avoid the spurious penetration observed in the previous simulation, a third `GLIS` sliding condition is added. The master surface consists of the shell elements in a zone of the plate near the clamping border, while the slave nodes are the lower frame nodes in the same region.

```
LINK COUP
     BLOQ 123 LECT NSET 1 TERM
     GLIS 3 FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #1
```

```
PGAP 0.4E-3
MAIT LECT part_1 TERM
PESC LECT part_2  TERM
FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #2
PGAP 0.4E-3
MAIT LECT part_4 TERM
PESC LECT part_2 TERM
FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #3
PGAP 0.4E-3
CMAI LECT cmai3 TERM EXTE LECT pext TERM
PESC LECT pesc3 TERM
```

This calculation reaches the final time of 3.4 ms in 6842 time steps and 1619 s (27 minutes) of CPU time. As shown in Figure 101, the contact conditions are now respected.



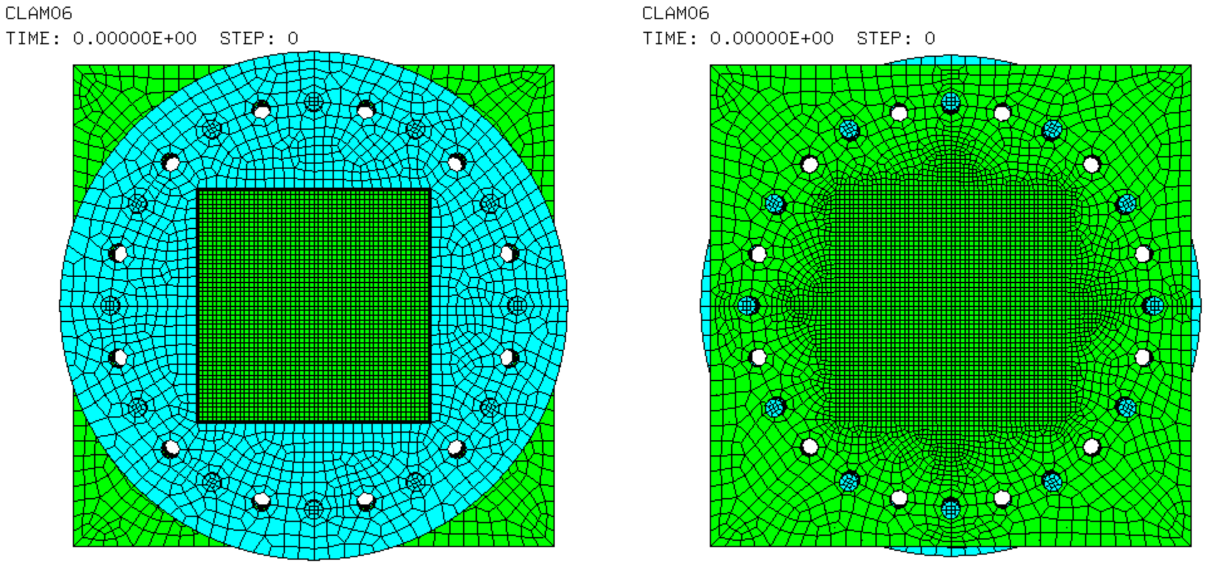
(a) Complete assembly

(b) Lower frame and plate

Figure 101: Plate deformation at 3.4 ms in case CLAM02 (1/4 of model shown)

7.1.3 Case CLAM06

This is a repetition of case `CLAM01` with a twice finer mesh in the exposed part of the plate, as shown in Figure 102. In fact, in order to treat plate failure in later simulations, a fine mesh (and probably also adaptive mesh refinement) will be necessary.



(a) Complete assembly

(b) Lower frame and plate

Figure 102: Initial mesh in case CLAM06

This solution takes 10 354 time steps and 1876 (31 minutes) of CPU time in order to reach the chosen final time of 3.4 ms. Strangely, some elements of the plate in the clamped area near the exposed zone of the plate fail starting at 2.13 ms. The “holes” corresponding to the 9 elements failed so far can be seen in Figure 103, which shows the plate at $t = 3.4$ ms. Similar spurious failures had been observed previously with the `GLIS` contact model in simulations of the blast-loaded plate presented in Section 6. The reason for this pathological behaviour of the `GLIS` model is under investigation.

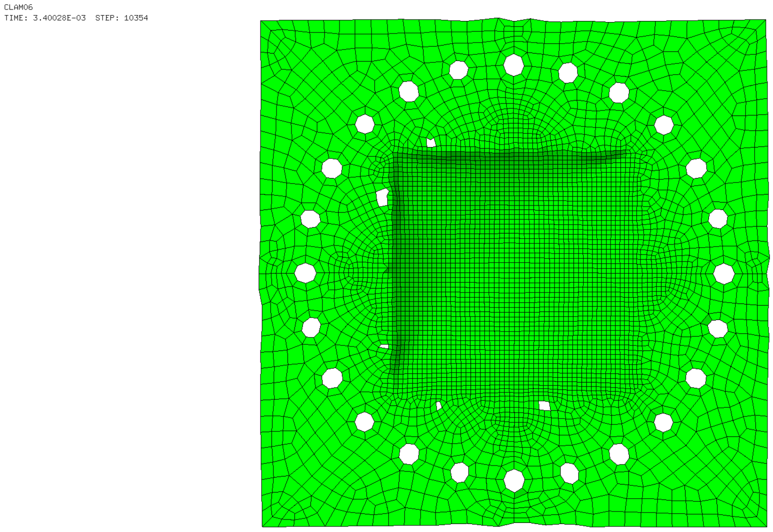
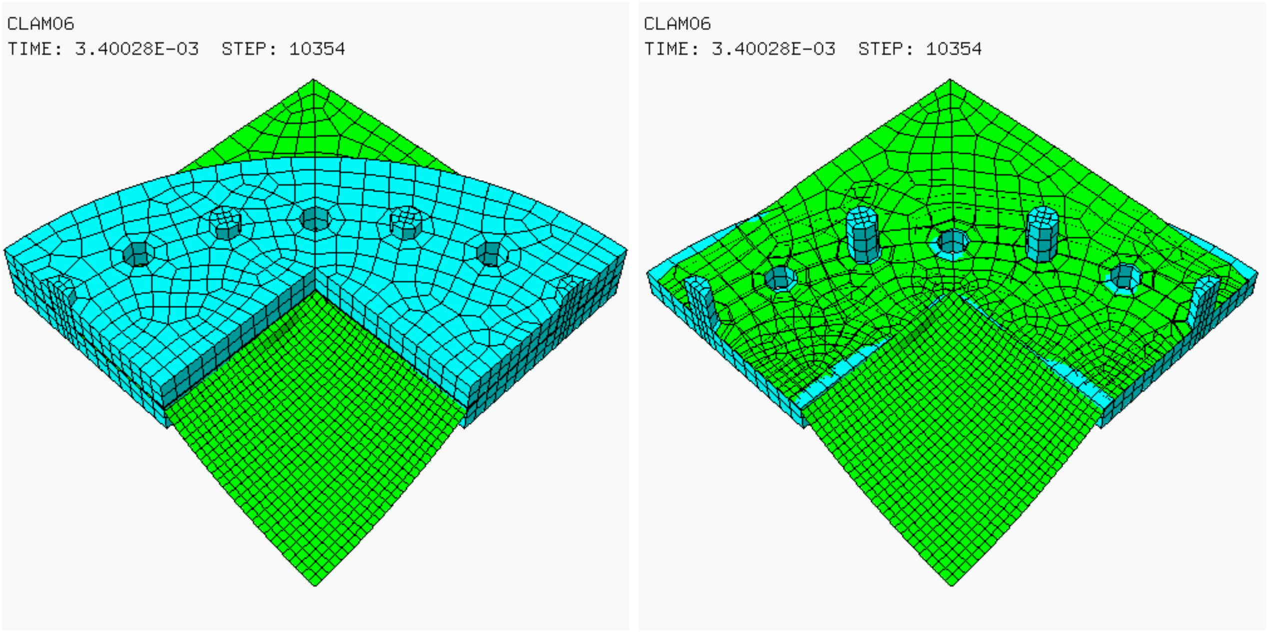


Figure 103: Final plate mesh with spurious failures in case CLAM06

Figure 104 shows 1/4 of the deformed assembly at $t = 3.4$ ms for comparison with the other solutions. One can notice some spurious penetrations of the plate into the lower clamping frame near the perimeter of the exposed area. This is not surprising since this test uses only two `GLIS` sliding surfaces similarly to case `CLAM01`. Introducing a third contact surface like in case `CLAM02` should avoid this problem.



(a) Complete assembly

(b) Lower frame and plate

Figure 104: Plate deformation at 3.4 ms in case CLAM06 (1/4 of model shown)

7.2 Solutions with PINB

We now try to obtain solutions with the `PINB` model of contact.

7.2.1 Case CLAM03

This test uses the same mesh as `CLAM01` and the `GLIS` contact directive is tentatively replaced by a `PINB` directive (coupled approach, using Lagrange Multipliers).

```
LINK COUP SPLT NONE
     BLOQ 123 LECT NSET 1 TERM
     PINB BODY DMIN 1.0E-4 MLEV 3 LECT PART 1 PART 4 TERM
          BODY DMIN 1.0E-4 MLEV 3 LECT PART 2 TERM
```

Tentatively, a minimum diameter `DMIN` of 0.1 mm is set for the (descendent) pinballs. This is a very small value, since the thickness of the plate is 0.8 mm. At the same time, a maximum recursion level (`MLEV`) of 3 is set for the descendent pinballs. Although the EPX manual states that the user should choose between `DMIN` and `MLEV`, i.e. specify only one of these two parameters, the code accepts the input without generating error messages during the input reading phase. It is believed that the given value of `MLEV` is then retained for the calculations, since this is the most constraining of the two conditions.

The code generates an error message when trying to solve the links at step 0, saying that the problem is ill-posed. This is due to the fact that with the chosen parameters (gap between the frames, thickness of the plate, size of the descendent pinballs) there occurs a huge number of pinball interpenetrations already in the initial configuration, so that the number of constraints is larger than the number of degrees of freedom of the problem. This may happen when, like in the present case, one uses a hierarchic pinball method of (too) large level.

Due to this error, the code stops and it is not possible to visualize the contacts (penetretions) in the initial configuration.

7.2.2 Case CLAM04

In order to avoid the error obtained in the previous simulation and to be able at least to visualize the found initial penetrations, a penalty-based pinball method is used instead of the LM-based method. The penalty method does not require the solution of a linear system of equations and therefore the above mentioned singularity is avoided. The input commands now read:

```
LINK COUP SPLT NONE
     BLOQ 123 LECT NSET 1 TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY DMIN 1.0E-4 MLEV 3 LECT PART 1 PART 4 TERM
          BODY DMIN 1.0E-4 MLEV 3 LECT PART 2 TERM
```

The initial penetrations visualized as red segments by the "*Show PINB joints*" interactive command are shown in Figure 105. Indeed, the number of interpenetration is excessive. Therefore, no attempt is made to obtain a numerical solution in these circumstances.

7.2.3 Case STPM01

A mesh (for 1/4 only) of the plate assembly in the shock tube is re-fabricated anew with Cast3m in order to gain more flexibility in setting up the contact conditions in EPX. While an attempt is made to produce a mesh as similar as possible to the K-mesh used in the previous simulations, it appears that Cast3m is unable to produce a quads-only mesh when filling the non-simply connected surface with holes of the plate. Some triangles are produced as well and therefore also the volumetric mesh of the clamps consists of some triangular prisms in addition to hexahedra. Also, the Cast3m mesh looks less regular than the K-mesh altogether.

The boundary-condition input directives are:

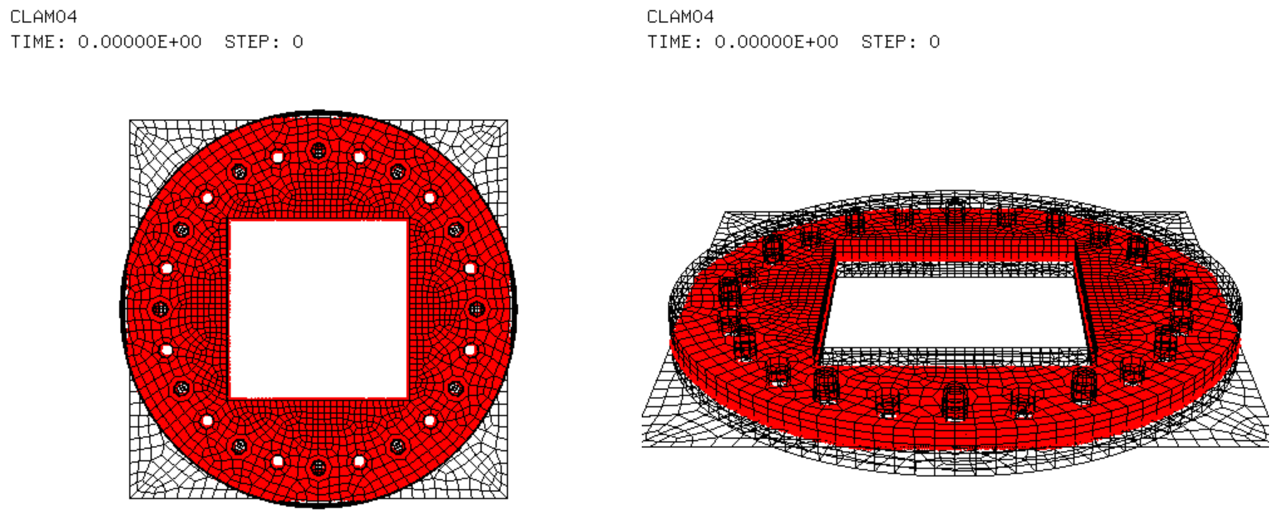


Figure 105: Initial pinball interpenetrations (contact joints) in case CLAM04

```
LINK COUP SPLT NONE
     BLOQ 3 LECT bloz TERM
     CONT SPLA NX 1 NY 0 NZ 0 LECT symx TERM
     CONT SPLA NX 0 NY 1 NZ 0 LECT symy TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY FROT MUST 0.15 MUDY 0.1 GAMM 0 MLEV 5
          LECT lframeb TERM
          BODY FROT MUST 0.15 MUDY 0.1 GAMM 0 MLEV 5
          LECT uframe TERM
          BODY FROT MUST 0.15 MUDY 0.1 GAMM 0 DIAM 0.0008
          LECT nplate TERM
```

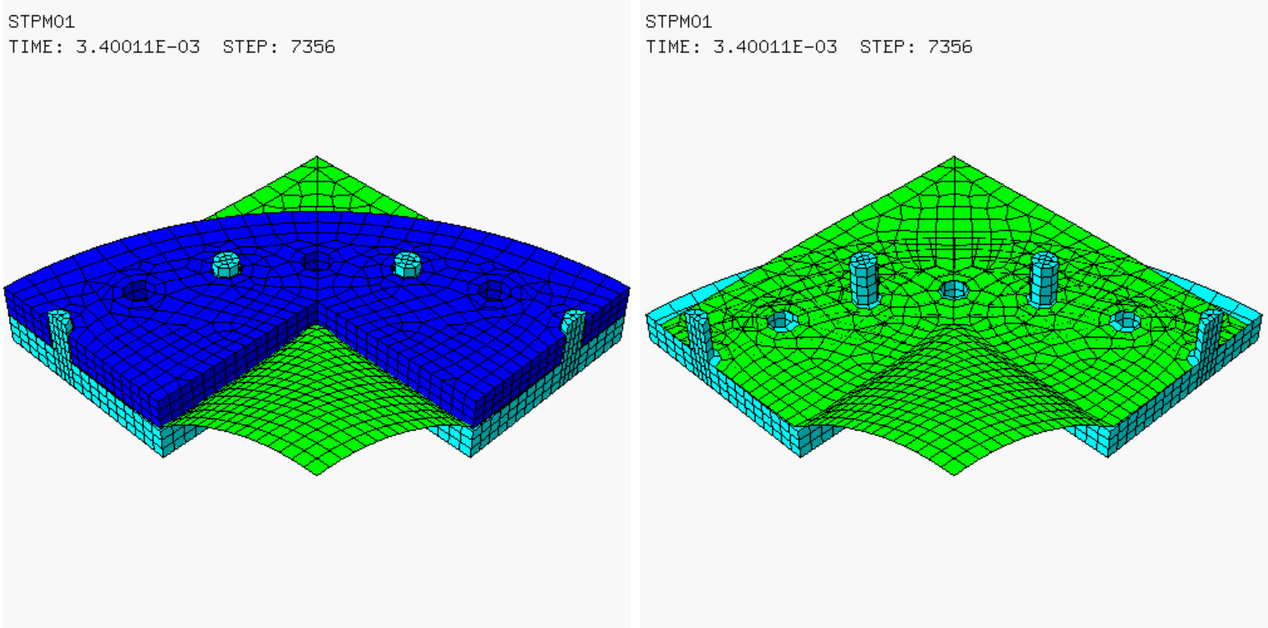
Three bodies are defined to be filled with pinballs. The first two correspond to the lower frame with bolts and to the upper frame, respectively. A hierarchy level as high as 5 is chosen (`MLEV 5`) in order to limit the number of contacts in the initial configuration. The third body is defined as the *nodes* belonging to the plate (`nplate` object). This object is fabricated in Cast3m from the plate mesh (`plate` object) by the Gibiane command:

```
nplate = chan poi1 plate;
```

In EPX, `PMAT` (one-node) elements (material points) are associated with the `nplate` object by the technique already described in Section 6.2.1. A thickness of 0.8 mm (`DIAM 0.0008`), equal to the physical thickness of the plate, is assigned to the pinballs attached to the material points.

This calculation ran until the chosen final time of 10 ms in 21 635 time steps and 9616 s (2.7 hours) of CPU time and was therefore substantially slower than the calculations with `GLIS` by considering that here only 1/4 of the model was used, by exploiting the symmetries. No failure occurred in the plate. The deformed mesh at $t = 3.4$ ms is shown in Figure 106 for comparison with the previous simulations.

Note that, due to an error in the sign of the applied pressure (or in the orientation of the pressurized plate zone) the plate displacement occurs in the opposite direction to the one obtained in cases `CLAM01` and `CLAM02`. The displacements in time of the plate center in the three cases `CLAM01`, `CLAM02` (with `GLIS`) and `STPM01` (with `PINB`) are drawn in Figure 107 for finer comparison. In the Figure, the displacement obtained in case `STPM01` has been multiplied by (-1.0). As one could expect, the displacement in case `CLAM02` (red curve) is slightly smaller than in case `CLAM01` (green curve), because of the stricter (and more realistic) contact conditions. The solution with pinballs `STPM01` (black curve) is somewhat softer than both solutions with `GLIS`.



(a) Complete assembly

(b) Lower frame and plate

Figure 106: Plate deformation at 3.4 ms in case STPM01

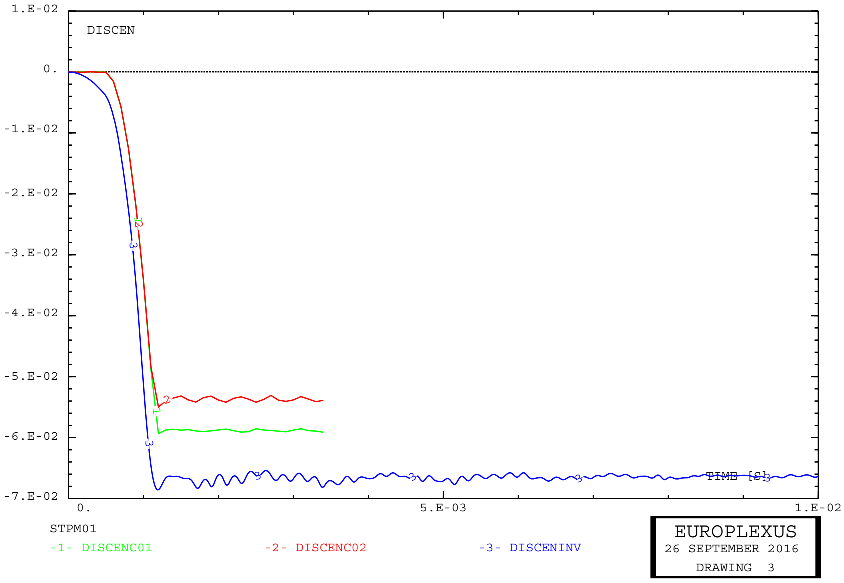


Figure 107: Comparison of plate center displacements in cases CLAM01, CLAM02 and STPM01

7.2.4 Case CLAM05

In this calculation we want to perform a simulation with `PINB` using exactly the same mesh as in cases `CLAM01` and `CLAM02` with `GLIS` (i.e., the K-file mesh provided by NTNU), in order to have a more fair comparison between the two contact methods.

The K-file mesh cannot be used directly because we need to add material points at the plate nodes as a support for pinballs in the plate, as already described in the previous case `STPM01`. Therefore, we will follow the following procedure:

1. Read the K-file with EPX and use a special option `K2MS` in the EPX input file in order to convert the K-file to a sort of simplified Cast3m mesh (see details below).
2. Open the Cast3m mesh with Cast3m, add the material points at the plate nodes and save the

modified mesh in Cast3m's `SAUV` format.

3. Use the modified Cast3m mesh to perform the EPX calculation.

The EPX option `K2MS` writes on a file a set of simple Gibiane (Cast3m) commands which can be read through Cast3m in order to generate the mesh used in the EPX calculation. All nodes and elements are generated. In addition, if one uses the `K2MS MANU` form of the option (as opposed to `K2MS READ`, which is the other possibility), all named element and node groups present in the EPX input file are also converted into Cast3m objects of the same name (but only the first 8 letters of the name are retained).

A simple EPX (preliminary) input file `CLAM050.epx` is prepared (e. g. from file `CLAM01.epx`) which contains just enough instructions to read the K-file and to produce the Cast3m input file by means of the `K2MS MANU` option:

```
CLAM050
ECHO
!CONV win
KFIL
TRID LAGR
GEOM CUB8 PART 1 PART 4 Q4GS PART 2 COQ4 PART 3 PART 5 TERM
COMP EPAI 0.8e-3 LECT PART 2 PART 3 PART 5 TERM
     GROU 5 'lframeb' LECT PART 1 TERM
            'plate'   LECT PART 2 TERM
            'preplat' LECT PART 3 TERM
            'uframe'  LECT PART 4 TERM
            'preclam' LECT PART 5 TERM
     NGRO 1 'bloc'    LECT NSET 1 TERM
MATE FANT 1.0 LECT tous TERM
OPTI NOTE CSTA 0.8
     K2MS MANU
CALC TINI 0 TEND 3.4E-3 NMAX 0
FIN
```

When run through EPX this input produces two files: a file `PXTOK200.dgibi` and another file `PXREST.dgibi`, which will be ignored (since it is just a file to verify the Cast3m mesh if needed). The first file (`PXTOK200.dgibi`) contains Gibiane instructions of the form:

```
OPTI ECHO 1 DIME 3;
P000001 =  -1.676500000000D-01  1.987300000000D-01 -3.999993200000D-04 ;
P000002 = . . .
 . . .
E000001=MANU CUB8 P000032 P000103 P000029 P000001 P000259 P000330
                  P000256 P000228;
MESH = E000001;
E000002=MANU CUB8 P000103 P000030 P000002 P000029 P000330 P000257
                  P000229 P000256;
MESH = MESH ET E000002;
 . . .

E000003= . . .
 . . .
```

These are Gibiane commands to create the nodes and elements of the mesh. In addition, Cast3m objects containing the named element and node groups (`lframeb`, `plate` etc.) present in the EPX file are also created.

Before running this file with Cast3m we rename it `CLAM05.dgibi` and modify it slightly. At the beginning of the file we insert a Gibiane procedure which allows to extract parts of a mesh, that will be used next in order to reduce the model from a full model to 1/4 model. We also change the `ECHO` setting from 1 to 0 in order to have less verbose outputs from Cast3m (and to speed up the execution).

```
'DEBPROC' pxextr3d m*'MAILLAGE' x1*'FLOTTANT' x2*'FLOTTANT'
                  y1*'FLOTTANT' y2*'FLOTTANT'
                  z1*'FLOTTANT' z2*'FLOTTANT';
*
*-------------------------------------------------
* Extracts from the 3D mesh m the elements whose nodes are
* located in the box [x1-x2,y1-y2,z1-z2].
```

```
*
* Input :
* -----
*     m                       : 3D mesh
*     x1, x2, y1, y2, z1, z2 : extremes of the box
* Output :
* ------
*     box : mesh contained in the box
*-------------------------------------------------
*
x = coor 1 m;
sx = x POIN COMP x1 x2;
y = coor 2 sx;
sy = y POIN COMP y1 y2;
z = coor 3 sy;
sz = z POIN COMP z1 z2;
box = m ELEM APPU STRI sz NOVE;
*
finproc box;
**********************************************************************
OPTI ECHO 0 DIME 3;
P000001 =  -1.676500000000D-01  1.987300000000D-01 -3.999993200000D-04 ;
P000002 = . . .
```

The end of the file is modified as follows. The `mesh` object is temporarily forgotten (`OUBL`), since it will be re-defined later. We use the initial procedure to reduce each geometric object to 1/4, i.e. we take only the portion located in the first quadrant. Then, we re-define the `mesh` object as the union of the just created objects (where the names of the original objects have been kept unchanged). We also create the plate nodes object `nplate`. We use the `TASS` operator to eliminate any holes in the node numbers that could have resulted from the previous operations, and we save the mesh.

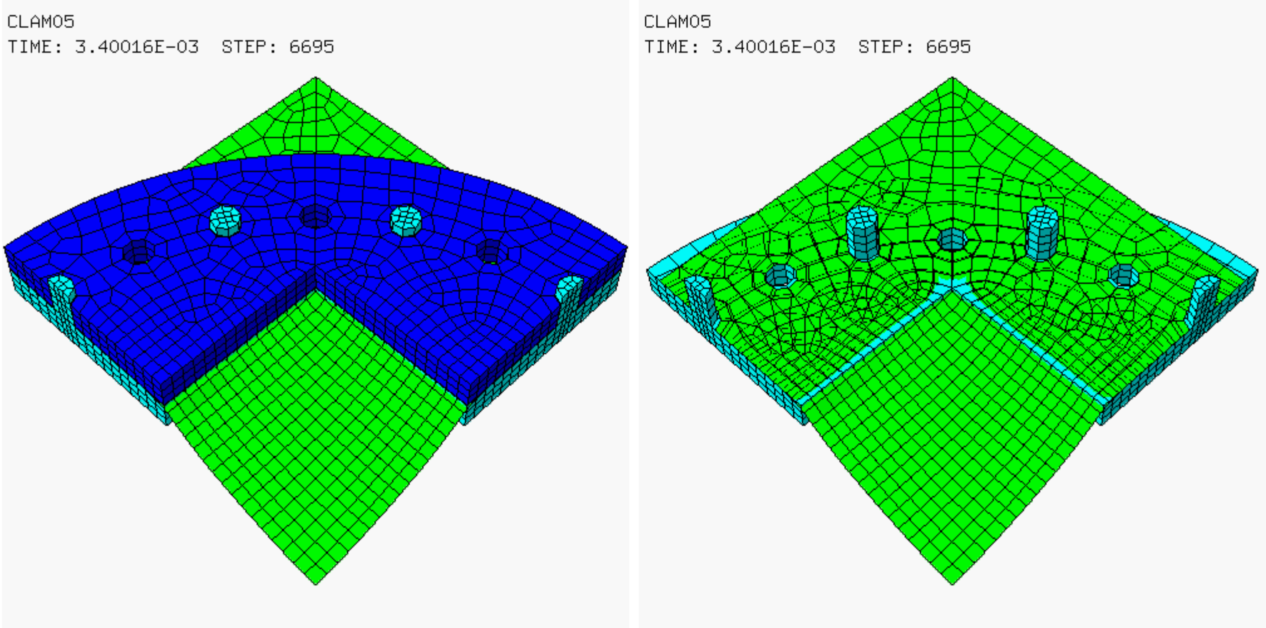
```
 . . .
OUBL E013325 ;
*
OUBL mesh;
lframeb = PXEXTR3D lframeb -0.0001 100.0 -0.0001 100.0 -100.0 100.0;
plate = PXEXTR3D plate -0.0001 100.0 -0.0001 100.0 -100.0 100.0;
preplat = PXEXTR3D preplat -0.0001 100.0 -0.0001 100.0 -100.0 100.0;
uframe = PXEXTR3D uframe -0.0001 100.0 -0.0001 100.0 -100.0 100.0;
preclam = PXEXTR3D preclam -0.0001 100.0 -0.0001 100.0 -100.0 100.0;
nplate = CHAN POI1 plate;
mesh = lframeb ET plate ET preplat ET uframe ET preclam ET
       nplate;
tass mesh noop;
*
OPTI TRAC PSC FTRA 'clam05_mesh.ps';
TRAC CACH QUAL MESH;
OPTI SAUV FORM 'clam05.msh';
SAUV FORM MESH;
FIN;
```

When run through Cast3m (which requires just a few seconds of CPU), this input produces the desired Cast3m mesh file `clam05.msh`. Next, we prepare an EPX input file `CLAM05.epx` similar to the previous case `STPM01`, i.e. using penalty-based pinballs for the contact, and run the simulation.

The calculation reaches the final time $t = 10$ ms in 19 691 time steps and 13 273 s (3.7 hours) of CPU time, being roughly 7 times slower than the corresponding calculation with `GLIS`.

The deformed mesh at $t = 3.4$ ms is shown in Figure 108 for comparison with the previous simulations. Note that some penetration of the plate into the lower frame near the preimeter of the exposed plate area occurs, similarly to case `CLAM01`. Unfortunataly, such a phenomenon may occur with penalty-based pinballs if the pinballs are very small (like in this case) and if the pinball stiffness is too low.

The displacements in time of the plate center in the various solutions obtained so far are compared in Figure 109. In the Figure, the displacement obtained in case `STPM01` has been multiplied by (-1.0). The present `CLAM05` solution (blue curve) is in slightly better agreement with the `GLIS`-based solutions (red and green curves) than the `STPM01` solution (cyan curve), both as concerns the maximum displacement and the rise shape of the displacement curve.



(a) Complete assembly

(b) Lower frame and plate

Figure 108: Plate deformation at 3.4 ms in case CLAM05

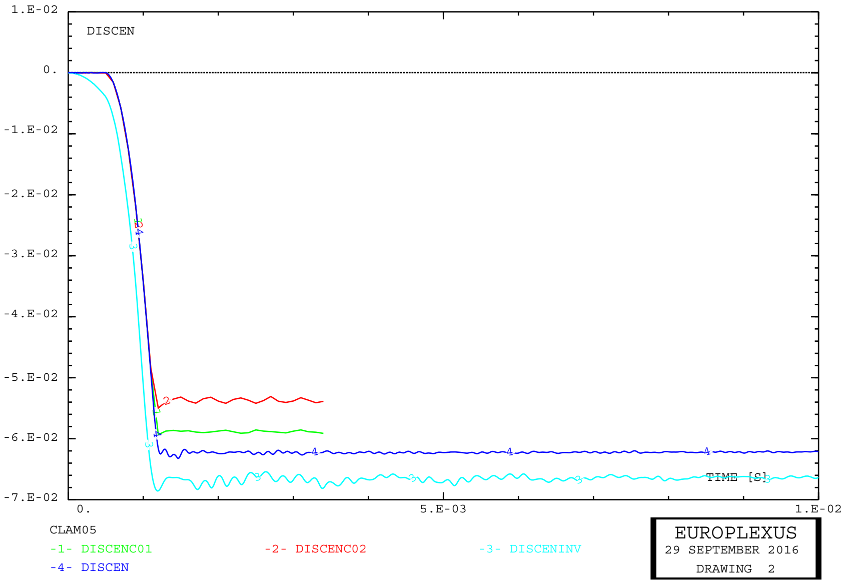


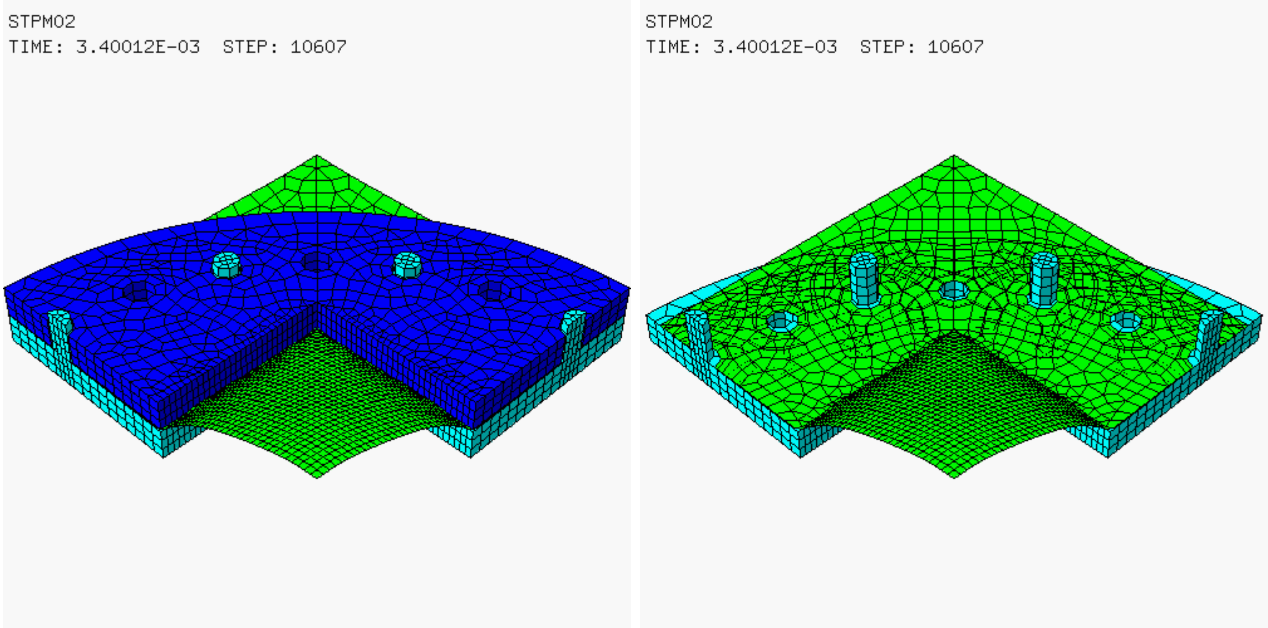
Figure 109: Plate center displacements in cases CLAM01, CLAM02, STPM01 and CLAM05

7.2.5 Case STPM02

This case is similar to `STPM01` (using a Cast3m native mesh of the 1/4 assembly) but the plate mesh in the exposed part of the plate is twice finer, similarly to case `CLAM06` with `GLIS` contact. The scope is to check if there is a sensitivity of the `PINB` model to the refinement of the mesh (locally near the clamped border of the plate), similar to what has been observed with `GLIS` in case `CLAM06`.

The calculation took 31 196 time steps and 15 100 s (4.2 hours) of CPU to reach the chosen final time of 10 ms. No spurious element failures (actually no failures at all) were observed. The deformed mesh at $t = 3.4$ ms is shown in Figure 110 for comparison with the previous simulations.

The displacements in time of the plate center in the various solutions obtained so far are compared in Figure 111.



(a) Complete assembly

(b) Lower frame and plate

Figure 110: Plate deformation at 3.4 ms in case STPM02

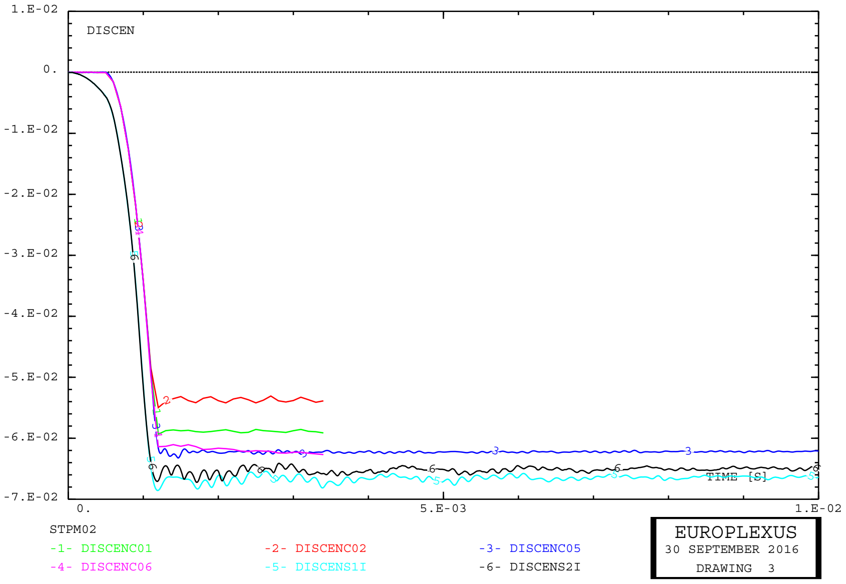


Figure 111: Plate center displacements in all cases

References

[1] EUROPLEXUS User's Manual, on-line version: http://europlexus.jrc.ec.europa.eu.

[2] Cast3m Software: http://www-cast3m.cea.fr/.

[3] J.O. Hallquist, G.L. Goudreau, D.J. Benson. *Sliding interfaces with contact-impact in large-scale Lagrangian computation.* Computer Methods in Applied Mechanics and Engineering **51** : 107–137, 1985.

[4] D.J. Benson, J.O. Hallquist. *A single surface contact algorithm for the postbuckling analysis of shell structures.* Report to the University of California at San Diego, CA, 1987.

[5] T. Belytschko, S.E. Law. *An Assembled Surface Normal Algorithm for Interior Node Removal in Three-Dimensional Finite Element Meshes.* Engineering with Computers **1** : 55–60, 1985.

[6] T. Belytschko, J.I. Lin. *A Three-Dimensional Impact-Penetration Algorithm with Erosion.* Computers and Structures **25** : 95–104, 1987.

[7] T. Belytschko, M.O. Neal. *Contact-Impact by the Pinball Algorithm with Penalty, Projection and Lagrangian Methods.* Computational Techniques for Contact, Impact, Penetration and Perforation of Solids, Winter Annual Meeting of the American Society of Mechanical Engineers, San Francisco, California **AMD 103** : 97–140, December 10–15, 1989.

[8] T. Belytschko, M.O. Neal. *The Vectorized Pinball Contact Impact Routine.* SMiRT-10 Conference, 1989.

[9] R.E. Sarwas. *Hidden Line Elimination and a Modified Pinball Algorithm for Finite Element Contact-Impact Problems with Shell Elements.* Master's Thesis, Northwestern University, Evanston, Illinois, December 1989.

[10] T. Belytschko, M.O. Neal. *Contact-Impact by the Pinball Algorithm with Penalty and Lagrangian Methods.* International Journal for Numerical Methods in Engineering **31** : 547–572, 1991.

[11] T. Belytschko, I.S. Yeh. *The Splitting Pinball Method for Contact-Impact Problems.* Computer Methods in Applied Mechanics and Engineering **105** : 375–393, 1993.

[12] E. Plaskacz, T. Belytschko, H.Y. Chiang. *Contact-Impact Simulations on Massively Parallel SIMD Supercomputers.* Argonne National Laboratory Report ANL/CP-76686, 1993.

[13] M. Lepareux, B. Schwab, A. Hoffmann, P. Jamet, H. Bung. *Un programme général pour l'analyse dynamique rapide – Cas des tuyauteries.* Colloque: Tendances Actuelles en Calcul des Structures, Bastia, 6–8 November, 1985.

[14] F. Casadei. *A Hierarchic Pinball Method for Contact-Impact in Fast Transient Dynamics.* VI Congresso Nazionale della Società Italiana di Matematica Applicata e Industriale (SIMAI 2002), Chia (Cagliari), Italy, 27–31 May 2002.
`178 -- Pdf/2002/Simai2002/Simai2002.pdf`

[15] F. Casadei. *A General Impact-Contact Algorithm Based on Hierarchic Pinballs for the EUROPLEXUS Software System.* Technical Note N. I.03.176, December 2003.
`202 -- Pdf/2003/Pinballs/Pinballs.pdf`

[16] F. Casadei. *Implementation of Fast Search Algorithms in EUROPLEXUS.* Technical Note, PUBSY No. JRC38086, October 2007.
`230 -- Pdf/2007/Fast_Search/FastSearch.pdf`

[17] F. Casadei. *Validation of the EUROPLEXUS Pinball Impact-Contact Model on an Indentation Problem.* Technical Note N. I.05.51, July 2005.
`214 -- Pdf/2005/Indentation/Indentation.pdf`

[18] F. Casadei, B. Langrand, M. Larcher, G. Valsamos. *Pinball-based Contact-Impact Model with Parabolic Elements in EUROPLEXUS.* Technical Note, PUBSY No. JRC89913, EUR Report 26629 EN, 2014.
`314 -- Pdf/2014/Ortiz_Pinballs_Q9/Report/Pinballs_parabolic_JRC.pdf`

[19] F. Casadei, M. Larcher, G. Valsamos, V. Faucher. *Implementation of Assembled Surface Normals and of a Penalty Contact Formulation in the Pinball Model of Europlexus.* Technical Note, PUBSY No. JRC90939, EUR 26714 EN, 2014.
`316 -- Pdf/2014/Pinballs_ASN_Penalty/Report/Pinballs_ASN_Penalty_JRC.pdf`

[20] F. Casadei, V. Aune, G. Valsamos, M. Larcher. *Generalization of the pinball contact/impact model for use with mesh adaptivity and element erosion in EUROPLEXUS.* JRC Technical Report, PUBSY No. JRC101013, EUR Report 27888 EN, 2016.
`351 -- Pdf/2016/ADAP_PINB_EROS/Report/Front/FullRepAdapPinb.pdf`

[21] F. Casadei, G. Valsamos, M. Larcher, V. Aune. *Implementation of friction in the pinball-based contact-impact model of EUROPLEXUS.* JRC Technical Report, PUBSY No. JRC112047, 2018.
`365 -- Pdf/2016/PINB_Friction/Report/Front/Req_JRC112047.pdf`

[22] F. Casadei, G. Valsamos, M. Larcher. *A contact-impact model based on generalized pinballs in EUROPLEXUS.* JRC Technical Report, PUBSY No. JRCxxxxxx, in preparation.
`708 -- Pdf/2019/Gpinballs_2019/Report/Gpinballs.pdf`

[23] S. Potapov. *Correction de l'algorithme de contact frottant 3D dans EUROPLEXUS.* Compte Rendu, EDF, CR-T61-2018-xxx, 2018.

[24] R. Galon, H. Bung, M. Lepareux. *Programme PLEXUS. Algorithme de traitement des surfaces de glissement (3D).* CEA Rapport DEMT 90/092, 1990.

[25] H. Bung, M. Lepareux, R. Galon. *Programme PLEXUS. Prise en compte du frottement dans les lignes et surfaces de glissement.* CEA Rapport DEMT 90/203, 1990.

[26] J. Bonini, H. Bung. *Programme PLEXUS. Fiche théorique. Modélisation des impacts avec frottement par la méthode des multiplicateurs de Lagrange.* CEA Rapport DMT 97/367, 1997.

[27] J. Bonini. *Programme PLEXUS. Fiches de validation. Modélisation des impacts avec frottement par la méthode des multiplicateurs de Lagrange.* CEA Rapport DMT 97/368, 1997.

[28] P. Galon. *Calculs dynamiques d'un essai Charpy miniature avec prise en compte ou non du frottement.* CEA Rapport DMT/DEMT/DYN/RT/00-024/A, 2000.

[29] S. Moulin, S. Potapov. *Modélisation du contact dans EUROPLEXUS. Théorie et éléments de validation.* EDF Rapport HT-62/05/012/A, 2006.

[30] H. Bung. *Prise en compte du contact avec ou sans frottement.* Hand-written notes (private communication), 29 July 2014.

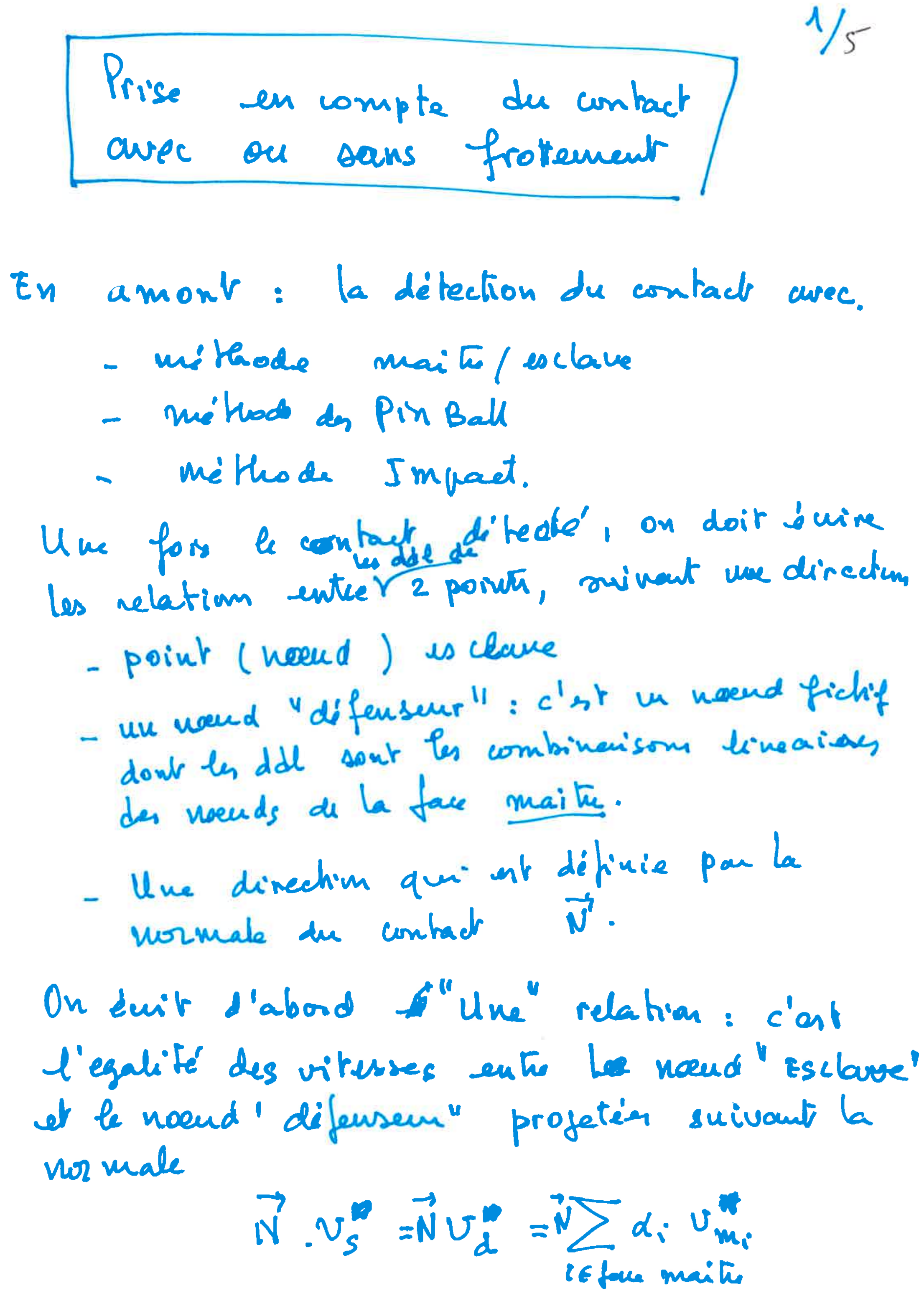
[31] F. Casadei, D. Markovic, G. Valsamos, M. Larcher. *Correction of friction implementation in the LM-based PINB contact model of EUROPLEXUS.* JRC Technical Report, PUBSY No. JRCxxxxxx, in preparation.
`718 -- Pdf/2023/Markovic/2023_05_17_Friction/Report/FRIC_LM_PINB.pdf`

[32] F. Bliard. *Modélisation de la perforation du bois avec EUROPLEXUS : Mise en oeuvre d'un processus de décohésion des éléments reposant sur un critère de rupture.* CEA Rapport DEN/DANS/DM2S/SEMT/DYN/NT/14-019/A, 2014.

[33] F. Casadei, G. Valsamos, M. Larcher. *Testing of the GLIS contact model in EUROPLEXUS.* JRC Technical Note Pubsy N. JRC101012, EUR Report 27889 EN, 2016.
`346 -- Pdf/2015/Glis_test/Report/Front/JRC101012_fullreportglisnew.pdf`

Appendix A — Hand notes on friction

Here are the hand notes corresponding to reference [30].



1/5

Prise en compte du contact
avec ou sans frottement

En amont : la détection du contact avec.

- méthode maître/esclave
- méthode des Pin Ball
- méthode Impact.

Une fois le contact détecté, on doit écrire les relation entre les ddl de 2 points, suivant une direction

- point (noeud) esclave
- un noeud "défenseur" : c'est un noeud fictif dont les ddl sont les combinaisons linéaires des noeuds de la face maître.
- Une direction qui est définie par la normale du contact $\vec{N}$.

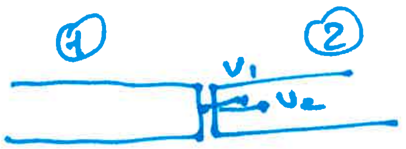
On écrit d'abord "Une" relation : c'est l'égalité des vitesses entre les noeud 'esclave' et le noeud 'défenseur" projetées suivant la normale

$$\vec{N} . v_s^{*} = \vec{N} v_d^{*} = \vec{N} \sum_{i \in \text{face maître}} d_i \, v_{m_i}^{*}$$

2/5

Quelles vitesse à prendre en compte?

Prenons 1 exemple.



Deux solides ① et ② qui viennent d'impacter à l'interface de contact. On a 2 vitesses v_1 et v_2, et 2 forces F_1 et F_2.

Or le système composé de ces 2 solides est un système "fermé". Donc l'énergie totale reste constante ⇒ l'incrément de l'énergie de choc doit être nul.

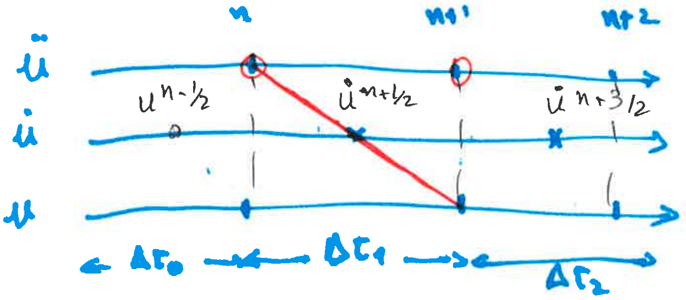
$$\Delta W = F_1 \cdot \Delta u_1 + F_2 \cdot \Delta u_2 .$$

Principe égalité action/réaction ⇒ $F_1 = -F_2$

Donc $\Delta W = 0 \Rightarrow \boxed{\Delta u_1 = \Delta u_2}$

Les incrément de déplacements doivent être égaux

Revenons à l'algorithme temporel de EPX basé sur la méthode différence centrée.



Au pas n, on connait $\ddot{u}^{n}$, puis 3/5

- vitesse au pas $(n+1/2)$ $= \dot{u}^{n+1/2} = \dot{u}^{n-\frac{1}{2}} + \frac{\Delta t_0 + \Delta t_1}{2} \ddot{u}^{n}$
- depl. au pas $n+1$: $u^{n+1} = u^{n} + \Delta t_1 . \dot{u}^{n+\frac{1}{2}}$

Après on écrit l'équation du mouvement au pas $n+1$:

$$M\ddot{u}^{n+1} = F_e - F_i + F_c$$

Les valeurs situées à gauche du trait rouge ne peuvent plus changer, donc les relations de liaison sur les vitesses ne peuvent porter que sur $\dot{u}^{n+1}$, $\dot{u}^{n+3/2}$ et $\dot{u}^{n+2}$. Or l'incrément sur le déplacement $\Delta u_s = \Delta u_d$ et $u^{n+2} - u^{n+1} = \Delta t_2 . \dot{u}^{n+3/2}$ implique que l'on a une relation j avec les vitesses au $n+3/2$

$$\vec{N} . \dot{u}_s^{n+3/2} = \vec{N} . \dot{u}_d^{n+3/2} = \vec{N} \sum \alpha_i \dot{u}_{mi}^{n+3/2}$$

associée au multiplicateur de Lagrange λ_j

Prise en compte du frottement.

Au vecteur normal $\vec{N}$, on cherche le vecteur $\vec{T}$ tangentiel. Pour calculer ce vecteur on propose de projecter le vecteur de vitesse relative $(\dot{u}_s^{n+1/2} - \dot{u}_d^{n+1/2})$ dans le plan perpendiculaire au vecteur normal $\vec{N}$

Puis on écrit une deuxieme relation (j+1) 4/5

$$\vec{T}\,\dot{u}_s^{n+3/2} = \vec{T}\,\dot{u}_d^{n+3/2} = \vec{T}\sum \alpha_i\,\dot{u}_{m_i}^{n+3/2}$$

En resumé, on a une liaisons (suivant la normale) associée au multiplicateur λ_j.
La prise compte du frottement ajoute une 2e relation (suivant le vecteur tangentiel $\vec{T}$) associé à d_{j+1}

On resond alors le système avec des multiplicateur de Lagrange ⇒
On obtient d_j et $F_c = C^T\lambda$

1) $\lambda_j \geq 0$ c'est une traction ⇒ un decollement
On supprime la liaison en mettant $d_j = 0$
s'il y a prise en compte du frottement on met $d_{j+1} = 0$
et on recalculer $F_c = C^T\lambda$ avec les nouvelles valeurs de $d_j = 0$ (et $d_{j+1} = 0$).

2) $d_j < 0$, l'impact a toujours lieu.

a) pas de prise en compte du frottement
on sort de la séquence.

b) Prise en compte du frottement.
On calcule la force F'_{cT} = [illegible] contenu dans $C^T d_{j+1}$ appliquée au noeud esclave

5/5

$$\rightarrow \quad |F'_T| \leq \mu |F'_N|$$

μ : coefficient de frottement de Coulomb

F'_N = Force suivant $\vec{N}$, ~~obtenue~~ appliquée au noeud esclave, obtenue par la relation précédente λ_i

Dans ce cas, on laisse poursuivre la séquence

$$\rightarrow \quad |F'_T| > \mu |F'_N| \quad \text{on a} \quad \alpha = \frac{\mu |F'_N|}{|F'_T|} < 1$$

On modifie la Force,

$$F'_T = \alpha F'_{T\,\text{ancien}}$$

De même pour les forces appliquées aux noeuds maitres, on les multiplie par le coefficient α.

Appendix B — Input files

All the input files used in the previous Sections are listed below.

clam01.epx

```
CLAM01
ECHO
!CONV win
KFIL
EROS 1.0
TRID LAGR
GEOM CUB8 PART 1 PART 4 Q4GS PART 2 CL3D PART 3 PART 5 TERM
COMP EPAI 0.8e-3 LECT PART 2 TERM
     COUL TURQ LECT PART 1 TERM
          TURQ LECT PART 4 TERM
          VERT LECT PART 2 TERM
          ROSE LECT PART 5 TERM
     NGRO 3 ’centd’ LECT PART 2 TERM COND NEAR POIN 0.0 0.0 0.0
            ’centl’ LECT PART 3 TERM COND NEAR POIN 0.0 0.0 0.0
            ’axis’  LECT PART 2 TERM
            COND LINE X1 -0.2 Y1 -0.001 Z1 -0.001
                              X2  0.2 Y2  0.001 Z2  0.001 TOL 1.E-5
     ORIE INVE LECT PART 3 TERM
INCLUDE ’p77_75.txt’ ! Fonc definition
MATE VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 3.257E8 mxit 500
          QR1 2.348E8 CR1 56.2 QR2 4.457E8 CR2 4.7
          PDOT 5.E-4 C 1.E-2 TQ 0.9 CP 452.0
          TM 1800.0 M 1.0 DC 1.0 WC 555.0E6
          LECT PART 1 PART 2 PART 4 TERM
     IMPE PIMP RO 7850.0 PRES 42.1e6 PREF 0
          LECT PART 5 TERM
     IMPE PIMP RO 7850 PRES 1.0 PREF 1.0E5 FONC 1
          LECT PART 3 TERM
LINK COUP
     BLOQ 123 LECT NSET 1 TERM
     GLIS 2 FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #1
            PGAP 0.4E-3
            MAIT LECT PART 1 TERM
            PESC LECT PART 2  TERM
            FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #2
            PGAP 0.4E-3
            MAIT LECT PART 4 TERM
            PESC LECT PART 2 TERM
ECRI DEPL VITE ECRO FAIL TFRE 1.E-3
          POIN LECT 1 TERM
          ELEM LECT 1 TERM
     FICH SPLI ALIC TFRE 0.1E-3
OPTI NOTE CSTA 0.8
     LOG 1
     JAUM
     LMST
CALC TINI 0 TEND 3.4E-3
FIN
```

clam01a.epx

```
CLAM01A
ECHO
 CONV WIN
RESU SPLI ALIC ’clam01.ali’ GARD PSCR
COMP GROU 3 ’one4’ LECT PART 1 TERM
                   COND XB GT 0.0
                   COND YB GT 0.0
            ’two4’ LECT PART 2 TERM
                   COND XB GT 0.0
                   COND YB GT 0.0
            ’fou4’ LECT PART 4 TERM
                   COND XB GT 0.0
                   COND YB GT 0.0
OPTI PRIN
SORT VISU NSTO 1
PLAY
CAME   1 EYE  -3.21616E-01 -3.21616E-01  8.21785E-01
!        Q     8.73545E-01  3.00786E-01 -1.24589E-01 -3.61834E-01
         VIEW  4.35338E-01  4.35339E-01 -7.88011E-01
         RIGH  7.07107E-01 -7.07107E-01 -4.22336E-07
         UP    5.57208E-01  5.57208E-01  6.15662E-01
         FOV   2.48819E+01
!NAVIGATION MODE: FREE CAMERA
!CENTER :  1.62500E-01  1.62500E-01  4.60000E-03
!RSPHERE:  2.31760E-01
!RADIUS :  1.06608E+00
!ASPECT :  1.00000E+00
!NEAR   :  8.33700E-01
!FAR    :  1.29722E+00
SCEN GEOM NAVI FREE
          FACE SBAC
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 34 FPS 15 KFRE 10 COMP -1 NFAI
!    OBJE LECT one4 two4 fou4 TERM REND
     OBJE LECT one4 two4 TERM REND
GOTR LOOP 32 OFFS FICH AVI CONT NOCL NFAI
!    OBJE LECT one4 two4 fou4 TERM REND
     OBJE LECT one4 two4 TERM REND
GO
TRAC OFFS FICH AVI CONT NFAI
!    OBJE LECT one4 two4 fou4 TERM REND
     OBJE LECT one4 two4 TERM REND
ENDPLAY
FIN
```

clam01b.epx

```
CLAM01B
ECHO
RESU SPLI ALIC ’clam01.ali’ GARD PSCR
COMP NGRO 1 ’centd’ LECT PART 2 TERM COND NEAR POIN 0.0 0.0 0.0
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR 1 ’DisCen’ DEPL COMP 3 POIN LECT centd TERM
TRAC 1 AXES 1.0 ’DisCen’ YZER
     COLO bleu
     THIC 0.8
LIST 1 AXES 1.0 ’DisCen’ YZER
FIN
```

clam02.epx

```
CLAM02
ECHO
!CONV win
KFIL
EROS 1.0
TRID LAGR
GEOM CUB8 PART 1 PART 4 Q4GS PART 2 CL3D PART 3 PART 5 TERM
COMP EPAI 0.8e-3 LECT PART 2 TERM
     COUL TURQ LECT PART 1 TERM
          VERT LECT PART 2 TERM
          ROUG LECT PART 3 TERM
          BLEU LECT PART 4 TERM
          ROSE LECT PART 5 TERM
     GROU 8 ’part_1’ LECT part 1 TERM
            ’part_2’ LECT part 2 TERM
            ’part_3’ LECT part 3 TERM
            ’part_4’ LECT part 4 TERM
            ’part_5’ LECT part 5 TERM
            ’dmai3’  LECT part_2 TERM
                     COND XB LT 0.180
                     COND XB GT -0.180
                     COND YB LT 0.180
                     COND YB GT -0.180
            ’imai3’  LECT part_2 TERM
                     COND XB LT 0.140
                     COND XB GT -0.140
                     COND YB LT 0.140
                     COND YB GT -0.140
            ’cmai3’ LECT dmai3 DIFF imai3 TERM
     NGRO 5 ’centd’ LECT part_2 TERM COND NEAR POIN 0.0 0.0 0.0
            ’centl’ LECT part_3 TERM COND NEAR POIN 0.0 0.0 0.0
            ’axis’  LECT part_2 TERM
            COND LINE X1 -0.2 Y1 -0.001 Z1 -0.001
                              X2  0.2 Y2  0.001 Z2  0.001 TOL 1.E-5
            ’pesc3’ LECT part_1 TERM
                    COND X LT 0.165
                    COND X GT -0.165
                    COND Y LT 0.165
                    COND Y GT -0.165
                    COND Z GT -0.005
            ’pext’  LECT part_1 TERM
                    COND NEAR POIN 0.15 0.15 -0.1
     ORIE INVE LECT part_3 TERM
INCLUDE ’p77_75.txt’ ! Fonc definition
MATE VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 3.257E8 mxit 500
          QR1 2.348E8 CR1 56.2 QR2 4.457E8 CR2 4.7
          PDOT 5.E-4 C 1.E-2 TQ 0.9 CP 452.0
          TM 1800.0 M 1.0 DC 1.0 WC 555.0E6
          LECT part_1 part_2 part_4 TERM
     IMPE PIMP RO 7850.0 PRES 42.1e6 PREF 0
          LECT part_5 TERM
     IMPE PIMP RO 7850 PRES 1.0 PREF 1.0E5 FONC 1
          LECT part_3 TERM
LINK COUP
     BLOQ 123 LECT NSET 1 TERM
     GLIS 3 FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #1
            PGAP 0.4E-3
            MAIT LECT part_1 TERM
            PESC LECT part_2  TERM
            FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #2
            PGAP 0.4E-3
            MAIT LECT part_4 TERM
            PESC LECT part_2 TERM
            FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #3
            PGAP 0.4E-3
            CMAI LECT cmai3 TERM EXTE LECT pext TERM
            PESC LECT pesc3 TERM
ECRI DEPL VITE ECRO FAIL TFRE 1.E-3
          POIN LECT 1 TERM
```

```
          ELEM LECT 1 TERM
     FICH SPLI ALIC TFRE 0.1E-3
OPTI NOTE CSTA 0.8
     LOG 1
     JAUM
     LMST
CALC TINI 0 TEND 3.4E-3
FIN
```

clam02b.epx

```
CLAM02B
ECHO
RESU SPLI ALIC ’clam02.ali’ GARD PSCR
COMP NGRO 1 ’centd’ LECT PART 2 TERM COND NEAR POIN 0.0 0.0 0.0
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR 1 ’DisCen’ DEPL COMP 3 POIN LECT centd TERM
TRAC 1 AXES 1.0 ’DisCen’ YZER
     COLO bleu
     THIC 0.8
LIST 1 AXES 1.0 ’DisCen’ YZER
FIN
```

clam03.epx

```
CLAM03
ECHO
 CONV win
KFIL
EROS 1.0
TRID LAGR
GEOM CUB8 PART 1 PART 4 Q4GS PART 2 CL3D PART 3 PART 5 TERM
COMP EPAI 0.8e-3 LECT PART 2 TERM
     COUL TURQ LECT PART 1 TERM
          TURQ LECT PART 4 TERM
          VERT LECT PART 2 TERM
          ROSE LECT PART 5 TERM
     NGRO 3 ’centd’ LECT PART 2 TERM COND NEAR POIN 0.0 0.0 0.0
            ’centl’ LECT PART 3 TERM COND NEAR POIN 0.0 0.0 0.0
            ’axis’  LECT PART 2 TERM
            COND LINE X1 -0.2 Y1 -0.001 Z1 -0.001
                              X2  0.2 Y2  0.001 Z2  0.001 TOL 1.E-5
     ORIE INVE LECT PART 3 TERM
INCLUDE ’p77_75.txt’
MATE VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 3.257E8 mxit 500
          QR1 2.348E8 CR1 56.2 QR2 4.457E8 CR2 4.7
          PDOT 5.E-4 C 1.E-2 TQ 0.9 CP 452.0
          TM 1800.0 M 1.0 DC 1.0 WC 555.0E6
          LECT PART 1 PART 2 PART 4 TERM
     IMPE PIMP RO 7850.0 PRES 42.1e6 PREF 0
          LECT PART 5 TERM
     IMPE PIMP RO 7850 PRES 1.0 PREF 1.0E5 FONC 1
          LECT PART 3 TERM
LINK COUP SPLT NONE
     BLOQ 123 LECT NSET 1 TERM
     PINB BODY DMIN 1.0E-4 MLEV 3 LECT PART 1 PART 4 TERM
          BODY DMIN 1.0E-4 MLEV 3 LECT PART 2 TERM
*    GLIS 2 FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #1
*           PGAP 0.4E-3
*           MAIT LECT PART 1 TERM
*           PESC LECT PART 2  TERM
*           FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #2
*           PGAP 0.4E-3
*           MAIT LECT PART 4 TERM
*           PESC LECT PART 2 TERM
ECRI DEPL VITE ECRO FAIL TFRE 1.E-3
          POIN LECT 1 TERM
          ELEM LECT 1 TERM
     FICH SPLI ALIC TFRE 0.1E-3
OPTI NOTE CSTA 0.8
     LOG 1
     JAUM
     LMST
     lnks stat
CALC TINI 0 TEND 3.4E-3
FIN
```

clam04.epx

```
CLAM04
ECHO
 CONV win
KFIL
EROS 1.0
TRID LAGR
GEOM CUB8 PART 1 PART 4 Q4GS PART 2 CL3D PART 3 PART 5 TERM
COMP EPAI 0.8e-3 LECT PART 2 TERM
     COUL TURQ LECT PART 1 TERM
          TURQ LECT PART 4 TERM
          VERT LECT PART 2 TERM
          ROSE LECT PART 5 TERM
     NGRO 3 ’centd’ LECT PART 2 TERM COND NEAR POIN 0.0 0.0 0.0
            ’centl’ LECT PART 3 TERM COND NEAR POIN 0.0 0.0 0.0
            ’axis’  LECT PART 2 TERM
            COND LINE X1 -0.2 Y1 -0.001 Z1 -0.001
                              X2  0.2 Y2  0.001 Z2  0.001 TOL 1.E-5
     ORIE INVE LECT PART 3 TERM
INCLUDE ’p77_75.txt’
MATE VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 3.257E8 mxit 500
          QR1 2.348E8 CR1 56.2 QR2 4.457E8 CR2 4.7
          PDOT 5.E-4 C 1.E-2 TQ 0.9 CP 452.0
          TM 1800.0 M 1.0 DC 1.0 WC 555.0E6
          LECT PART 1 PART 2 PART 4 TERM
     IMPE PIMP RO 7850.0 PRES 42.1e6 PREF 0
          LECT PART 5 TERM
     IMPE PIMP RO 7850 PRES 1.0 PREF 1.0E5 FONC 1
          LECT PART 3 TERM
LINK COUP SPLT NONE
     BLOQ 123 LECT NSET 1 TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY DMIN 1.0E-4 MLEV 3 LECT PART 1 PART 4 TERM
          BODY DMIN 1.0E-4 MLEV 3 LECT PART 2 TERM
*    GLIS 2 FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #1
*           PGAP 0.4E-3
*           MAIT LECT PART 1 TERM
*           PESC LECT PART 2  TERM
*           FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #2
*           PGAP 0.4E-3
*           MAIT LECT PART 4 TERM
*           PESC LECT PART 2 TERM
ECRI DEPL VITE ECRO FAIL TFRE 1.E-3
          POIN LECT 1 TERM
          ELEM LECT 1 TERM
     FICH SPLI ALIC TFRE 0.1E-3
OPTI NOTE CSTA 0.8
     LOG 1
     JAUM
     LMST
     lnks stat
CALC TINI 0 TEND 3.4E-3
FIN
```

clam05.epx

```
CLAM05
ECHO
!CONV win
CAST mesh
EROS 1.0
TRID LAGR
GEOM CUB8 lframeb uframe
     Q4GS plate
     PMAT nplate
     CL3D preplat preclam
TERM
COMP EPAI 0.8e-3 LECT plate nplate TERM
     COUL TURQ LECT lframeb TERM
          BLEU LECT uframe TERM
          VERT LECT plate TERM
          ROSE LECT preclam TERM
          ROUG LECT preplat nplate TERM
     NGRO 4 ’bloz’  LECT lframeb TERM COND Z LT -0.0253
            ’symx’  LECT mesh TERM    COND X LT 0.0001
            ’symy’  LECT mesh TERM    COND Y LT 0.0001
            ’p0’    LECT plate TERM   COND NEAR POIN 0.0 0.0 0.0
     GROU 1 ’prpl’  LECT preplat TERM COND NEAR POIN 0.0 0.0 0.0
     ORIE INVE LECT preplat TERM
INCLUDE ’p77_75.txt’ ! Fonc definition
MATE VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 3.257E8 mxit 500
          QR1 2.348E8 CR1 56.2 QR2 4.457E8 CR2 4.7
          PDOT 5.E-4 C 1.E-2 TQ 0.9 CP 452.0
          TM 1800.0 M 1.0 DC 1.0 WC 555.0E6
          LECT lframeb plate uframe TERM
     MASS 0.0 YOUN 2.1E11 NU 0.33
          LECT nplate TERM
     IMPE PIMP RO 7850.0 PRES 43.1e6 PREF 1.0E5
          LECT preclam TERM
     IMPE PIMP RO 7850 PRES 1.0 PREF 1.0E5 FONC 1
          LECT preplat TERM
OPTI PINS ASN
LINK COUP SPLT NONE
     BLOQ 3 LECT bloz TERM
     CONT SPLA NX 1 NY 0 NZ 0 LECT symx TERM
     CONT SPLA NX 0 NY 1 NZ 0 LECT symy TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY FROT MUST 0.15 MUDY 0.1 GAMM 0 MLEV 5
          LECT lframeb TERM
          BODY FROT MUST 0.15 MUDY 0.1 GAMM 0 MLEV 5
          LECT uframe TERM
          BODY FROT MUST 0.15 MUDY 0.1 GAMM 0 DIAM 0.0008
          LECT nplate TERM
ECRI DEPL VITE ECRO FAIL TFRE 1.E-3
          POIN LECT 1 TERM
          ELEM LECT 1 TERM
     FICH SPLI ALIC TFRE 0.1E-3
     FICH ALIC TEMP TFRE 0.01E-3
          POIN LECT p0 TERM
          ELEM LECT prpl TERM
OPTI NOTE CSTA 0.8
     LOG 1
     JAUM
     LMST
     PINS GRID DPIN 1.01
    !lnks stat
CALC TINI 0 TEND 10.0E-3
FIN
```

clam050.epx

```
CLAM050
ECHO
!CONV win
KFIL
TRID LAGR
GEOM CUB8 PART 1 PART 4 Q4GS PART 2 COQ4 PART 3 PART 5 TERM
COMP EPAI 0.8e-3 LECT PART 2 PART 3 PART 5 TERM
     GROU 5 'lframeb' LECT PART 1 TERM
            'plate'   LECT PART 2 TERM
            'preplat' LECT PART 3 TERM
            'uframe'  LECT PART 4 TERM
            'preclam' LECT PART 5 TERM
     NGRO 1 'bloc'    LECT NSET 1 TERM
MATE FANT 1.0 LECT tous TERM
OPTI NOTE CSTA 0.8
     K2MS MANU
CALC TINI 0 TEND 3.4E-3 NMAX 0
FIN
```

clam05a.epx

```
CLAM05A
ECHO
 CONV WIN
RESU SPLI ALIC 'clam05.ali' GARD PSCR
OPTI PRIN
SORT VISU NSTO 1
PLAY
CAME   1 EYE  -3.21616E-01 -3.21616E-01  8.21785E-01
!        Q     8.73545E-01  3.00786E-01 -1.24589E-01 -3.61834E-01
         VIEW  4.35338E-01  4.35339E-01 -7.88011E-01
         RIGH  7.07107E-01 -7.07107E-01 -4.22336E-07
         UP    5.57208E-01  5.57208E-01  6.15662E-01
         FOV   2.48819E+01
!NAVIGATION MODE: FREE CAMERA
!CENTER :  1.62500E-01  1.62500E-01  4.60000E-03
!RSPHERE:  2.31760E-01
!RADIUS :  1.06608E+00
!ASPECT :  1.00000E+00
!NEAR   :  8.33700E-01
!FAR    :  1.29722E+00
SCEN GEOM NAVI FREE
          FACE SBAC
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 101 FPS 15 KFRE 10 COMP -1 NFAI
!    OBJE LECT lframeb plate uframe TERM REND
     OBJE LECT lframeb plate TERM REND
GOTR LOOP 99 OFFS FICH AVI CONT NOCL NFAI
!    OBJE LECT lframeb plate uframe TERM REND
     OBJE LECT lframeb plate TERM REND
GO
TRAC OFFS FICH AVI CONT NFAI
!    OBJE LECT lframeb plate uframe TERM REND
     OBJE LECT lframeb plate TERM REND
ENDPLAY
FIN
```

clam05b.epx

```
CLAM05B
ECHO
RESU ALIC TEMP 'clam05.alt' GARD PSCR
SORT GRAP
AXTE 1.0 'Time [s]'
COUR 1 'DisCen' DEPL COMP 3 POIN LECT p0 TERM
TRAC 1 AXES 1.0 'DisCen' YZER
     COLO bleu
     THIC 0.8
LIST 1 AXES 1.0 'DisCen' YZER
RCOU 11 'DisCen' FICH 'clam01b.pun' RENA 'DisCenC01'
RCOU 21 'DisCen' FICH 'clam02b.pun' RENA 'DisCenC02'
RCOU 31 'DisCen' FICH 'stpm01b.pun' RENA 'DisCenST1'
COUR 51 'DisCenInv' MULC 31 -1.0
TRAC 11 21 51 1 AXES 1.0 'DisCen' YZER
COLO VERT ROUG TURQ BLEU
FIN
```

clam06.epx

```
CLAM06
ECHO
!CONV win
KFIL
EROS 1.0
TRID LAGR
GEOM CUB8 PART 1 PART 4 Q4GS PART 2 CL3D PART 3 PART 5 TERM
COMP EPAI 0.8e-3 LECT PART 2 TERM
     COUL TURQ LECT PART 1 TERM
          TURQ LECT PART 4 TERM
         VERT LECT PART 2 TERM
         ROSE LECT PART 5 TERM
     NGRO 3 'centd' LECT PART 2 TERM COND NEAR POIN 0.0 0.0 0.0
           'centl' LECT PART 3 TERM COND NEAR POIN 0.0 0.0 0.0
           'axis'  LECT PART 2 TERM
           COND LINE X1 -0.2 Y1 -0.001 Z1 -0.001
                             X2  0.2 Y2  0.001 Z2  0.001 TOL 1.E-5
     ORIE INVE LECT PART 3 TERM
INCLUDE 'p77_75.txt'
MATE VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 3.257E8 mxit 2000
         QR1 2.348E8 CR1 56.2 QR2 4.457E8 CR2 4.7
         PDOT 5.E-4 C 1.E-2 TQ 0.9 CP 452.0
         TM 1800.0 M 1.0 DC 1.0 WC 555.0E6
         LECT PART 1 PART 2 PART 4 TERM
     IMPE PIMP RO 7850.0 PRES 42.1e6 PREF 0
         LECT PART 5 TERM
     IMPE PIMP RO 7850 PRES 1.0 PREF 1.0E5 FONC 1
         LECT PART 3 TERM
LINK COUP
     BLOQ 123 LECT NSET 1 TERM
     GLIS 2 FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #1
           PGAP 0.4E-3
           MAIT LECT PART 1 TERM
           PESC LECT PART 2  TERM
           FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #2
           PGAP 0.4E-3
           MAIT LECT PART 4 TERM
           PESC LECT PART 2 TERM
ECRI DEPL VITE ECRO FAIL TFRE 1.E-3
         POIN LECT 1 TERM
         ELEM LECT 1 TERM
     FICH SPLI ALIC TFRE 0.1E-3
OPTI NOTE CSTA 0.8
     LOG 1
     JAUM
     LMST
CALC TINI 0 TEND 3.4E-3
FIN
```

clam06a.epx

```
CLAM06A
ECHO
 CONV WIN
RESU SPLI ALIC 'clam06.ali' GARD PSCR
COMP GROU 3 'one4' LECT PART 1 TERM
                  COND XB GT 0.0
                  COND YB GT 0.0
           'two4' LECT PART 2 TERM
                  COND XB GT 0.0
                  COND YB GT 0.0
           'fou4' LECT PART 4 TERM
                  COND XB GT 0.0
                  COND YB GT 0.0
OPTI PRIN
SORT VISU NSTO 1
PLAY
CAME   1 EYE  -3.21616E-01 -3.21616E-01  8.21785E-01
!        Q     8.73545E-01  3.00786E-01 -1.24589E-01 -3.61834E-01
         VIEW  4.35338E-01  4.35339E-01 -7.88011E-01
         RIGH  7.07107E-01 -7.07107E-01 -4.22336E-07
         UP    5.57208E-01  5.57208E-01  6.15662E-01
         FOV   2.48819E+01
!NAVIGATION MODE: FREE CAMERA
!CENTER :  1.62500E-01  1.62500E-01  4.60000E-03
!RSPHERE:  2.31760E-01
!RADIUS :  1.06608E+00
!ASPECT :  1.00000E+00
!NEAR   :  8.33700E-01
!FAR    :  1.29722E+00
SCEN GEOM NAVI FREE
          FACE SBAC
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 35 FPS 15 KFRE 10 COMP -1 NFAI
!    OBJE LECT one4 two4 fou4 TERM REND
     OBJE LECT one4 two4 TERM REND
GOTR LOOP 33 OFFS FICH AVI CONT NOCL NFAI
!    OBJE LECT one4 two4 fou4 TERM REND
     OBJE LECT one4 two4 TERM REND
GO
TRAC OFFS FICH AVI CONT NFAI
!    OBJE LECT one4 two4 fou4 TERM REND
     OBJE LECT one4 two4 TERM REND
ENDPLAY
FIN
```

clam06b.epx

```
CLAM06B
ECHO
RESU SPLI ALIC 'clam06.ali' GARD PSCR
COMP NGRO 1 'centd' LECT PART 2 TERM COND NEAR POIN 0.0 0.0 0.0
SORT GRAP
AXTE 1.0 'Time [s]'
COUR 1 'DisCen' DEPL COMP 3 POIN LECT centd TERM
TRAC 1 AXES 1.0 'DisCen' YZER
     COLO bleu
     THIC 0.8
LIST 1 AXES 1.0 'DisCen' YZER
FIN
```

fro100.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’fro100_mesh.ps’;
opti sauv form ’fro100.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

fro100.epx

```
FRO100
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY NOCG LECT circ TERM
          BODY NOCG LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB1
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

fro101.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’fro101_mesh.ps’;
opti sauv form ’fro101.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
```

```
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

fro101.epx

```
FRO101
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY NOCG FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY NOCG FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB1
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

fro102.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’fro102_mesh.ps’;
opti sauv form ’fro102.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
```

```
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

fro102.epx

```
FRO102
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY NOCG LECT circ TERM
          BODY NOCG LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
          MASL PAIR 2 1
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB1
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
         !LINE HEOU SSHA
         !LNKS SHOW GPIN JOIN
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

fro103.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’fro103_mesh.ps’;
opti sauv form ’fro103.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

fro103.epx

```
FRO103
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
```

```
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY NOCG FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY NOCG FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
          MASL PAIR 2 1
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB1
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

frot23.dgibi

```
opti echo 0;
opti donn ’px4cir2d.proc’;
opti donn ’pxordpoi.proc’;
opti echo 1;
opti dime 2 elem qua4;
opti trac psc ftra ’frot23_mesh.ps’;
opti sauv form ’frot23.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r;
n = 8;
tol = 0.001;
c1 ier = px4cir2d (x0 0) ((x0+r) r) pc n tol;
c2 = c1 tour 90 pc;
c3 = c2 tour 90 pc;
c4 = c3 tour 90 pc;
circ = c1 et c2 et c3 et c4;
elim tol circ;
p1 = x0 0;
p2 = (x0+r) r;
p3 = x0 (r+r);
p4 = (x0-r) r;
elim tol (circ et p1 et p2 et p3 et p4);
cc1 = cerc n p1 pc p2;
cc2 = cc1 tour 90 pc;
cc3 = cc2 tour 90 pc;
cc4 = cc3 tour 90 pc;
elim tol (circ et cc1 et cc2 et cc3 et cc4);
pc1 = pxordpoi (chan POI1 cc1) p1;
pc2 = pxordpoi (chan POI1 cc2) p2;
pc3 = pxordpoi (chan POI1 cc3) p3;
pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r);
elim tol (pm et circ);
mark = manu poi1 pm;
h = 3.0;
len = 10.0;
nh = enti (h / 1.0);
nb = enti (len / 1.0);
bas = (0 (0-h)) d nb (len (0-h));
base = bas tran nh (0 h);
bas2 = (len 0) d nb (0 0);
*list (nbno bas2);
elim tol (bas2 et base);
p5 = len 0;
elim tol (p5 et bas2);
pbas2 = chan POI1 bas2;
*list (nbno pbas2);
pb2 = pxordpoi pbas2 p5;
list (nbno pb2);
mesh = circ et base et pb2 et pc1 et pc2 et pc3 et pc4 et mark;
tass mesh noop;
sauv form mesh;
trac qual mesh;
fin;
```

frot23.epx

```
FROT23
ECHO
!CONV WIN
CAST mesh
LAGR DPLA
DIME GLIS 1 200 TERM
GEOM CAR4 circ base PMAT mark TERM
COMP EPAI 0.2 LECT mark TERM ! Only for visualization
     NGRO 1 ’b1’ LECT circ TERM COND NEAR POIN 3 0
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
!LINK COUP
LIAI
     BLOQ 12 LECT bas TERM
     GLIS 1
          MAIT LECT pb2 TERM
          ESCL LECT pc1 pc2 pc3 pc4 TERM
CHAR 1 FACT 2
     FORC 1  6.E8 LECT pc TERM
     FORC 2 -6.E8 LECT pc TERM
     TABL 3 0. 0.  6.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 10000
     FICH ALIC FREQ 5000
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 pc mark TERM
OPTI PAS UTIL
     LOG 1
     LNKS STAT
CALC TINI 0 TFIN 60.E-3 PASF 2.E-6 NMAX 30000
PLAY
CAME   1 EYE   5.00000E+00  5.50000E-01  3.06645E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  5.50000E-01  0.00000E+00
!RSPHERE:  6.13290E+00
!RADIUS :  3.06645E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.39183E+01
!FAR    :  4.29303E+01
SCEN GEOM NAVI FREE
          POIN SPHP
     VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
SLER CAM1 1 NFRA 1
FREQ 5000
TRAC OFFS FICH AVI NOCL NFTO 7 FPS 5 KFRE 5 COMP -1 REND
GOTR LOOP 5 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1. ’Time [s]’
```

```
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot24.dgibi

```
opti echo 0;
opti donn ’px4cir2d.proc’;
opti donn ’pxordpoi.proc’;
opti echo 1;
opti dime 2 elem qua4;
opti trac psc ftra ’frot24_mesh.ps’;
opti sauv form ’frot24.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r;
n = 8;
tol = 0.001;
c1 ier = px4cir2d (x0 0) ((x0+r) r) pc n tol;
c2 = c1 tour 90 pc;
c3 = c2 tour 90 pc;
c4 = c3 tour 90 pc;
circ = c1 et c2 et c3 et c4;
elim tol circ;
p1 = x0 0;
p2 = (x0+r) r;
p3 = x0 (r+r);
p4 = (x0-r) r;
elim tol (circ et p1 et p2 et p3 et p4);
cc1 = cerc n p1 pc p2;
cc2 = cc1 tour 90 pc;
cc3 = cc2 tour 90 pc;
cc4 = cc3 tour 90 pc;
elim tol (circ et cc1 et cc2 et cc3 et cc4);
pc1 = pxordpoi (chan POI1 cc1) p1;
pc2 = pxordpoi (chan POI1 cc2) p2;
pc3 = pxordpoi (chan POI1 cc3) p3;
pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r);
elim tol (pm et circ);
mark = manu poi1 pm;
h = 3.0;
len = 10.0;
nh = enti (h / 1.0);
nb = enti (len / 1.0);
bas = (0 (0-h)) d nb (len (0-h));
base = bas tran nh (0 h);
bas2 = (len 0) d nb (0 0);
*list (nbno bas2);
elim tol (bas2 et base);
p5 = len 0;
elim tol (p5 et bas2);
pbas2 = chan POI1 bas2;
*list (nbno pbas2);
pb2 = pxordpoi pbas2 p5;
list (nbno pb2);
mesh = circ et base et pb2 et pc1 et pc2 et pc3 et pc4 et mark;
tass mesh noop;
sauv form mesh;
trac qual mesh;
fin;
```

frot24.epx

```
FROT24
ECHO
!CONV WIN
CAST mesh
LAGR DPLA
DIME GLIS 1 200 TERM
GEOM CAR4 circ base PMAT mark TERM
COMP EPAI 0.2 LECT mark TERM ! Only for visualization
     NGRO 1 ’b1’ LECT circ TERM COND NEAR POIN 3 0
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
     FROT 1 MU0 0.25 MU1 0.25 GAMM 0.1
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
!LINK COUP
LIAI
     BLOQ 12 LECT bas TERM
     GLIS 1
          MAIT LECT pb2 TERM
          ESCL LECT pc1 pc2 pc3 pc4 TERM
CHAR 1 FACT 2
     FORC 1  6.E8 LECT pc TERM
     FORC 2 -6.E8 LECT pc TERM
     TABL 3 0. 0.  6.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 10000
     FICH ALIC FREQ 5000
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 pc mark TERM
OPTI PAS UTIL
     LOG 1
     LNKS STAT
CALC TINI 0 TFIN 60.E-3 PASF 2.E-6 NMAX 30000
PLAY
CAME   1 EYE   5.00000E+00  5.50000E-01  3.06645E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  5.50000E-01  0.00000E+00
!RSPHERE:  6.13290E+00
!RADIUS :  3.06645E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.39183E+01
!FAR    :  4.29303E+01
SCEN GEOM NAVI FREE
          POIN SPHP
     VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
SLER CAM1 1 NFRA 1
FREQ 5000
TRAC OFFS FICH AVI NOCL NFTO 7 FPS 5 KFRE 5 COMP -1 REND
GOTR LOOP 5 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1. ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot25.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot25_mesh.ps’;
opti sauv form ’frot25.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
```

```
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot25.epx

```
FROT25
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB6 circ base PMAT mark TERM
COMP NGRO 1 'bloc' LECT base TERM COND Y LT -0.99
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 1
          MAIT LECT circ TERM
          ESCL LECT base TERM
CHAR 1 FACT 2
     FORC 1  2.E9 LECT axe TERM
     FORC 2 -2.E9 LECT axe TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
OPTI PAS UTIL
     LOG 1
     LNKS STAT
CALC TINI 0 TFIN 24.E-3 PASF 2.E-5 NMAX 1200
PLAY
CAME   1 EYE   2.11849E+01  1.56070E+01  2.16885E+01
!        Q     9.17087E-01 -2.30200E-01  3.15900E-01  7.85263E-02
         VIEW -5.43262E-01 -4.71839E-01 -6.94431E-01
         RIGH  7.88082E-01 -1.40916E-03 -6.15569E-01
         UP   -2.89471E-01  8.81683E-01 -3.72613E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
     VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 100
TRAC OFFS FICH AVI NOCL NFTO 13 FPS 5 KFRE 5 COMP -1 REND
GOTR LOOP 11 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
FIN
```

frot26.dgibi

```
opti echo 0;
opti donn 'px4cir3d.proc';
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra 'frot26_mesh.ps';
opti sauv form 'frot26.msh';
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot26.epx

```
FROT26
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB6 circ base PMAT mark TERM
COMP NGRO 1 'bloc' LECT base TERM COND Y LT -0.99
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 1
          FROT MUST 0.25 MUDY 0.25 GAMM 0.1
          MAIT LECT circ TERM
          ESCL LECT base TERM
CHAR 1 FACT 2
     FORC 1  2.E9 LECT axe TERM
     FORC 2 -2.E9 LECT axe TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
OPTI PAS UTIL
     LOG 1
     LNKS STAT
CALC TINI 0 TFIN 24.E-3 PASF 2.E-5 NMAX 1200
PLAY
CAME   1 EYE   2.11849E+01  1.56070E+01  2.16885E+01
!        Q     9.17087E-01 -2.30200E-01  3.15900E-01  7.85263E-02
         VIEW -5.43262E-01 -4.71839E-01 -6.94431E-01
         RIGH  7.88082E-01 -1.40916E-03 -6.15569E-01
         UP   -2.89471E-01  8.81683E-01 -3.72613E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
```

```
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
     VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 100
TRAC OFFS FICH AVI NOCL NFTO 13 FPS 5 KFRE 5 COMP -1 REND
GOTR LOOP 11 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
FIN
```

frot27.dgibi

```
opti echo 0;
opti donn 'px4cir3d.proc';
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra 'frot27_mesh.ps';
opti sauv form 'frot27.msh';
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot27.epx

```
FROT27
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 1 'bloc' LECT base TERM COND Y LT -0.99
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 1
          MAIT LECT circ TERM
          ESCL LECT base TERM
CHAR 1 FACT 2
     FORC 1  2.E9 LECT axe TERM
     FORC 2 -2.E9 LECT axe TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
OPTI PAS UTIL
     LOG 1
     LNKS STAT
CALC TINI 0 TFIN 24.E-3 PASF 2.E-5 NMAX 1200
PLAY
CAME   1 EYE   2.11849E+01  1.56070E+01  2.16885E+01
!        Q     9.17087E-01 -2.30200E-01  3.15900E-01  7.85263E-02
         VIEW -5.43262E-01 -4.71839E-01 -6.94431E-01
         RIGH  7.88082E-01 -1.40916E-03 -6.15569E-01
         UP   -2.89471E-01  8.81683E-01 -3.72613E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
     VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 100
TRAC OFFS FICH AVI NOCL NFTO 13 FPS 5 KFRE 5 COMP -1 REND
GOTR LOOP 11 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
FIN
```

frot28.dgibi

```
opti echo 0;
opti donn 'px4cir3d.proc';
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra 'frot28_mesh.ps';
opti sauv form 'frot28.msh';
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
```

```
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot28.epx

```
FROT28
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 1 'bloc' LECT base TERM COND Y LT -0.99
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 1
          FROT MUST 0.25 MUDY 0.25 GAMM 0.1
          MAIT LECT circ TERM
          ESCL LECT base TERM
CHAR 1 FACT 2
     FORC 1  2.E9 LECT axe TERM
     FORC 2 -2.E9 LECT axe TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
OPTI PAS UTIL
     LOG 1
     LNKS STAT
CALC TINI 0 TFIN 24.E-3 PASF 2.E-5 NMAX 1200
PLAY
CAME   1 EYE   2.11849E+01  1.56070E+01  2.16885E+01
!        Q     9.17087E-01 -2.30200E-01  3.15900E-01  7.85263E-02
         VIEW -5.43262E-01 -4.71839E-01 -6.94431E-01
         RIGH  7.88082E-01 -1.40916E-03 -6.15569E-01
         UP   -2.89471E-01  8.81683E-01 -3.72613E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
     VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 100
TRAC OFFS FICH AVI NOCL NFTO 13 FPS 5 KFRE 5 COMP -1 REND
GOTR LOOP 11 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
FIN
```

frot58.dgibi

```
opti echo 0;
opti donn 'px4cir3d.proc';
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra 'frot58_mesh.ps';
opti sauv form 'frot58.msh';
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot58.epx

```
FROT58
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 1 'bloc' LECT base TERM COND Y LT -0.99
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 1
          MAIT LECT circ TERM
          ESCL LECT base TERM
CHAR 1 FACT 2
     FORC 1  2.E9 LECT axe TERM
     FORC 2 -2.E9 LECT axe TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
OPTI PAS UTIL
     LOG 1
     LNKS STAT VISU
CALC TINI 0 TFIN 24.E-3 PASF 2.E-5 NMAX 1200
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GLIS JOIN
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 1201 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1199 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
FIN
```

frot59.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot59_mesh.ps’;
opti sauv form ’frot59.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot59.epx

```
FROT59
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 1 ’bloc’ LECT base TERM COND Y LT -0.99
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 1
          MAIT LECT circ TERM
          ESCL LECT base TERM
CHAR 1 FACT 2
     FORC 1  2.E9 LECT axe TERM
     FORC 2 -2.E9 LECT axe TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
OPTI PAS UTIL
     LOG 1
     GLIS NORM ELEM
     LNKS STAT VISU
CALC TINI 0 TFIN 24.E-3 PASF 2.E-5 NMAX 1200
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GLIS JOIN
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 1201 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1199 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
FIN
```

frot60.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot60_mesh.ps’;
opti sauv form ’frot60.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot60.epx

```
FROT60
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 1 ’bloc’ LECT base TERM COND Y LT -0.99
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 1 ELIM
          MAIT LECT circ TERM
          ESCL LECT base TERM
CHAR 1 FACT 2
     FORC 1  2.E9 LECT axe TERM
     FORC 2 -2.E9 LECT axe TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
OPTI PAS UTIL
     LOG 1
     GLIS NORM ELEM
     LNKS STAT VISU
CALC TINI 0 TFIN 24.E-3 PASF 2.E-5 NMAX 1200
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GLIS JOIN
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 1201 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1199 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
FIN
```

frot65.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot65_mesh.ps’;
opti sauv form ’frot65.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot65.epx

```
FROT65
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 9 ’bloc’ LECT base TERM COND Y LT -0.99
            ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
            ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
            ’f1’   LECT f1a f1b TERM
            ’f2’   LECT axe DIFF f1 TERM
            ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
            ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
            ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
            ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 1 ELIM
          MAIT LECT circ TERM
          ESCL LECT base TERM
CHAR 1 FACT 2
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GLIS NORM ELEM
     LNKS STAT VISU
CALC TINI 0 TFIN 24.E-3 PASF 2.E-5 NMAX 1200
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
        ! LINE HEOU SSHA
        ! LNKS SHOW GLIS JOIN
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 1201 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1199 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
```

```
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

frot66.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot66_mesh.ps’;
opti sauv form ’frot66.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot66.epx

```
FROT66
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 9 ’bloc’ LECT base TERM COND Y LT -0.99
            ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
            ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
            ’f1’   LECT f1a f1b TERM
            ’f2’   LECT axe DIFF f1 TERM
            ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
            ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
            ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
            ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 1 ELIM
          MAIT LECT circ TERM
          ESCL LECT base TERM
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GLIS NORM ELEM
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
        ! LINE HEOU SSHA
        ! LNKS SHOW GLIS JOIN
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

frot67.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot67_mesh.ps’;
opti sauv form ’frot67.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
```

```
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot67.epx

```
FROT67
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 9 ’bloc’ LECT base TERM COND Y LT -0.99
            ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
            ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
            ’f1’   LECT f1a f1b TERM
            ’f2’   LECT axe DIFF f1 TERM
            ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
            ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
            ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
            ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 1 ELIM
          FROT MUST 0.25 MUDY 0.25 GAMM 0.1
          MAIT LECT circ TERM
          ESCL LECT base TERM
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GLIS NORM ELEM
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
        ! LINE HEOU SSHA
        ! LNKS SHOW GLIS JOIN
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

frot68.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot68_mesh.ps’;
opti sauv form ’frot68.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
depl circ plus (2 0 0);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
```

```
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot68.epx

```
FROT68
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 9 ’bloc’ LECT base TERM COND Y LT -0.99
            ’f1a’  LECT axe TERM COND NEAR POIN 5 2 0
            ’f1b’  LECT axe TERM COND NEAR POIN 5 2 2
            ’f1’   LECT f1a f1b TERM
            ’f2’   LECT axe DIFF f1 TERM
            ’b1’   LECT circ TERM COND NEAR POIN 5 0 0
            ’b2’   LECT circ TERM COND NEAR POIN 5 0 0.667
            ’b3’   LECT circ TERM COND NEAR POIN 5 0 1.333
            ’b4’   LECT circ TERM COND NEAR POIN 5 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 1 ELIM
          MAIT LECT circ TERM
          ESCL LECT base TERM
CHAR 1 FACT 2
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GLIS NORM ELEM
     LNKS STAT VISU
CALC TINI 0 TFIN 24.E-3 PASF 2.E-5 NMAX 1200
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
        ! LINE HEOU SSHA
        ! LNKS SHOW GLIS JOIN
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 1201 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1199 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
```

```
FIN
```

frot69.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot69_mesh.ps’;
opti sauv form ’frot69.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot69.epx

```
FROT69
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 9 ’bloc’ LECT base TERM COND Y LT -0.99
            ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
            ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
            ’f1’   LECT f1a f1b TERM
            ’f2’   LECT axe DIFF f1 TERM
            ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
            ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
            ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
            ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GLIS 2 ELIM
          FROT MUST 0.25 MUDY 0.25 GAMM 0.1
          MAIT LECT circ TERM
```

```
          ESCL LECT base TERM
            ELIM
          FROT MUST 0.25 MUDY 0.25 GAMM 0.1
          MAIT LECT base TERM
          ESCL LECT circ TERM
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GLIS NORM ELEM
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
        ! LINE HEOU SSHA
        ! LNKS SHOW GLIS JOIN
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 12 ’vy_b2’  VITE COMP 2 NOEU LECT b2 TERM
COUR 13 ’vy_b3’  VITE COMP 2 NOEU LECT b3 TERM
COUR 14 ’vy_b4’  VITE COMP 2 NOEU LECT b4 TERM
COUR 15 ’vy_f1b’ VITE COMP 2 NOEU LECT f1b TERM
COUR 16 ’vy_f1a’ VITE COMP 2 NOEU LECT f1a TERM
COUR 17 ’vx_f1b’ VITE COMP 1 NOEU LECT f1b TERM
COUR 18 ’vx_f1a’ VITE COMP 1 NOEU LECT f1a TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 12 13 14 15 16 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 16 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 16 17 18 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot70.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot70_mesh.ps’;
opti sauv form ’frot70.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot70.epx

```
FROT70
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY LECT circ TERM
          BODY LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
CHAR 1 FACT 2
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB1
     LNKS STAT VISU
CALC TINI 0 TFIN 24.E-3 PASF 2.E-5 NMAX 1200
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
```

```
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 1201 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1199 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

frot71.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot71_mesh.ps’;
opti sauv form ’frot71.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot71.epx

```
FROT71
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY LECT circ TERM
          BODY LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB1
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
```

```
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

frot72.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot72_mesh.ps’;
opti sauv form ’frot72.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot72.epx

```
FROT72
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB1
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

frot72a.epx

```
FROT72A
ECHO
RESU ALIC ’frot72.ali’ GARD PSCR
SORT VISU NSTO 1
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
```

```
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
        ! LINE HEOU SSHA
        ! LNKS SHOW GLIS JOIN
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH BMP OBJE LECT circ mark TERM REND
GOTR LOOP 19 OFFS FICH BMP OBJE LECT circ mark TERM REND
GO
TRAC OFFS FICH BMP OBJE LECT circ mark TERM REND
ENDPLAY
FIN
```

frot72b.epx

```
FROT72B
ECHO
RESU ALIC 'frot72.ali' GARD PSCR
SORT VISU NSTO 1
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
        ! LINE HEOU SSHA
        ! LNKS SHOW GLIS JOIN
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH BMP OBJE LECT base TERM REND
GOTR LOOP 19 OFFS FICH BMP OBJE LECT base TERM REND
GO
TRAC OFFS FICH BMP OBJE LECT base TERM REND
ENDPLAY
FIN
```

frot73.dgibi

```
opti echo 0;
opti donn 'px4cir3d.proc';
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra 'frot73_mesh.ps';
opti sauv form 'frot73.msh';
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
depl circ plus (2 0 0);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot73.epx

```
FROT73
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 'bloc' LECT base TERM COND Y LT -0.99
             'basu' LECT base TERM cond Y GT -0.01
             'ncy1' LECT circ TERM
                   COND CYLI X1 5.0 Y1 2.0 Z1 -0.01
                             X2 5.0 Y2 2.0 Z2  2.01 R 2.01
             'ncy2' LECT circ TERM
                   COND CYLI X1 5.0 Y1 2.0 Z1 -0.01
                             X2 5.0 Y2 2.0 Z2  2.01 R 1.99
             'ncyl' LECT ncy1 DIFF ncy2 TERM
             'f1a'  LECT axe TERM COND NEAR POIN 5 2 0
             'f1b'  LECT axe TERM COND NEAR POIN 5 2 2
             'f1'   LECT f1a f1b TERM
             'f2'   LECT axe DIFF f1 TERM
             'b1'   LECT circ TERM COND NEAR POIN 5 0 0
             'b2'   LECT circ TERM COND NEAR POIN 5 0 0.667
             'b3'   LECT circ TERM COND NEAR POIN 5 0 1.333
             'b4'   LECT circ TERM COND NEAR POIN 5 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY LECT circ TERM
          BODY LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
CHAR 1 FACT 2
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS REB1
     LNKS STAT VISU
CALC TINI 0 TFIN 24.E-3 PASF 2.E-5 NMAX 1200
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 1201 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1199 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
```

```
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

frot74.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot74_mesh.ps’;
opti sauv form ’frot74.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot74.epx

```
FROT74
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB1 REBC
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

frot75.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot75_mesh.ps’;
```

```
opti sauv form ’frot75.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot75.epx

```
FROT75
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB2
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 1275 ! 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
!TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
!GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
TRAC OFFS FICH AVI NOCL NFTO 1276 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1274 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

frot76.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot76_mesh.ps’;
opti sauv form ’frot76.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
```

```
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot76.epx

```
FROT76
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.20 LECT ncyl TERM
          DIAM 0.40 LECT basu TERM
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB2
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 12 ’vy_b2’  VITE COMP 2 NOEU LECT b2 TERM
COUR 13 ’vy_b3’  VITE COMP 2 NOEU LECT b3 TERM
COUR 14 ’vy_b4’  VITE COMP 2 NOEU LECT b4 TERM
COUR 15 ’vy_f1b’ VITE COMP 2 NOEU LECT f1b TERM
COUR 16 ’vy_f1a’ VITE COMP 2 NOEU LECT f1a TERM
COUR 17 ’vx_f1b’ VITE COMP 1 NOEU LECT f1b TERM
COUR 18 ’vx_f1a’ VITE COMP 1 NOEU LECT f1a TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 12 13 14 15 16 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 16 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 16 17 18 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot77.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot77_mesh.ps’;
opti sauv form ’frot77.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
```

```
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot77.epx

```
FROT77
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.10 LECT ncyl basu TERM
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB2
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 12 ’vy_b2’  VITE COMP 2 NOEU LECT b2 TERM
COUR 13 ’vy_b3’  VITE COMP 2 NOEU LECT b3 TERM
COUR 14 ’vy_b4’  VITE COMP 2 NOEU LECT b4 TERM
COUR 15 ’vy_f1b’ VITE COMP 2 NOEU LECT f1b TERM
COUR 16 ’vy_f1a’ VITE COMP 2 NOEU LECT f1a TERM
COUR 17 ’vx_f1b’ VITE COMP 1 NOEU LECT f1b TERM
COUR 18 ’vx_f1a’ VITE COMP 1 NOEU LECT f1a TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 12 13 14 15 16 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 16 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 16 17 18 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot78.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot78_mesh.ps’;
opti sauv form ’frot78.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot78.epx

```
FROT78
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
```

```
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.20 LECT ncyl basu TERM
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB2
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          LINE HEOU SSHA
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 12 ’vy_b2’  VITE COMP 2 NOEU LECT b2 TERM
COUR 13 ’vy_b3’  VITE COMP 2 NOEU LECT b3 TERM
COUR 14 ’vy_b4’  VITE COMP 2 NOEU LECT b4 TERM
COUR 15 ’vy_f1b’ VITE COMP 2 NOEU LECT f1b TERM
COUR 16 ’vy_f1a’ VITE COMP 2 NOEU LECT f1a TERM
COUR 17 ’vx_f1b’ VITE COMP 1 NOEU LECT f1b TERM
COUR 18 ’vx_f1a’ VITE COMP 1 NOEU LECT f1a TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 12 13 14 15 16 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 16 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 16 17 18 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot81.dgibi

```
opti echo 0;
opti donn ’px4cir2d.proc’;
opti donn ’pxordpoi.proc’;
opti echo 1;
opti dime 2 elem qua4;
opti trac psc ftra ’frot81_mesh.ps’;
opti sauv form ’frot81.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r;
n = 8;
tol = 0.001;
c1 ier = px4cir2d (x0 0) ((x0+r) r) pc n tol;
c2 = c1 tour 90 pc;
c3 = c2 tour 90 pc;
c4 = c3 tour 90 pc;
circ = c1 et c2 et c3 et c4;
elim tol circ;
pm = pc plus (0 r);
elim tol (pm et circ);
mark = manu poi1 pm;
h = 3.0;
len = 10.0;
nh = enti (h / 1.0);
nb = enti (len / 1.0);
bas = (0 (0-h)) d nb (len (0-h));
base = bas tran nh (0 h);
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac qual mesh;
fin;
```

frot81.epx

```
FROT81
ECHO
!CONV WIN
CAST mesh
LAGR DPLA
GEOM CAR4 circ base PMAT mark TERM
COMP EPAI 0.2 LECT mark TERM ! Only for visualization
     NGRO 3 ’cir’   LECT circ TERM COND ENVE
            ’basup’ LECT base TERM COND Y GT -0.01
            ’b1’    LECT circ TERM COND NEAR POIN 3 0
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 12 LECT bas TERM
     GPIN BODY LECT circ TERM
          BODY LECT base TERM
          DIAM 0.05 LECT cir basup TERM
CHAR 1 FACT 2
     FORC 1  6.E8 LECT pc TERM
     FORC 2 -6.E8 LECT pc TERM
     TABL 3 0. 0.  6.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 1000
     FICH ALIC FREQ 150
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 pc mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT
     LNKS STAT VISU
CALC TINI 0 TFIN 60.E-3 PASF 2.E-5 NMAX 3000
PLAY
CAME   1 EYE   5.00000E+00  5.50000E-01  3.06645E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  5.50000E-01  0.00000E+00
!RSPHERE:  6.13290E+00
!RADIUS :  3.06645E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.39183E+01
!FAR    :  4.29303E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          !LINE HEOU SFRE
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 3001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 2999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1. ’Time [s]’
```

```
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot81a.epx

```
FROT81A
ECHO
!CONV WIN
RESU ALIC TEMP ’frot81.alt’ GARD PSCR
SORT GRAP FENE 0.0 40.E-3
AXTE 1. ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot82.dgibi

```
opti echo 0;
opti donn ’px4cir2d.proc’;
opti donn ’pxordpoi.proc’;
opti echo 1;
opti dime 2 elem qua4;
opti trac psc ftra ’frot82_mesh.ps’;
opti sauv form ’frot82.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r;
n = 8;
tol = 0.001;
c1 ier = px4cir2d (x0 0) ((x0+r) r) pc n tol;
c2 = c1 tour 90 pc;
c3 = c2 tour 90 pc;
c4 = c3 tour 90 pc;
circ = c1 et c2 et c3 et c4;
elim tol circ;
pm = pc plus (0 r);
elim tol (pm et circ);
mark = manu poi1 pm;
h = 3.0;
len = 10.0;
nh = enti (h / 1.0);
nb = enti (len / 1.0);
bas = (0 (0-h)) d nb (len (0-h));
base = bas tran nh (0 h);
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac qual mesh;
fin;
```

frot82.epx

```
FROT82
ECHO
!CONV WIN
CAST mesh
LAGR DPLA
GEOM CAR4 circ base PMAT mark TERM
COMP EPAI 0.2 LECT mark TERM ! Only for visualization
     NGRO 3 ’cir’   LECT circ TERM COND ENVE
            ’basup’ LECT base TERM COND Y GT -0.01
            ’b1’    LECT circ TERM COND NEAR POIN 3 0
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 12 LECT bas TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT cir basup TERM
CHAR 1 FACT 2
     FORC 1  6.E8 LECT pc TERM
     FORC 2 -6.E8 LECT pc TERM
     TABL 3 0. 0.  6.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 1000
     FICH ALIC FREQ 150
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 pc mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT
     LNKS STAT VISU
CALC TINI 0 TFIN 60.E-3 PASF 2.E-5 NMAX 3000
PLAY
CAME   1 EYE   5.00000E+00  5.50000E-01  3.06645E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  5.50000E-01  0.00000E+00
!RSPHERE:  6.13290E+00
!RADIUS :  3.06645E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.39183E+01
!FAR    :  4.29303E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          !LINE HEOU SFRE
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 3001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 2999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1. ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot82a.epx

```
FROT82A
ECHO
!CONV WIN
RESU ALIC TEMP ’frot82.alt’ GARD PSCR
SORT GRAP FENE 0.0 40.E-3
AXTE 1. ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot83.dgibi

```
opti echo 0;
opti donn ’px4cir2d.proc’;
opti donn ’pxordpoi.proc’;
opti echo 1;
opti dime 2 elem qua4;
opti trac psc ftra ’frot83_mesh.ps’;
opti sauv form ’frot83.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r;
n = 8;
tol = 0.001;
c1 ier = px4cir2d (x0 0) ((x0+r) r) pc n tol;
c2 = c1 tour 90 pc;
c3 = c2 tour 90 pc;
c4 = c3 tour 90 pc;
circ = c1 et c2 et c3 et c4;
elim tol circ;
pm = pc plus (0 r);
elim tol (pm et circ);
mark = manu poi1 pm;
h = 1.0;
len = 10.0;
nh = enti (h / 0.5);
nb = enti (len / 1.0);
bas = (0 (0-h)) d nb (len (0-h));
base = bas tran nh (0 h);
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac qual mesh;
fin;
```

frot83.epx

```
FROT83
ECHO
!CONV WIN
CAST mesh
LAGR CPLA
GEOM CAR4 circ base PMAT mark TERM
COMP EPAI 0.2 LECT mark TERM ! Only for visualization
     NGRO 3 ’cir’   LECT circ TERM COND ENVE
            ’basup’ LECT base TERM COND Y GT -0.01
            ’b1’    LECT circ TERM COND NEAR POIN 3 0
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 12 LECT bas TERM
     GPIN BODY LECT circ TERM
          BODY LECT base TERM
          DIAM 0.05 LECT cir basup TERM
CHAR 1 FACT 2
     FORC 1  1.E9 LECT pc TERM
     FORC 2 -1.E9 LECT pc TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 1000
     FICH ALIC FREQ 150
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 pc mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   5.00000E+00  1.55000E+00  2.80680E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  0.00000E+00
!RSPHERE:  5.61360E+00
!RADIUS :  2.80680E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.18930E+01
!FAR    :  3.92952E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          !LINE HEOU SFRE
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1. ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot83_gpn.epx

```
FROT83_GPN
ECHO
*CONV WIN
LAGR CPLA
GEOM LIBR POIN 4 CAR1 1 TERM
   0 0  1 0  1 1  0 1
   1 2 3 4
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT 1 TERM
ECRI FICH ALIC FREQ 1
CALC TINI 0.0 TEND 1.0 NMAX 0
*=================================================================
SUIT
Post-treatment
ECHO
RESU ALIC GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
RCOU  1 ’n_raw’   FICH ’frot83_gpn.pun’
RCOU  2 ’n_rebo’  FICH ’frot83_gpn.pun’
RCOU  3 ’penemx’  FICH ’frot83_gpn.pun’
RCOU  4 ’penemax’ FICH ’frot83_gpn.pun’
RCOU  5 ’pdotmx’  FICH ’frot83_gpn.pun’
RCOU  6 ’pdotmax’ FICH ’frot83_gpn.pun’
TRAC  1 AXES 1.0 ’N_RAW’ YZER ! XGRD YGRD
TRAC  2 AXES 1.0 ’N_REBO’ YZER ! XGRD YGRD
TRAC  3 AXES 1.0 ’PENEMX [M]’ YZER ! XGRD YGRD
TRAC  4 AXES 1.0 ’PENEMAX [M]’ YZER ! XGRD YGRD
TRAC  5 AXES 1.0 ’PDOTMX [M/S]’ YZER ! XGRD YGRD
TRAC  6 AXES 1.0 ’PDOTMAX [M/S]’ YZER ! XGRD YGRD
TRAC 3 4 AXES 1.0 ’PENE [M]’ YZER ! XGRD YGRD
COLO NOIR ROUG
TRAC 5 6 AXES 1.0 ’PDOT [M/S]’ YZER ! XGRD YGRD
COLO NOIR ROUG
*=================================================================
FIN
```

frot84.dgibi

```
opti echo 0;
opti donn ’px4cir2d.proc’;
opti donn ’pxordpoi.proc’;
opti echo 1;
opti dime 2 elem qua4;
opti trac psc ftra ’frot84_mesh.ps’;
opti sauv form ’frot84.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r;
n = 8;
tol = 0.001;
c1 ier = px4cir2d (x0 0) ((x0+r) r) pc n tol;
c2 = c1 tour 90 pc;
c3 = c2 tour 90 pc;
c4 = c3 tour 90 pc;
circ = c1 et c2 et c3 et c4;
elim tol circ;
pm = pc plus (0 r);
elim tol (pm et circ);
mark = manu poi1 pm;
h = 1.0;
len = 10.0;
nh = enti (h / 0.5);
nb = enti (len / 1.0);
bas = (0 (0-h)) d nb (len (0-h));
base = bas tran nh (0 h);
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac qual mesh;
fin;
```

frot84.epx

```
FROT84
ECHO
!CONV WIN
CAST mesh
LAGR CPLA
GEOM CAR4 circ base PMAT mark TERM
COMP EPAI 0.2 LECT mark TERM ! Only for visualization
     NGRO 3 ’cir’   LECT circ TERM COND ENVE
            ’basup’ LECT base TERM COND Y GT -0.01
            ’b1’    LECT circ TERM COND NEAR POIN 3 0
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 12 LECT bas TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT cir basup TERM
CHAR 1 FACT 2
     FORC 1  1.E9 LECT pc TERM
     FORC 2 -1.E9 LECT pc TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 1000
     FICH ALIC FREQ 150
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 pc mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   5.00000E+00  1.55000E+00  2.80680E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  0.00000E+00
!RSPHERE:  5.61360E+00
!RADIUS :  2.80680E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.18930E+01
!FAR    :  3.92952E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          !LINE HEOU SFRE
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1. ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot84_gpn.epx

```
FROT84_GPN
ECHO
*CONV WIN
LAGR CPLA
GEOM LIBR POIN 4 CAR1 1 TERM
   0 0  1 0  1 1  0 1
   1 2 3 4
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT 1 TERM
ECRI FICH ALIC FREQ 1
CALC TINI 0.0 TEND 1.0 NMAX 0
*=================================================================
SUIT
Post-treatment
ECHO
RESU ALIC GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
RCOU  1 ’n_raw’   FICH ’frot84_gpn.pun’
RCOU  2 ’n_rebo’  FICH ’frot84_gpn.pun’
RCOU  3 ’penemx’  FICH ’frot84_gpn.pun’
RCOU  4 ’penemax’ FICH ’frot84_gpn.pun’
RCOU  5 ’pdotmx’  FICH ’frot84_gpn.pun’
RCOU  6 ’pdotmax’ FICH ’frot84_gpn.pun’
TRAC  1 AXES 1.0 ’N_RAW’ YZER ! XGRD YGRD
TRAC  2 AXES 1.0 ’N_REBO’ YZER ! XGRD YGRD
TRAC  3 AXES 1.0 ’PENEMX [M]’ YZER ! XGRD YGRD
TRAC  4 AXES 1.0 ’PENEMAX [M]’ YZER ! XGRD YGRD
TRAC  5 AXES 1.0 ’PDOTMX [M/S]’ YZER ! XGRD YGRD
TRAC  6 AXES 1.0 ’PDOTMAX [M/S]’ YZER ! XGRD YGRD
TRAC 3 4 AXES 1.0 ’PENE [M]’ YZER ! XGRD YGRD
COLO NOIR ROUG
TRAC 5 6 AXES 1.0 ’PDOT [M/S]’ YZER ! XGRD YGRD
COLO NOIR ROUG
*=================================================================
FIN
```

frot85.dgibi

```
opti echo 0;
opti donn ’px4cir2d.proc’;
opti donn ’pxordpoi.proc’;
opti echo 1;
opti dime 2 elem qua4;
opti trac psc ftra ’frot85_mesh.ps’;
opti sauv form ’frot85.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r;
n = 8;
tol = 0.001;
c1 ier = px4cir2d (x0 0) ((x0+r) r) pc n tol;
c2 = c1 tour 90 pc;
c3 = c2 tour 90 pc;
c4 = c3 tour 90 pc;
circ = c1 et c2 et c3 et c4;
elim tol circ;
pm = pc plus (0 r);
elim tol (pm et circ);
mark = manu poi1 pm;
h = 1.0;
len = 10.0;
nh = enti (h / 0.5);
nb = enti (len / 1.0);
bas = (0 (0-h)) d nb (len (0-h));
base = bas tran nh (0 h);
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac qual mesh;
fin;
```

frot85.epx

```
FROT85
ECHO
!CONV WIN
CAST mesh
LAGR CPLA
GEOM CAR4 circ base PMAT mark TERM
opti dump dpma
COMP EPAI 0.2 LECT mark TERM ! Only for visualization
     NGRO 3 ’cir’   LECT circ TERM COND ENVE
            ’basup’ LECT base TERM COND Y GT -0.01
            ’b1’    LECT circ TERM COND NEAR POIN 3 0
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 12 LECT bas TERM
     GPIN BODY LECT circ TERM
          BODY LECT base TERM
          DIAM 0.05 LECT cir basup TERM
          MASL PAIR 2 1
CHAR 1 FACT 2
     FORC 1  1.E9 LECT pc TERM
     FORC 2 -1.E9 LECT pc TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT ACCE FLIA NUPA LECT 1432 PAS 1 1435 TERM
                                 ! FREQ 1000
          poin lect 238 239 240 200 25 24 term noel
     FICH ALIC FREQ 150
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 pc mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS DUMP STAT
     LNKS DUMP STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   5.00000E+00  1.55000E+00  2.80680E+01
```

```
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  0.00000E+00
!RSPHERE:  5.61360E+00
!RADIUS :  2.80680E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.18930E+01
!FAR    :  3.92952E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          !LINE HEOU SFRE
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1
     !OBJE LECT circ mark TERM
     REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL
     !OBJE LECT circ mark TERM
     REND
GO
TRAC OFFS FICH AVI CONT
     !OBJE LECT circ mark TERM
     REND
ENDPLAY
SUIT
Post-treatment
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1. ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot85_gpn.epx

```
FROT85_GPN
ECHO
*CONV WIN
LAGR CPLA
GEOM LIBR POIN 4 CAR1 1 TERM
   0 0  1 0  1 1  0 1
   1 2 3 4
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT 1 TERM
ECRI FICH ALIC FREQ 1
CALC TINI 0.0 TEND 1.0 NMAX 0
*=================================================================
SUIT
Post-treatment
ECHO
RESU ALIC GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
RCOU  1 ’n_raw’   FICH ’frot85_gpn.pun’
RCOU  2 ’n_rebo’  FICH ’frot85_gpn.pun’
RCOU  3 ’penemx’  FICH ’frot85_gpn.pun’
RCOU  4 ’penemax’ FICH ’frot85_gpn.pun’
RCOU  5 ’pdotmx’  FICH ’frot85_gpn.pun’
RCOU  6 ’pdotmax’ FICH ’frot85_gpn.pun’
TRAC  1 AXES 1.0 ’N_RAW’ YZER ! XGRD YGRD
TRAC  2 AXES 1.0 ’N_REBO’ YZER ! XGRD YGRD
TRAC  3 AXES 1.0 ’PENEMX [M]’ YZER ! XGRD YGRD
TRAC  4 AXES 1.0 ’PENEMAX [M]’ YZER ! XGRD YGRD
TRAC  5 AXES 1.0 ’PDOTMX [M/S]’ YZER ! XGRD YGRD
TRAC  6 AXES 1.0 ’PDOTMAX [M/S]’ YZER ! XGRD YGRD
TRAC 3 4 AXES 1.0 ’PENE [M]’ YZER ! XGRD YGRD
COLO NOIR ROUG
TRAC 5 6 AXES 1.0 ’PDOT [M/S]’ YZER ! XGRD YGRD
COLO NOIR ROUG
*=================================================================
FIN
```

frot85a.epx

```
FROT85A
ECHO
!CONV WIN
RESU ALIC TEMP ’frot85.alt’ GARD PSCR
SORT GRAP FENE 0.0 40.E-3
AXTE 1. ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot86.dgibi

```
opti echo 0;
opti donn ’px4cir2d.proc’;
opti donn ’pxordpoi.proc’;
opti echo 1;
opti dime 2 elem qua4;
opti trac psc ftra ’frot86_mesh.ps’;
opti sauv form ’frot86.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r;
n = 8;
tol = 0.001;
c1 ier = px4cir2d (x0 0) ((x0+r) r) pc n tol;
c2 = c1 tour 90 pc;
c3 = c2 tour 90 pc;
c4 = c3 tour 90 pc;
circ = c1 et c2 et c3 et c4;
elim tol circ;
pm = pc plus (0 r);
elim tol (pm et circ);
mark = manu poi1 pm;
h = 1.0;
len = 10.0;
nh = enti (h / 0.5);
nb = enti (len / 1.0);
bas = (0 (0-h)) d nb (len (0-h));
base = bas tran nh (0 h);
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac qual mesh;
fin;
```

frot86.epx

```
FROT86
ECHO
!CONV WIN
CAST mesh
LAGR CPLA
GEOM CAR4 circ base PMAT mark TERM
opti dump dpma
COMP EPAI 0.2 LECT mark TERM ! Only for visualization
     NGRO 3 ’cir’   LECT circ TERM COND ENVE
            ’basup’ LECT base TERM COND Y GT -0.01
            ’b1’    LECT circ TERM COND NEAR POIN 3 0
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 12 LECT bas TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT cir basup TERM
          MASL PAIR 2 1
CHAR 1 FACT 2
     FORC 1  1.E9 LECT pc TERM
     FORC 2 -1.E9 LECT pc TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT ACCE FLIA NUPA LECT 1432 PAS 1 1435 TERM
                                 ! FREQ 1000
          poin lect 238 239 240 200 25 24 term noel
     FICH ALIC FREQ 150
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 pc mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS DUMP STAT
     LNKS DUMP STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   5.00000E+00  1.55000E+00  2.80680E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
```

```
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  0.00000E+00
!RSPHERE:  5.61360E+00
!RADIUS :  2.80680E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.18930E+01
!FAR    :  3.92952E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          !LINE HEOU SFRE
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1
     !OBJE LECT circ mark TERM
     REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL
     !OBJE LECT circ mark TERM
     REND
GO
TRAC OFFS FICH AVI CONT
     !OBJE LECT circ mark TERM
     REND
ENDPLAY
SUIT
Post-treatment
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1. ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot86_gpn.epx

```
FROT86_GPN
ECHO
*CONV WIN
LAGR CPLA
GEOM LIBR POIN 4 CAR1 1 TERM
   0 0  1 0  1 1  0 1
   1 2 3 4
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT 1 TERM
ECRI FICH ALIC FREQ 1
CALC TINI 0.0 TEND 1.0 NMAX 0
*=================================================================
SUIT
Post-treatment
ECHO
RESU ALIC GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
RCOU  1 ’n_raw’   FICH ’frot86_gpn.pun’
RCOU  2 ’n_rebo’  FICH ’frot86_gpn.pun’
RCOU  3 ’penemx’  FICH ’frot86_gpn.pun’
RCOU  4 ’penemax’ FICH ’frot86_gpn.pun’
RCOU  5 ’pdotmx’  FICH ’frot86_gpn.pun’
RCOU  6 ’pdotmax’ FICH ’frot86_gpn.pun’
TRAC  1 AXES 1.0 ’N_RAW’ YZER ! XGRD YGRD
TRAC  2 AXES 1.0 ’N_REBO’ YZER ! XGRD YGRD
TRAC  3 AXES 1.0 ’PENEMX [M]’ YZER ! XGRD YGRD
TRAC  4 AXES 1.0 ’PENEMAX [M]’ YZER ! XGRD YGRD
TRAC  5 AXES 1.0 ’PDOTMX [M/S]’ YZER ! XGRD YGRD
TRAC  6 AXES 1.0 ’PDOTMAX [M/S]’ YZER ! XGRD YGRD
TRAC 3 4 AXES 1.0 ’PENE [M]’ YZER ! XGRD YGRD
COLO NOIR ROUG
TRAC 5 6 AXES 1.0 ’PDOT [M/S]’ YZER ! XGRD YGRD
COLO NOIR ROUG
*=================================================================
FIN
```

frot86a.epx

```
FROT86A
ECHO
!CONV WIN
RESU ALIC TEMP ’frot86.alt’ GARD PSCR
SORT GRAP FENE 0.0 40.E-3
AXTE 1. ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot87.epx

```
FROT87
ECHO
 CONV WIN
LAGR CPLA
GEOM LIBR POIN 12 CAR4 4 TERM
  0 -0.5  1 -0.5  2 -0.5
  0  0    1  0    2  0
  0.43 0.11  0.93 -0.02  1.43 0.11
  0.43 0.61  0.93  0.58  1.43 0.61
  1 2 5 4
  2 3 6 5
  7 8 11 10
  8 9 12 11
opti dump dpma
COMP GROU 2 ’proj’ LECT 3 4 TERM
            ’targ’ LECT 1 2 TERM
     NGRO 2 ’cir’   LECT 7 8 9 TERM
            ’basup’ LECT targ TERM COND Y GT -0.01
     COUL TURQ LECT proj TERM
          VERT LECT targ TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT proj targ TERM
LINK COUP
     veri
     GPIN BODY LECT proj TERM
          BODY LECT targ TERM
          DIAM 0.05 LECT cir basup TERM
INIT VITE 1 100 LECT proj TERM
          2 -1  LECT proj TERM
ECRI DEPL VITE COOR FINT ACCE FLIA FREQ 1
     FICH ALIC FREQ 1
OPTI PAS UTIL
     LOG 1
     GPNS DUMP STAT
     LNKS DUMP STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 0
FIN
```

frot88.epx

```
FROT88
ECHO
 CONV WIN
LAGR CPLA
GEOM LIBR POIN 12 CAR4 4 TERM
  0 -0.5  1 -0.5  2 -0.5
  0  0    1  0    2  0
  0.43 0.11  0.93 -0.02  1.43 0.11
  0.43 0.61  0.93  0.58  1.43 0.61
  1 2 5 4
  2 3 6 5
  7 8 11 10
  8 9 12 11
opti dump dpma
COMP GROU 2 ’proj’ LECT 3 4 TERM
            ’targ’ LECT 1 2 TERM
     NGRO 2 ’cir’   LECT 7 8 9 TERM
            ’basup’ LECT targ TERM COND Y GT -0.01
     COUL TURQ LECT proj TERM
          VERT LECT targ TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT proj targ TERM
LINK COUP
     veri
     GPIN BODY LECT proj TERM
          BODY LECT targ TERM
          DIAM 0.05 LECT cir basup TERM
          MASL PAIR 2 1
INIT VITE 1 100 LECT proj TERM
          2 -1  LECT proj TERM
ECRI DEPL VITE COOR FINT ACCE FLIA FREQ 1
     FICH ALIC FREQ 1
OPTI PAS UTIL
     LOG 1
     GPNS DUMP STAT
     LNKS DUMP STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 0
FIN
```

frot89.epx

```
FROT89
ECHO
 CONV WIN
LAGR CPLA
DIME GLIS 2 200 TERM
GEOM LIBR POIN 12 CAR4 4 TERM
  0 -0.5  1 -0.5  2 -0.5
  0  0    1  0    2  0
  0.43 0.11  0.93 -0.02  1.43 0.11
  0.43 0.61  0.93  0.58  1.43 0.61
  1 2 5 4
  2 3 6 5
  7 8 11 10
  8 9 12 11
opti dump dpma
COMP GROU 2 'proj' LECT 3 4 TERM
            'targ' LECT 1 2 TERM
     NGRO 2 'cir'   LECT 7 8 9 TERM
            'basup' LECT targ TERM COND Y GT -0.01
     COUL TURQ LECT proj TERM
          VERT LECT targ TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT proj targ TERM
LIAI GLIS 2
          MAIT LECT 6 5 4 TERM
          ESCL LECT 7 8 9 TERM
          MAIT LECT 7 8 9 TERM
          ESCL LECT 6 5 4 TERM
INIT VITE 1 100 LECT proj TERM
          2 -1  LECT proj TERM
ECRI DEPL VITE COOR FINT ACCE FLIA FREQ 1
     FICH ALIC FREQ 1
OPTI PAS UTIL
     LOG 1
     LNKS DUMP STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 0
FIN
```

frot90.epx

```
FROT90
ECHO
 CONV WIN
LAGR CPLA
DIME GLIS 2 200 TERM
GEOM LIBR POIN 12 CAR4 4 TERM
  0 -0.5  1 -0.5  2 -0.5
  0  0    1  0    2  0
  0.43 0.11  0.93 -0.02  1.43 0.11
  0.43 0.61  0.93  0.58  1.43 0.61
  1 2 5 4
  2 3 6 5
  7 8 11 10
  8 9 12 11
opti dump dpma
COMP GROU 2 'proj' LECT 3 4 TERM
            'targ' LECT 1 2 TERM
     NGRO 2 'cir'   LECT 7 8 9 TERM
            'basup' LECT targ TERM COND Y GT -0.01
     COUL TURQ LECT proj TERM
          VERT LECT targ TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT proj targ TERM
LIAI GLIS 1
          MAIT LECT 6 5 4 TERM
          ESCL LECT 7 8 9 TERM
INIT VITE 1 100 LECT proj TERM
          2 -1  LECT proj TERM
ECRI DEPL VITE COOR FINT ACCE FLIA FREQ 1
     FICH ALIC FREQ 1
OPTI PAS UTIL
     LOG 1
     LNKS DUMP STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 0
FIN
```

frot91.dgibi

```
opti echo 0;
opti donn 'px4cir2d.proc';
opti donn 'pxordpoi.proc';
opti echo 1;
opti dime 2 elem qua4;
opti trac psc ftra 'frot91_mesh.ps';
opti sauv form 'frot91.msh';
r = 2.0;
x0 = 3.0;
pc = x0 r;
n = 8;
tol = 0.001;
c1 ier = px4cir2d (x0 0) ((x0+r) r) pc n tol;
c2 = c1 tour 90 pc;
c3 = c2 tour 90 pc;
c4 = c3 tour 90 pc;
circ = c1 et c2 et c3 et c4;
elim tol circ;
pm = pc plus (0 r);
elim tol (pm et circ);
mark = manu poi1 pm;
h = 1.0;
len = 10.0;
nh = enti (h / 0.5);
nb = enti (len / 1.0);
bas = (0 (0-h)) d nb (len (0-h));
base = bas tran nh (0 h);
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac qual mesh;
fin;
```

frot91.epx

```
FROT91
ECHO
!CONV WIN
CAST mesh
LAGR CPLA
GEOM CAR4 circ base PMAT mark TERM
opti dump dpma
COMP EPAI 0.2 LECT mark TERM ! Only for visualization
     NGRO 3 'cir'   LECT circ TERM COND ENVE
            'basup' LECT base TERM COND Y GT -0.01
            'b1'    LECT circ TERM COND NEAR POIN 3 0
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP SPLT NONE
     veri
     BLOQ 12 LECT bas TERM
     GPIN BODY LECT circ TERM
          BODY LECT base TERM
          DIAM 0.05 LECT cir basup TERM
CHAR 1 FACT 2
     FORC 1  1.E9 LECT pc TERM
     FORC 2 -1.E9 LECT pc TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT ACCE FLIA NUPA LECT 1432 PAS 1 1435 TERM
                                 ! FREQ 1000
          poin lect 238 239 240 200 25 24 term noel
     FICH ALIC FREQ 150
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 pc mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS DUMP STAT
     LNKS DUMP STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   5.00000E+00  1.55000E+00  2.80680E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  0.00000E+00
!RSPHERE:  5.61360E+00
!RADIUS :  2.80680E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.18930E+01
!FAR    :  3.92952E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          !LINE HEOU SFRE
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1
    !OBJE LECT circ mark TERM
     REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL
    !OBJE LECT circ mark TERM
     REND
GO
TRAC OFFS FICH AVI CONT
    !OBJE LECT circ mark TERM
     REND
ENDPLAY
SUIT
Post-treatment
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1. 'Time [s]'
COUR  1 'y_b1'   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 'dy_mk'  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 'dy_b1'  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 'dy_pc'  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 'dx_mk'  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 'dx_b1'  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 'dx_pc'  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 'vy_b1'  VITE COMP 2 NOEU LECT b1 TERM
```

```
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot92.dgibi

```
opti echo 0;
opti donn ’px4cir2d.proc’;
opti donn ’pxordpoi.proc’;
opti echo 1;
opti dime 2 elem qua4;
opti trac psc ftra ’frot92_mesh.ps’;
opti sauv form ’frot92.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r;
n = 8;
tol = 0.001;
c1 ier = px4cir2d (x0 0) ((x0+r) r) pc n tol;
c2 = c1 tour 90 pc;
c3 = c2 tour 90 pc;
c4 = c3 tour 90 pc;
circ = c1 et c2 et c3 et c4;
elim tol circ;
pm = pc plus (0 r);
elim tol (pm et circ);
mark = manu poi1 pm;
h = 1.0;
len = 10.0;
nh = enti (h / 0.5);
nb = enti (len / 1.0);
bas = (0 (0-h)) d nb (len (0-h));
base = bas tran nh (0 h);
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac qual mesh;
fin;
```

frot92.epx

```
FROT92
ECHO
!CONV WIN
CAST mesh
LAGR CPLA
GEOM CAR4 circ base PMAT mark TERM
COMP EPAI 0.2 LECT mark TERM ! Only for visualization
     NGRO 3 ’cir’   LECT circ TERM COND ENVE
            ’basup’ LECT base TERM COND Y GT -0.01
            ’b1’    LECT circ TERM COND NEAR POIN 3 0
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP SPLT NONE
     BLOQ 12 LECT bas TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT cir basup TERM
CHAR 1 FACT 2
     FORC 1  1.E9 LECT pc TERM
     FORC 2 -1.E9 LECT pc TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 1000
     FICH ALIC FREQ 150
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 pc mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   5.00000E+00  1.55000E+00  2.80680E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  0.00000E+00
!RSPHERE:  5.61360E+00
!RADIUS :  2.80680E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.18930E+01
!FAR    :  3.92952E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          !LINE HEOU SFRE
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1. ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot93.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot93_mesh.ps’;
opti sauv form ’frot93.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot93.epx

```
FROT93
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY LECT circ TERM
          BODY LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
          MASL PAIR 2 1
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB1
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
         !LINE HEOU SSHA
         !LNKS SHOW GPIN JOIN
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

frot94.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot94_mesh.ps’;
opti sauv form ’frot94.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot94.epx

```
FROT94
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
```

```
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
          MASL PAIR 2 1
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB1
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
         !LINE HEOU SSHA
         !LNKS SHOW GPIN JOIN
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

frot95.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot95_mesh.ps’;
opti sauv form ’frot95.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot95.epx

```
FROT95
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                   COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                             X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                   COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                             X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
          MASL PAIR 2 1
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB1 REBC
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
```

```
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
         !LINE HEOU SSHA
         !LNKS SHOW GPIN JOIN
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 ’dy_f1b’ DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b4’  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 ’dx_f1b’ DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
FIN
```

frot96.dgibi

```
opti echo 0;
opti donn ’px4cir3d.proc’;
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’frot96_mesh.ps’;
opti sauv form ’frot96.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot96.epx

```
FROT96
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 ’bloc’ LECT base TERM COND Y LT -0.99
             ’basu’ LECT base TERM cond Y GT -0.01
             ’ncy1’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             ’ncy2’ LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             ’ncyl’ LECT ncy1 DIFF ncy2 TERM
             ’f1a’  LECT axe TERM COND NEAR POIN 3 2 0
             ’f1b’  LECT axe TERM COND NEAR POIN 3 2 2
             ’f1’   LECT f1a f1b TERM
             ’f2’   LECT axe DIFF f1 TERM
             ’b1’   LECT circ TERM COND NEAR POIN 3 0 0
             ’b2’   LECT circ TERM COND NEAR POIN 3 0 0.667
             ’b3’   LECT circ TERM COND NEAR POIN 3 0 1.333
             ’b4’   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT basu ncyl TERM
          MASL PAIR 2 1
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB2
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
         !LINE HEOU SSHA
         !LNKS SHOW GPIN JOIN
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 ’y_b2’   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 ’y_b3’   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 ’y_b4’   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b4’  DEPL COMP 2 NOEU LECT b4 TERM
```

```
COUR  7 'dy_f1b' DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 'dx_mk'  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 'dx_b4'  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 'dx_f1b' DEPL COMP 1 NOEU LECT f1b TERM
TRAC 1 2 3 4 AXES 1.0 'Y-COOR. [M]' YZER
TRAC 5 6 7 AXES 1.0 'DY [M]' YZER
TRAC 8 9 10 AXES 1.0 'DX [M]' YZER
FIN
```

frot97.dgibi

```
opti echo 0;
opti donn 'px4cir3d.proc';
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra 'frot97_mesh.ps';
opti sauv form 'frot97.msh';
r = 2.0;
x0 = 3.0;
pc = x0 r 0;
pz = 0 0 1;
n = 8;
tol = 0.001;
ci1 ier = px4cir3d (x0 0 0) ((x0+r) r 0) pc (pc plus pz) n tol;
w = 2.0;
vzcir = 0 0 w;
nzc = 3;
c1 = ci1 volu tran nzc vzcir;
c2 = c1 tour 90 pc (pc plus pz);
c3 = c2 tour 90 pc (pc plus pz);
c4 = c3 tour 90 pc (pc plus pz);
circ = c1 et c2 et c3 et c4;
elim tol circ;
axe = pc d nzc (pc plus vzcir);
elim tol (circ et axe);
*trac cach qual circ;
*p1 = x0 0;
*p2 = (x0+r) r;
*p3 = x0 (r+r);
*p4 = (x0-r) r;
*elim tol (circ et p1 et p2 et p3 et p4);
*cc1 = cerc n p1 pc p2;
*cc2 = cc1 tour 90 pc;
*cc3 = cc2 tour 90 pc;
*cc4 = cc3 tour 90 pc;
*elim tol (circ et cc1 et cc2 et cc3 et cc4);
*pc1 = pxordpoi (chan POI1 cc1) p1;
*pc2 = pxordpoi (chan POI1 cc2) p2;
*pc3 = pxordpoi (chan POI1 cc3) p3;
*pc4 = pxordpoi (chan POI1 cc4) p4;
pm = pc plus (0 r w);
elim tol (pm et circ);
mark = manu poi1 pm;
trac cach qual (circ et mark);
*
h = 1.0;
len = 10.0;
a = 4.0;
nh = 2;
nb = enti (len / 1.0);
na = 6;
bas2d = (0 (0-h) -1.0) d nb (len (0-h) -1.0);
base2d = bas2d tran nh (0 h 0);
base = base2d volu tran na (0 0 a);
trac cach qual base;
trac cach qual (base et circ et mark);
*
*bas2 = (len 0) d nb (0 0);
*elim tol (bas2 et base);
*p5 = len 0;
*elim tol (p5 et bas2);
*pbas2 = chan POI1 bas2;
*pb2 = pxordpoi pbas2 p5;
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac cach qual mesh;
fin;
```

frot97.epx

```
FROT97
ECHO
!CONV WIN
CAST mesh
LAGR TRID
GEOM CUB8 circ base PMAT mark TERM
COMP NGRO 13 'bloc' LECT base TERM COND Y LT -0.99
             'basu' LECT base TERM cond Y GT -0.01
             'ncy1' LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 2.01
             'ncy2' LECT circ TERM
                    COND CYLI X1 3.0 Y1 2.0 Z1 -0.01
                              X2 3.0 Y2 2.0 Z2  2.01 R 1.99
             'ncyl' LECT ncy1 DIFF ncy2 TERM
             'f1a'  LECT axe TERM COND NEAR POIN 3 2 0
             'f1b'  LECT axe TERM COND NEAR POIN 3 2 2
             'f1'   LECT f1a f1b TERM
             'f2'   LECT axe DIFF f1 TERM
             'b1'   LECT circ TERM COND NEAR POIN 3 0 0
             'b2'   LECT circ TERM COND NEAR POIN 3 0 0.667
             'b3'   LECT circ TERM COND NEAR POIN 3 0 1.333
             'b4'   LECT circ TERM COND NEAR POIN 3 0 2
     EPAI 0.2 LECT mark TERM ! Only for visualization
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 123 LECT bloc TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.20 LECT ncyl TERM
          DIAM 0.40 LECT basu TERM
          MASL PAIR 2 1
CHAR 1 FACT 2
     FORC 1  3.33333E8 LECT f1 TERM
     FORC 1  6.66667E8 LECT f2 TERM
     FORC 2 -3.33333E8 LECT f1 TERM
     FORC 2 -6.66667E8 LECT f2 TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT FEXT FLIA FREQ 600
     FICH ALIC FREQ 100
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 b2 b3 b4 f1 f2 mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS STAT REB2
     LNKS STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   1.84447E+01  1.93365E+01  2.07597E+01
!        Q     9.11780E-01 -3.31861E-01  2.41333E-01 -1.68756E-02
         VIEW -4.51285E-01 -5.97022E-01 -6.63254E-01
         RIGH  8.82948E-01 -1.90951E-01 -4.28884E-01
         UP   -1.29404E-01  7.79167E-01 -6.13313E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  1.00000E+00
!RSPHERE:  5.95840E+00
!RADIUS :  2.97920E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.32377E+01
!FAR    :  4.17088E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
         !LINE HEOU SSHA
         !LNKS SHOW GPIN JOIN
     VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1 REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL REND
GO
TRAC OFFS FICH AVI CONT REND
ENDPLAY
SUIT
Post-treatment (time curves from alice temp file)
ECHO
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1.0 'Time [s]'
COUR  1 'y_b1'   COOR COMP 2 NOEU LECT b1 TERM
COUR  2 'y_b2'   COOR COMP 2 NOEU LECT b2 TERM
COUR  3 'y_b3'   COOR COMP 2 NOEU LECT b3 TERM
COUR  4 'y_b4'   COOR COMP 2 NOEU LECT b4 TERM
COUR  5 'dy_mk'  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 'dy_b4'  DEPL COMP 2 NOEU LECT b4 TERM
COUR  7 'dy_f1b' DEPL COMP 2 NOEU LECT f1b TERM
COUR  8 'dx_mk'  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 'dx_b4'  DEPL COMP 1 NOEU LECT b4 TERM
COUR 10 'dx_f1b' DEPL COMP 1 NOEU LECT f1b TERM
COUR 11 'vy_b1'  VITE COMP 2 NOEU LECT b1 TERM
COUR 12 'vy_b2'  VITE COMP 2 NOEU LECT b2 TERM
COUR 13 'vy_b3'  VITE COMP 2 NOEU LECT b3 TERM
COUR 14 'vy_b4'  VITE COMP 2 NOEU LECT b4 TERM
COUR 15 'vy_f1b' VITE COMP 2 NOEU LECT f1b TERM
COUR 16 'vy_f1a' VITE COMP 2 NOEU LECT f1a TERM
COUR 17 'vx_f1b' VITE COMP 1 NOEU LECT f1b TERM
COUR 18 'vx_f1a' VITE COMP 1 NOEU LECT f1a TERM
TRAC 1 2 3 4 AXES 1.0 'Y-COOR. [M]' YZER
TRAC 5 6 7 AXES 1.0 'DY [M]' YZER
TRAC 8 9 10 AXES 1.0 'DX [M]' YZER
TRAC 11 12 13 14 15 16 AXES 1.0 'VY [M/S]' YZER
TRAC 15 16 AXES 1.0 'VY [M/S]' YZER
TRAC 15 16 17 18 AXES 1.0 'V [M/S]' YZER
FIN
```

frot98.dgibi

```
opti echo 0;
opti donn 'px4cir2d.proc';
opti donn 'pxordpoi.proc';
opti echo 1;
opti dime 2 elem qua4;
```

```
opti trac psc ftra ’frot98_mesh.ps’;
opti sauv form ’frot98.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r;
n = 8;
tol = 0.001;
c1 ier = px4cir2d (x0 0) ((x0+r) r) pc n tol;
c2 = c1 tour 90 pc;
c3 = c2 tour 90 pc;
c4 = c3 tour 90 pc;
circ = c1 et c2 et c3 et c4;
elim tol circ;
pm = pc plus (0 r);
elim tol (pm et circ);
mark = manu poi1 pm;
h = 1.0;
len = 10.0;
nh = enti (h / 0.5);
nb = enti (len / 1.0);
bas = (0 (0-h)) d nb (len (0-h));
base = bas tran nh (0 h);
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac qual mesh;
fin;
```

frot98.epx

```
FROT98
ECHO
!CONV WIN
CAST mesh
LAGR CPLA
GEOM CAR4 circ base PMAT mark TERM
opti dump dpma
COMP EPAI 0.2 LECT mark TERM ! Only for visualization
     NGRO 3 ’cir’   LECT circ TERM COND ENVE
            ’basup’ LECT base TERM COND Y GT -0.01
            ’b1’    LECT circ TERM COND NEAR POIN 3 0
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 12 LECT bas TERM
     GPIN BODY LECT circ TERM
          BODY LECT base TERM
          DIAM 0.05 LECT cir basup TERM
CHAR 1 FACT 2
     FORC 1  1.E9 LECT pc TERM
     FORC 2 -1.E9 LECT pc TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT ACCE FLIA NUPA LECT 1432 PAS 1 1435 TERM
                                  ! FREQ 1000
          poin lect 238 239 240 200 25 24 term noel
     FICH ALIC FREQ 150
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 pc mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS RCEL DUMP STAT
     LNKS DUMP STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   5.00000E+00  1.55000E+00  2.80680E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  0.00000E+00
!RSPHERE:  5.61360E+00
!RADIUS :  2.80680E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.18930E+01
!FAR    :  3.92952E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          !LINE HEOU SFRE
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1
     !OBJE LECT circ mark TERM
     REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL
     !OBJE LECT circ mark TERM
     REND
GO
TRAC OFFS FICH AVI CONT
     !OBJE LECT circ mark TERM
     REND
ENDPLAY
SUIT
Post-treatment
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1. ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

frot99.dgibi

```
opti echo 0;
opti donn ’px4cir2d.proc’;
opti donn ’pxordpoi.proc’;
opti echo 1;
opti dime 2 elem qua4;
opti trac psc ftra ’frot99_mesh.ps’;
opti sauv form ’frot99.msh’;
r = 2.0;
x0 = 3.0;
pc = x0 r;
n = 8;
tol = 0.001;
c1 ier = px4cir2d (x0 0) ((x0+r) r) pc n tol;
c2 = c1 tour 90 pc;
c3 = c2 tour 90 pc;
c4 = c3 tour 90 pc;
circ = c1 et c2 et c3 et c4;
elim tol circ;
pm = pc plus (0 r);
elim tol (pm et circ);
mark = manu poi1 pm;
h = 1.0;
len = 10.0;
nh = enti (h / 0.5);
nb = enti (len / 1.0);
bas = (0 (0-h)) d nb (len (0-h));
base = bas tran nh (0 h);
mesh = circ et base et mark;
tass mesh noop;
sauv form mesh;
trac qual mesh;
fin;
```

frot99.epx

```
FROT99
ECHO
!CONV WIN
CAST mesh
LAGR CPLA
GEOM CAR4 circ base PMAT mark TERM
opti dump dpma
COMP EPAI 0.2 LECT mark TERM ! Only for visualization
     NGRO 3 ’cir’   LECT circ TERM COND ENVE
            ’basup’ LECT base TERM COND Y GT -0.01
            ’b1’    LECT circ TERM COND NEAR POIN 3 0
     COUL TURQ LECT circ TERM
          VERT LECT base TERM
          ROUG LECT mark TERM
MATE LINE RO 7800. YOUN 2.E11 NU 0.
          LECT circ base TERM
     MASS 0.0 LECT mark TERM
LINK COUP
     BLOQ 12 LECT bas TERM
     GPIN BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT circ TERM
          BODY FROT MUST 0.25 MUDY 0.25 GAMM 0.1 LECT base TERM
          DIAM 0.05 LECT cir basup TERM
CHAR 1 FACT 2
     FORC 1  1.E9 LECT pc TERM
     FORC 2 -1.E9 LECT pc TERM
     TABL 3 0. 0.  12.E-4 1.  6. 1.
ECRI DEPL VITE COOR FINT ACCE FLIA NUPA LECT 1432 PAS 1 1435 TERM
                                  ! FREQ 1000
          poin lect 238 239 240 200 25 24 term noel
     FICH ALIC FREQ 150
     FICH ALIC TEMP FREQ 1
          POIN LECT b1 pc mark TERM
OPTI PAS UTIL
     LOG 1
     GPNS RCEL DUMP STAT
     LNKS DUMP STAT VISU
CALC TINI 0 TFIN 40.E-3 PASF 2.E-5 NMAX 2000
PLAY
CAME   1 EYE   5.00000E+00  1.55000E+00  2.80680E+01
!        Q     1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
```

```
         VIEW  0.00000E+00  0.00000E+00 -1.00000E+00
         RIGH  1.00000E+00  0.00000E+00  0.00000E+00
         UP    0.00000E+00  1.00000E+00  0.00000E+00
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  5.00000E+00  1.55000E+00  0.00000E+00
!RSPHERE:  5.61360E+00
!RADIUS :  2.80680E+01
!ASPECT :  1.00000E+00
!NEAR   :  2.18930E+01
!FAR    :  3.92952E+01
SCEN GEOM NAVI FREE
          POIN SPHP
          FACE HFRO
          !LINE HEOU SFRE
          LNKS SHOW GPIN JOIN
    !VECT SCCO FIEL VITE SCAL USER PROG 100.0 PAS 100.0 1400.0 TERM
    !VECT SCCO FIEL FLIA SCAL USER PROG 0.25E8 PAS 0.25E8 3.5E8 TERM
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 2001 FPS 15 KFRE 10 COMP -1
     !OBJE LECT circ mark TERM
     REND
GOTR LOOP 1999 OFFS FICH AVI CONT NOCL
     !OBJE LECT circ mark TERM
     REND
GO
TRAC OFFS FICH AVI CONT
     !OBJE LECT circ mark TERM
     REND
ENDPLAY
SUIT
Post-treatment
RESU ALIC TEMP GARD PSCR
SORT GRAP
AXTE 1. ’Time [s]’
COUR  1 ’y_b1’   COOR COMP 2 NOEU LECT b1 TERM
COUR  5 ’dy_mk’  DEPL COMP 2 NOEU LECT mark TERM
COUR  6 ’dy_b1’  DEPL COMP 2 NOEU LECT b1 TERM
COUR  7 ’dy_pc’  DEPL COMP 2 NOEU LECT pc TERM
COUR  8 ’dx_mk’  DEPL COMP 1 NOEU LECT mark TERM
COUR  9 ’dx_b1’  DEPL COMP 1 NOEU LECT b1 TERM
COUR 10 ’dx_pc’  DEPL COMP 1 NOEU LECT pc TERM
COUR 11 ’vy_b1’  VITE COMP 2 NOEU LECT b1 TERM
COUR 15 ’vy_pc’  VITE COMP 2 NOEU LECT pc TERM
COUR 17 ’vx_pc’  VITE COMP 1 NOEU LECT pc TERM
TRAC 1 AXES 1.0 ’Y-COOR. [M]’ YZER
TRAC 5 6 7 AXES 1.0 ’DY [M]’ YZER
TRAC 8 9 10 AXES 1.0 ’DX [M]’ YZER
TRAC 11 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 AXES 1.0 ’VY [M/S]’ YZER
TRAC 15 17 AXES 1.0 ’V [M/S]’ YZER
FIN
```

gpn2plot.epx

```
GPN2PLOT
ECHO
*CONV WIN
LAGR CPLA
GEOM LIBR POIN 4 CAR1 1 TERM
   0 0  1 0  1 1  0 1
   1 2 3 4
MATE LINE RO 8000. YOUN 1.D11 NU 0.3
          LECT 1 TERM
ECRI FICH ALIC FREQ 1
CALC TINI 0.0 TEND 1.0 NMAX 0
*=================================================================
SUIT
Post-treatment
ECHO
RESU ALIC GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
RCOU  1 ’n_raw’   FICH ’frot82_gpn.pun’
RCOU  2 ’n_rebo’  FICH ’frot82_gpn.pun’
RCOU  3 ’penemx’  FICH ’frot82_gpn.pun’
RCOU  4 ’penemax’ FICH ’frot82_gpn.pun’
RCOU  5 ’pdotmx’  FICH ’frot82_gpn.pun’
RCOU  6 ’pdotmax’ FICH ’frot82_gpn.pun’
TRAC  1 AXES 1.0 ’N_RAW’ YZER ! XGRD YGRD
TRAC  2 AXES 1.0 ’N_REBO’ YZER ! XGRD YGRD
TRAC  3 AXES 1.0 ’PENEMX [M]’ YZER ! XGRD YGRD
TRAC  4 AXES 1.0 ’PENEMAX [M]’ YZER ! XGRD YGRD
TRAC  5 AXES 1.0 ’PDOTMX [M/S]’ YZER ! XGRD YGRD
TRAC  6 AXES 1.0 ’PDOTMAX [M/S]’ YZER ! XGRD YGRD
TRAC 3 4 AXES 1.0 ’PENE [M]’ YZER ! XGRD YGRD
COLO NOIR ROUG
TRAC 5 6 AXES 1.0 ’PDOT [M/S]’ YZER ! XGRD YGRD
COLO NOIR ROUG
*=================================================================
FIN
```

gpn2plot.f

```
      PROGRAM gpn2plot
*
* read europlexus gpn file and produce time history of the contained
* quantities, ready to be plotted by EUROPLEXUS itself
* - limit number of output points to 9999 (enough for a graph)
*
      IMPLICIT NONE
      integer, parameter :: nmax = 9999 ! max n. of output points
      CHARACTER*256 :: fname, fout
      INTEGER :: dum, n, l, i, nout, freq
      INTEGER, ALLOCATABLE :: step(:), n_raw(:), n_rebo(:)
      REAL(8) :: rat
      REAL(8), ALLOCATABLE :: time(:), penemx(:), penemax(:), pdotmx(:),
     >                        pdotmax(:)
      WRITE (6, *) "Enter gpn file name (with our without extension)"
      READ (5, *) fname
      l = LEN_TRIM (fname)
      IF (l > 3) THEN
        IF (fname(l-3:l) /= ’.gpn’ .AND. fname(l-3:l) /= ’.GPN’) THEN
          fname = fname(1:l) // ’.gpn’
        ENDIF
      ELSE
        fname = fname(1:l) // ’.gpn’
      ENDIF
      OPEN (1, FILE=fname, ERR=901)
      READ (1, 1001) dum
 1001 FORMAT (I1)
      n = 0
    1 READ (1, 1001, END=2) dum
      n = n + 1
      GO TO 1
    2 ALLOCATE (step(n), time(n), n_raw(n), n_rebo(n), penemx(n),
     >          penemax(n), pdotmx(n), pdotmax(n))
      REWIND (1)
      READ (1, 1001) dum
      DO i = 1, n
        READ (1, 1002, END=900) step(i), time(i), n_raw(i), n_rebo(i),
     >                          penemx(i), penemax(i), pdotmx(i),
     >                          pdotmax(i)
 1002   FORMAT (I7,E12.5,2I8,4E11.3)
      END DO
      CLOSE (1)
*
* write curves extracted from log file in separate files
*
      fout = fname(1:LEN_TRIM(fname)-4)
!fc   CALL woutr (fout, ’CPU   ’, time, cpu, n)
!fc   CALL woutr (fout, ’DTCRIT’, time, dtcrit, n)
!     CALL wouti (fout, ’ELCR  ’, time, elcr, n)
!fc   CALL woutr (fout, ’DEE   ’, time, dee, n)
!fc   CALL woutr (fout, ’DMMN  ’, time, dmmn, n)
!fc   CALL woutr (fout, ’DMME  ’, time, dmme, n)
!fc   CALL woutr (fout, ’DTMX  ’, time, dtmx, n)
!     CALL wouti (fout, ’EL    ’, time, el, n)
!fc   CALL woutr (fout, ’VITMAX’, time, vitmax, n)
!     CALL wouti (fout, ’NODE  ’, time, node, n)
!     CALL wout8 (fout, ’ELSTEP’, time, elstep, n)
*
* write same data in EUROPLEXUS "punch" format
* (but limited to nmax points, equally sampled)
*
      IF (n <= nmax) THEN
        nout = n
        freq = 1
      ELSE
        rat = DBLE(n) / DBLE(nmax)
        freq = CEILING (rat)
        rat = DBLE(n) / DBLE(freq)
        nout = FLOOR (rat) + 1
      ENDIF
      PRINT *, ’n=’, n, ’ nmax=’, nmax
      PRINT *, ’freq=’, freq, ’ nout=’, nout
*
      OPEN (1, FILE=TRIM(fout)//’_gpn.pun’)
*
      write(1,2001) nout
 2001 format(" VALUES",i5,"     COMPONENTS 1",/,
     >       "*    Time             N_RAW",/,
     >       "*    T                n_raw")
      write(1,2000) (time(i), DBLE(n_raw(i)), i=1,n,freq)
 2000 format(1p2e17.6)
*
      write(1,2002) nout
 2002 format(" VALUES",i5,"     COMPONENTS 1",/,
     >       "*    Time             N_REBO",/,
     >       "*    T                n_rebo")
      write(1,2000) (time(i), DBLE(n_rebo(i)), i=1,n,freq)
*
      write(1,2004) nout
 2004 format(" VALUES",i5,"     COMPONENTS 1",/,
     >       "*    Time             PENEMX",/,
     >       "*    T                penemx")
      write(1,2000) (time(i), penemx(i), i=1,n,freq)
*
      write(1,2005) nout
 2005 format(" VALUES",i5,"     COMPONENTS 1",/,
     >       "*    Time             PENEMAX",/,
     >       "*    T                penemax")
      write(1,2000) (time(i), penemax(i), i=1,n,freq)
*
      write(1,2006) nout
 2006 format(" VALUES",i5,"     COMPONENTS 1",/,
     >       "*    Time             PDOTMX",/,
     >       "*    T                pdotmx")
      write(1,2000) (time(i), pdotmx(i), i=1,n,freq)
*
```

```
      write(1,2007) nout
 2007 format(" VALUES",i5,"     COMPONENTS 1",/,
     >       "*    Time             PDOTMAX",/,
     >       "*    T                pdotmax")
      write(1,2000) (time(i), pdotmax(i), i=1,n,freq)
*
      CLOSE (1)
*
  900 DEALLOCATE (step, time, n_raw, n_rebo, penemx, penemax,
     >            pdotmx, pdotmax)
      STOP
  901 WRITE (6, *) "ERROR - file not found:", TRIM(fname)
      END PROGRAM gpn2plot
*
!fc   SUBROUTINE woutr (fout, nam, t, v, n)
!fc   IMPLICIT NONE
!fc   CHARACTER*256, INTENT(IN) :: fout
!fc   CHARACTER*6, INTENT(IN) :: nam
!fc   INTEGER, INTENT(IN) :: n
!fc   REAL(8), INTENT(IN) :: t(n), v(n)
!fc   INTEGER :: i
!fc   OPEN (1, FILE=TRIM(fout)//'.'//TRIM(nam))
!fc   WRITE (1, 2001) TRIM(nam), n
!2001 FORMAT ("DCOU 1 '",A,"'",I8)
!fc   WRITE (1, 2002) (t(i), v(i), i = 1, n)
!2002 FORMAT (1p6E12.5)
!fc   CLOSE (1)
!fc   END SUBROUTINE woutr
```

osto03.dgibi

```
'DEBPROC' pxbox3d x0*'FLOTTANT' y0*'FLOTTANT' z0*'FLOTTANT'
                  lx*'FLOTTANT' ly*'FLOTTANT' lz*'FLOTTANT'
                  dd*'FLOTTANT';
*
*---------------------------------------------------
* Generates a parallelepiped mesh with origin in point
* (x0,y0,z0), sides of length (lx,ly,lz) and density (mesh size) dd.
* The mesh consists of CUB8 hexahedral elements and is oriented
* along the global axes.
*
* Input :
* -----
*     x0,y0,z0 : coordinates of 'origin' of the box
*     lx,ly,lz : length of the box sides
*     dd : "density" (size) of the mesh (the same in all directions)
* Output :
* ------
*     box : mesh consisting of CUB8 hexahedra
*---------------------------------------------------
*
dens dd;
p1 = x0 y0 z0;
p2 = (x0 + lx) y0 z0;
p3 = (x0 + lx) (y0 + ly) z0;
p4 = x0 (y0 + ly) z0;
*
c1 = p1 d p2;
c2 = p2 d p3;
c3 = p3 d p4;
c4 = p4 d p1;
base = dall c1 c2 c3 c4 plan;
*
box = base volu tran (0 0 lz);
*
finproc box;
*-------------------------------------------------------------------------
'DEBPROC' pxrec3d x0*'FLOTTANT' y0*'FLOTTANT' z0*'FLOTTANT'
                  lx*'FLOTTANT' ly*'FLOTTANT'
                  dd*'FLOTTANT';
*
*---------------------------------------------------
* Generates a rectangle mesh with origin in point
* (x0,y0,z0), sides of length (lx,ly) and density (mesh size) dd.
* The mesh consists of QUA4 quadrilateral elements and is oriented
* along the global axes.
*
* Input :
* -----
*     x0,y0,z0 : coordinates of 'origin' of the box
*     lx,ly    : length of the box sides
*     dd : "density" (size) of the mesh (the same in all directions)
* Output :
* ------
*     box : mesh consisting of CUB8 hexahedra
*---------------------------------------------------
*
dens dd;
p1 = x0 y0 z0;
p2 = (x0 + lx) y0 z0;
p3 = (x0 + lx) (y0 + ly) z0;
p4 = x0 (y0 + ly) z0;
*
c1 = p1 d p2;
c2 = p2 d p3;
c3 = p3 d p4;
c4 = p4 d p1;
rect = dall c1 c2 c3 c4 plan;
*
finproc rect;
```

```
*-------------------------------------------------------------------------
*$$$$ PX4CIR3D
*
* Pour generer le maillage 3D (plan) d'un quart de cercle
* avec seulement des quadrilateres a 4 noeuds.
* Le quart de cercle est defini par les deux extremes
* d'un arc (de 90 degrees), par le centre du cercle
* et par un autre point qui definit l'axe de rotation
* (axe perpendiculaire au plan du cercle, passant pour son centre).
*
* Input:
* =====
* P1 = premiere extremite de l'arc
* P2 = deuxieme extremite de l'arc
* PC = centre de l'arc
* PZ = autre point de l'axe
* N  = nombre de mailles a generer sur chaque cote (doit etre pair)
* TOL= tolerance pour l'elimination des noeuds doubles
*
* Output:
* ======
* SUR = objet MAILLAGE d'elements de type QUA4
* IER = 0: pas d'erreur, .NE.0: erreur dans la generation de SUR
*
'DEBPROC' PX4CIR3D P1*'POINT' P2*'POINT' PC*'POINT' PZ*'POINT'
                   N*'ENTIER' TOL*'FLOTTANT';
*-------------------------------------------------------------------
*
ier=0;
n2 = n / 2;
p0 = 0 0 0;
pm1 = p1 plus p0;
depl pm1 tour 45 pc pz;
pm2 = 0.5*(pc plus p2);
pm3 = 0.5*(pc plus p1);
pm = 0.5*(pc plus pm1);
c1a = cerc n2 p1 pc pm1;
c1b = cerc n2 pm1 pc p2;
c2a = droi n2 p2 pm2;
c2b = droi n2 pm2 pc;
c3a = droi n2 pc pm3;
c3b = droi n2 pm3 p1;
c4a = droi n2 pm pm1;
c4b = droi n2 pm pm2;
c4c = droi n2 pm pm3;
sur1 = dall plan c4c c3b c1a (inve c4a);
sur2 = dall plan c4a c1b c2a (inve c4b);
sur3 = dall plan c2b c3a (inve c4c) c4b;
sur = sur1 et sur2 et sur3;
*
elim tol sur;
*
'FINPROC' sur ier;
*-------------------------------------------------------------------------
*
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra 'osto03_mesh.ps';
opti sauv form 'osto03.msh';
*
*dd = 0.0025;
dd = 0.00625;
dens dd;
tol = 0.1*dd;
*
* Plate
*
pla1 = pxrec3d 0.0    0.0    0.0 0.15   0.15   dd;
pla2 = pxrec3d 0.15   0.15   0.0 0.05   0.05   dd;
pla3 = pxrec3d 0.0875 0.15   0.0 0.0125 0.05   dd;
pla4 = pxrec3d 0.03   0.15   0.0 0.0075 0.05   dd;
pla5 = pxrec3d 0.15   0.0875 0.0 0.05   0.0125 dd;
pla6 = pxrec3d 0.15   0.03   0.0 0.05   0.0075 dd;
trac qual cach (pla3 ET pla4);
*
p0 = 0 0 0;
***/*/*/*/*/cont = p0 d (0.025 0 0) d (0.025 0.0185 0) c 4 (0.025 0.025 0)
cont = p0 d (0.025 0 0) d (0.025 0.0185 0) c 2 (0.025 0.025 0)
       (0.0185 0.025 0) d (0 0.025 0) d p0;
trac qual cach cont;
qua1 = cont surf plan;
trac qual cach qua1;
qua2 = qua1 syme plan (0.025 0 0) (0.025 0.025 0) (0.025 0 0.025);
qua12 = qua1 et qua2;
qua34 = qua12 syme plan (0 0.025 0) (0.025 0.025 0) (0 0.025 0.025);
q1 = qua12 et qua34;
elim tol q1;
trac qual cach q1;
orie q1 dire (0 0 1);
*
pla7 =  q1 plus (0.0375 0.15   0);
pla8 =  q1 plus (0.10   0.15   0);
pla9 =  q1 plus (0.15   0.10   0);
depl pla9 tour 90 (0.175 0.125 0) (0.175 0.125 1);
pla10 = q1 plus (0.15   0.0375 0);
depl pla10 tour 90 (0.175 0.0625 0) (0.175 0.0625 1);
*
dens dd;
*con5 = p0 d (0.030 0 0) d (0.030 0.025 0) d (0.012 0.025 0)
*       c (0 0.025 0) (0 0.013 0) d p0;
con5 = p0 d (0.0375 0 0) d (0.0375 0.025 0) d (0.012 0.025 0)
       c 2 (0 0.025 0) (0 0.013 0) d p0;
trac qual cach con5;
```

```
qua5 = con5 surf plan;
trac qual cach qua5;
orie qua5 dire (0 0 1);
qua6 = qua5 syme plan (0 0.025 0) (0.030 0.025 0) (0 0.025 0.025);
orie qua6 dire (0 0 1);
q56 = qua5 et qua6;
elim tol q56;
trac qual cach q56;
orie q56 dire (0 0 1);
*
pla11 = q56 plus (0 0.15 0);
pla12 = q56 plus (0 0 0);
depl pla12 tour 90 (0 0.025 0) (0 0.025 0.025);
depl pla12 plus (0.175 -0.025 0);
*
*plate = pla1 et pla2 et pla3 et pla4 et pla5 et pla6 et
*         pla7 et pla8 et pla9 et pla10 et pla11 et pla12;
plate = pla1 et pla2 et pla3 et pla5 et
        pla7 et pla8 et pla9 et pla10 et pla11 et pla12;
elim tol plate;
orie plate dire (0 0 1);
trac qual cach plate;
*
*platet = plate elem tri3;
*orie platet dire (0 0 1);
plateq = plate elem qua4;
orie plateq dire (0 0 1);
*
* Lower frame
*
blf = (plate diff pla1) plus (0 0 -0.0154);
lframe = blf volu tran (0 0 0.015);
trac qual cach lframe;
lfra8 = lframe elem cub8;
*lfra6 = lframe elem pri6;
*
* Upper frame (without the bolts)
*
uf2 = pla2 plus (0 0 0.0004);
uf3 = pla3 plus (0 0 0.0004);
uf4 = pla4 plus (0 0 0.0004);
uf5 = pla5 plus (0 0 0.0004);
uf6 = pla6 plus (0 0 0.0004);
uf11 = pla11 plus (0 0 0.0004);
uf12 = pla12 plus (0 0 0.0004);
couf = p0 d (0.025 0 0) d (0.025 0.0190 0) c 2 (0.025 0.025 0)
       (0.0190 0.025 0) d (0 0.025 0) d p0;
quf1 = couf surf plan;
quf2 = quf1 syme plan (0.025 0 0) (0.025 0.025 0) (0.025 0 0.025);
quf12 = quf1 et quf2;
quf34 = quf12 syme plan (0 0.025 0) (0.025 0.025 0) (0 0.025 0.025);
u1 = quf12 et quf34;
elim tol u1;
orie u1 dire (0 0 1);
*
uf7 =  u1 plus (0.0375 0.15   0.0004);
uf8 =  u1 plus (0.10   0.15   0.0004);
uf9 =  u1 plus (0.15   0.10   0.0004);
uf10 = u1 plus (0.15   0.0375 0.0004);
*
*buf = uf2 et uf3 et uf4 et uf5 et uf6 et
*      uf7 et uf8 et uf9 et uf10 et uf11 et uf12;
buf = uf2 et uf3 et uf5 et
      uf7 et uf8 et uf9 et uf10 et uf11 et uf12;
elim tol buf;
uframe = buf volu tran (0 0 0.015);
trac qual cach uframe;
ufra8 = uframe elem cub8;
*ufra6 = uframe elem pri6;
*
* Bolts
*
*n = 4;
n = 2;
pz = 0 0 1;
b1 ier = px4cir3d (0.006 0 0) (0 0.006 0) p0 pz n tol;
b2 = b1 tour 90 p0 pz;
b3 = b2 tour 90 p0 pz;
b4 = b3 tour 90 p0 pz;
basu = b1 et b2 et b3 et b4;
elim tol basu;
basl = basu plus p0;
depl basu plus (0 0 0.0004);
bolu = basu volu tran 2 (0 0 0.0150);
boll = basl volu tran 2 (0 0 0.0158);
depl boll plus (0 0 -0.0154);
bolt = boll et bolu;
elim tol bolt;
trac qual cach bolt;
*
bolt1 = bolt plus (0.0625 0.1750 0);
bolt2 = bolt plus (0.1250 0.1750 0);
bolt3 = bolt plus (0.1750 0.1250 0);
bolt4 = bolt plus (0.1750 0.0625 0);
bolts = bolt1 et bolt2 et bolt3 et bolt4;
*
uframeb = uframe et bolts;
elim tol uframeb;
*
* Pressurized surfaces
*
pairb = pla1 coul noir;
ppimp = blf coul noir;
*ppimp3 = ppimp elem tri3;
ppimp4 = ppimp elem qua4;
trac qual cach (pairb et ppimp);
*
mesh = plate et lframe et uframeb et pairb et ppimp;
tass mesh noop;
sauv form mesh;
trac qual cach mesh;
*
fin;
```

osto03.epx

```
OSTO03
ECHO
!CONV WIN
CAST mesh
EROS 1.0
TRID LAGR
DIME
  ADAP NPOI 7000 Q4GS 7000 CL3D 7000 !NPIN 7000
  ENDA
TERM
GEOM CUB8 lfra8 ufra8 bolts
     Q4GS plateq
     CL3D pairb ppimp4
TERM
COMP ORIE OBJE LECT plate TERM POIN 0 0 1
     EPAI 0.0008 LECT plate _q4gs TERM
     GROU 9 'pbol1'  LECT ppimp TERM
                     COND SPHE XC 0.0625 YC 0.175  ZC -0.0154 R 10.E-3
            'pbol2'  LECT ppimp TERM
                     COND SPHE XC 0.125  YC 0.175  ZC -0.0154 R 10.E-3
            'pbol3'  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.125  ZC -0.0154 R 10.E-3
            'pbol4'  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.0625 ZC -0.0154 R 10.E-3
            'pbolts' LECT pbol1 pbol2 pbol3 pbol4 TERM
            'PrCen' LECT pairb TERM
                     COND NEAR POIN 0.0 0.0 0.0
            'PrSen6' LECT pairb TERM
                   COND NEAR POIN 0.175 0.0 0.0
            'pinada' LECT plate TERM
                   COND BOX X0 0.0 Y0 0.0  Z0 -0.1
                            DX 0.15 DY 0.15 DZ 0.2
            'pinade' LECT plate TERM
                   COND BOX X0 0.0 Y0 0.0  Z0 -0.1
                            DX 0.18 DY 0.18 DZ 0.2
     NGRO 9 'bloc'   LECT uframe TERM COND Z GT 0.01539
            'symx'   LECT mesh TERM COND X LT 0.0001
            'symy'   LECT mesh TERM COND Y LT 0.0001
            'symxP'   LECT plate TERM COND X LT 0.0001
            'symyP'   LECT plate TERM COND Y LT 0.0001
            'cen'    LECT plate TERM COND NEAR POIN 0.0 0.0 0.0
            'axis1'   LECT plate TERM COND X LT 0.0001
            'axis2'   LECT plate TERM COND Y LT 0.0001
            'plateN'   LECT plate diff axis1 axis2 TERM
     COUL VERT LECT plate  TERM
          ROSE LECT lframe TERM
          TURQ LECT uframe bolts TERM
          ROUG LECT pairb TERM
          JAUN LECT pbolts TERM
          GR50 LECT ppimp DIFF pbolts TERM
 ADAP THRS ECRO 11 TMIN 0.05 TMAX 0.25 MAXL 3
           LECT pinade TERM
MATE VPJC RO 2700.0 YOUN 70.0E9 NU 0.3 ELAS 80.0E6 mxit 500
          QR1 49.3E6 CR1 1457.1 QR2 5.2E6 CR2 121.5
          PDOT 5.E-4 C 0.014 TQ 0.9 CP 910.0
          TM 893.0 M 0.0 DC 1.0 WC 65.0E6
     LECT plate _q4gs TERM
     VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 3.7E8 mxit 500
          QR1 2.364E8 CR1 39.3 QR2 4.081E8 CR2 4.5
          PDOT 5.E-4 C 1.E-3 TQ 0.9 CP 452.0
          TM 1800.0 M 0.0 DC 0.9 WC 473.0E6
          LECT lframe uframe bolts TERM
     IMPE PIMP RO 7850.0 PRES 0 PREF 0
          LECT ppimp diff pbol1 pbol2 pbol3 pbol4 TERM
     IMPE PIMP RO 7850.0 PRES 340.e6 PREF 0
          LECT pbol1 pbol2 pbol3 pbol4 TERM
     IMPE AIRB X 0.0 Y 0.0 Z -0.375
          MASS 0.0402 TAUT ANGL COEF 1.0
          LECT pairb _cl3d TERM
LINK COUP SPLT NONE SOLV PARD
     BLOQ 123 LECT bloc TERM
     BLOQ 1 LECT symx TERM
     BLOQ 2 LECT symy TERM
     BLOQ 56 LECT symxP TERM
     BLOQ 46 LECT symyP TERM
     GLIS 2 FROT MUST 0.1 MUDY 0.1 GAMM 0 ! Contact surface #1
            PGAP 0.4E-3
            MAIT LECT lframe TERM
            PESC LECT plateN  TERM
            FROT MUST 0.1 MUDY 0.1 GAMM 0 ! Contact surface #2
            PGAP 0.4E-3
            MAIT LECT uframeb TERM
            PESC LECT plateN   TERM
ECRI VITE TFRE 1.E-3
          POIN LECT 1 TERM
          NOEL
     FICH ALIT TFREQ 0.1E-4
          POIN LECT cen axis1 axis2 TERM
          ELEM LECT PrCen PrSen6 TERM
```

```
     FICH SPLI ALIC TFRE 1.0E-4
OPTI NOTE
     JAUM
     CSTA 0.5
     GLIS NORM ELEM
     LOG 1
     LMST
     ADAP RCON
MEAS ! Measurement commands (batch)
  OBJE LECT pbolts TERM
  EMIN LECT pbolts TERM
  EMAX LECT pbolts TERM
CALC TINI 0.0 TEND 10.E-3
FIN
```

osto03a.epx

```
OSTO03A
ECHO
!CONV WIN
RESU SPLI ALIC 'osto03.ali' GARD PSCR
SORT VISU NSTO 1
PLAY
CAME   1 EYE  -3.18123E-01 -3.98270E-01  3.11803E-01
!        Q     7.92529E-01  5.04896E-01 -1.83767E-01 -2.88457E-01
         VIEW  5.82563E-01  6.94272E-01 -4.22618E-01
         RIGH  7.66044E-01 -6.42788E-01  0.00000E+00
         UP    2.71654E-01  3.23744E-01  9.06308E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.00092E-01  1.00140E-01  8.40999E-03
!RSPHERE:  1.43578E-01
!RADIUS :  7.17889E-01
!ASPECT :  1.00000E+00
!NEAR   :  5.59953E-01
!FAR    :  1.00504E+00
SCEN GEOM NAVI FREE
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 101 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 99 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
FIN
```

osto03b.epx

```
OSTO03B
ECHO
RESU ALIC TEMP 'osto03.alt' GARD PSCR
COMP GROU 9 'pbol1'  LECT ppimp TERM
                     COND SPHE XC 0.0625 YC 0.175  ZC -0.0154 R 10.E-3
            'pbol2'  LECT ppimp TERM
                     COND SPHE XC 0.125  YC 0.175  ZC -0.0154 R 10.E-3
            'pbol3'  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.125  ZC -0.0154 R 10.E-3
            'pbol4'  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.0625 ZC -0.0154 R 10.E-3
            'pbolts' LECT pbol1 pbol2 pbol3 pbol4 TERM
            'PrCen' LECT pairb TERM
                     COND NEAR POIN 0.0 0.0 0.0
            'PrSen6' LECT pairb TERM
                   COND NEAR POIN 0.175 0.0 0.0
            'pinada' LECT plate TERM
                   COND BOX X0 0.0 Y0 0.0  Z0 -0.1
                            DX 0.15 DY 0.15 DZ 0.2
            'pinade' LECT plate TERM
                   COND BOX X0 0.0 Y0 0.0  Z0 -0.1
                            DX 0.18 DY 0.18 DZ 0.2
     NGRO 9 'bloc'   LECT uframe TERM COND Z GT 0.01539
            'symx'   LECT mesh TERM COND X LT 0.0001
            'symy'   LECT mesh TERM COND Y LT 0.0001
            'symxP'   LECT plate TERM COND X LT 0.0001
            'symyP'   LECT plate TERM COND Y LT 0.0001
            'cen'    LECT plate TERM COND NEAR POIN 0.0 0.0 0.0
            'axis1'   LECT plate TERM COND X LT 0.0001
            'axis2'   LECT plate TERM COND Y LT 0.0001
            'plateN'   LECT plate diff axis1 axis2 TERM
SORT GRAP
AXTE 1.0 'Time [s]'
COUR 1 'DisCen' DEPL COMP 3 POIN LECT cen TERM
TRAC 1 AXES 1.0 'DisCen' YZER
     COLO bleu
     THIC 0.8
LIST 1 AXES 1.0 'DisCen' YZER
COUR 2 'VitCen' VITE COMP 3 POIN LECT cen TERM
TRAC 2 AXES 1.0 'VITCen' YZER
     COLO bleu
     THIC 0.8
LIST 2 AXES 1.0 'VitCen' YZER
FIN
```

osto03c.epx

```
OSTO03C
ECHO
!CONV WIN
RESU SPLI ALIC 'osto03.ali' GARD PSCR
SORT VISU NSTO 1
PLAY
CAME   1 EYE  -3.18123E-01 -3.98270E-01  3.11803E-01
!        Q     7.92529E-01  5.04896E-01 -1.83767E-01 -2.88457E-01
         VIEW  5.82563E-01  6.94272E-01 -4.22618E-01
         RIGH  7.66044E-01 -6.42788E-01  0.00000E+00
         UP    2.71654E-01  3.23744E-01  9.06308E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.00092E-01  1.00140E-01  8.40999E-03
!RSPHERE:  1.43578E-01
!RADIUS :  7.17889E-01
!ASPECT :  1.00000E+00
!NEAR   :  5.59953E-01
!FAR    :  1.00504E+00
SCEN GEOM NAVI FREE
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 101 FPS 15 KFRE 10 COMP -1 NFAI
     OBJE LECT plate TERM REND
GOTR LOOP 99 OFFS FICH AVI CONT NOCL NFAI
     OBJE LECT plate TERM REND
GO
TRAC OFFS FICH AVI CONT NFAI
     OBJE LECT plate TERM REND
ENDPLAY
FIN
```

osto04.dgibi

```
'DEBPROC' pxbox3d x0*'FLOTTANT' y0*'FLOTTANT' z0*'FLOTTANT'
                  lx*'FLOTTANT' ly*'FLOTTANT' lz*'FLOTTANT'
                  dd*'FLOTTANT';
*
*--------------------------------------------------
* Generates a parallelepiped mesh with origin in point
* (x0,y0,z0), sides of length (lx,ly,lz) and density (mesh size) dd.
* The mesh consists of CUB8 hexahedral elements and is oriented
* along the global axes.
*
* Input :
* -----
*     x0,y0,z0 : coordinates of 'origin' of the box
*     lx,ly,lz : length of the box sides
*     dd : "density" (size) of the mesh (the same in all directions)
* Output :
* ------
*     box : mesh consisting of CUB8 hexahedra
*--------------------------------------------------
*
dens dd;
p1 = x0 y0 z0;
p2 = (x0 + lx) y0 z0;
p3 = (x0 + lx) (y0 + ly) z0;
p4 = x0 (y0 + ly) z0;
*
c1 = p1 d p2;
c2 = p2 d p3;
c3 = p3 d p4;
c4 = p4 d p1;
base = dall c1 c2 c3 c4 plan;
*
box = base volu tran (0 0 lz);
*
finproc box;
*-----------------------------------------------------------------------
'DEBPROC' pxrec3d x0*'FLOTTANT' y0*'FLOTTANT' z0*'FLOTTANT'
                  lx*'FLOTTANT' ly*'FLOTTANT'
                  dd*'FLOTTANT';
*
*--------------------------------------------------
* Generates a rectangle mesh with origin in point
* (x0,y0,z0), sides of length (lx,ly) and density (mesh size) dd.
* The mesh consists of QUA4 quadrilateral elements and is oriented
* along the global axes.
*
* Input :
* -----
*     x0,y0,z0 : coordinates of 'origin' of the box
*     lx,ly    : length of the box sides
*     dd : "density" (size) of the mesh (the same in all directions)
* Output :
* ------
*     box : mesh consisting of CUB8 hexahedra
*--------------------------------------------------
*
dens dd;
p1 = x0 y0 z0;
p2 = (x0 + lx) y0 z0;
p3 = (x0 + lx) (y0 + ly) z0;
p4 = x0 (y0 + ly) z0;
*
c1 = p1 d p2;
c2 = p2 d p3;
c3 = p3 d p4;
c4 = p4 d p1;
```

```
rect = dall c1 c2 c3 c4 plan;
*
finproc rect;
*-----------------------------------------------------------------------
*$$$$ PX4CIR3D
*
* Pour generer le maillage 3D (plan) d'un quart de cercle
* avec seulement des quadrilateres a 4 noeuds.
* Le quart de cercle est defini par les deux extremes
* d'un arc (de 90 degrees), par le centre du cercle
* et par un autre point qui definit l'axe de rotation
* (axe perpendiculaire au plan du cercle, passant pour son centre).
*
* Input:
* =====
* P1 = premiere extremite de l'arc
* P2 = deuxieme extremite de l'arc
* PC = centre de l'arc
* PZ = autre point de l'axe
* N  = nombre de mailles a generer sur chaque cote (doit etre pair)
* TOL= tolerance pour l'elimination des noeuds doubles
*
* Output:
* ======
* SUR = objet MAILLAGE d'elements de type QUA4
* IER = 0: pas d'erreur, .NE.0: erreur dans la generation de SUR
*
'DEBPROC' PX4CIR3D P1*'POINT' P2*'POINT' PC*'POINT' PZ*'POINT'
                   N*'ENTIER' TOL*'FLOTTANT';
*------------------------------------------------------------------
*
ier=0;
n2 = n / 2;
p0 = 0 0 0;
pm1 = p1 plus p0;
depl pm1 tour 45 pc pz;
pm2 = 0.5*(pc plus p2);
pm3 = 0.5*(pc plus p1);
pm = 0.5*(pc plus pm1);
c1a = cerc n2 p1 pc pm1;
c1b = cerc n2 pm1 pc p2;
c2a = droi n2 p2 pm2;
c2b = droi n2 pm2 pc;
c3a = droi n2 pc pm3;
c3b = droi n2 pm3 p1;
c4a = droi n2 pm pm1;
c4b = droi n2 pm pm2;
c4c = droi n2 pm pm3;
sur1 = dall plan c4c c3b c1a (inve c4a);
sur2 = dall plan c4a c1b c2a (inve c4b);
sur3 = dall plan c2b c3a (inve c4c) c4b;
sur = sur1 et sur2 et sur3;
*
elim tol sur;
*
'FINPROC' sur ier;
*-----------------------------------------------------------------------
*
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra 'osto04_mesh.ps';
opti sauv form 'osto04.msh';
*
dd = 0.0025;
dens dd;
tol = 0.1*dd;
*
* Plate
*
pla1 = pxrec3d 0.0    0.0    0.0 0.15   0.15   dd;
pla2 = pxrec3d 0.15   0.15   0.0 0.05   0.05   dd;
pla3 = pxrec3d 0.0875 0.15   0.0 0.0125 0.05   dd;
pla4 = pxrec3d 0.03   0.15   0.0 0.0075 0.05   dd;
pla5 = pxrec3d 0.15   0.0875 0.0 0.05   0.0125 dd;
pla6 = pxrec3d 0.15   0.03   0.0 0.05   0.0075 dd;
*
p0 = 0 0 0;
cont = p0 d (0.025 0 0) d (0.025 0.0185 0) c 4 (0.025 0.025 0)
       (0.0185 0.025 0) d (0 0.025 0) d p0;
qua1 = cont surf plan;
qua2 = qua1 syme plan (0.025 0 0) (0.025 0.025 0) (0.025 0 0.025);
qua12 = qua1 et qua2;
qua34 = qua12 syme plan (0 0.025 0) (0.025 0.025 0) (0 0.025 0.025);
q1 = qua12 et qua34;
elim tol q1;
orie q1 dire (0 0 1);
*
pla7 =  q1 plus (0.0375 0.15   0);
pla8 =  q1 plus (0.10   0.15   0);
pla9 =  q1 plus (0.15   0.10   0);
depl pla9 tour 90 (0.175 0.125 0) (0.175 0.125 1);
pla10 = q1 plus (0.15   0.0375 0);
depl pla10 tour 90 (0.175 0.0625 0) (0.175 0.0625 1);
*
dens dd;
con5 = p0 d (0.030 0 0) d (0.030 0.025 0) d (0.012 0.025 0)
       c (0 0.025 0) (0 0.013 0) d p0;
qua5 = con5 surf plan;
orie qua5 dire (0 0 1);
qua6 = qua5 syme plan (0 0.025 0) (0.030 0.025 0) (0 0.025 0.025);
orie qua6 dire (0 0 1);
q56 = qua5 et qua6;
elim tol q56;
orie q56 dire (0 0 1);
*
pla11 = q56 plus (0 0.15 0);
pla12 = q56 plus (0 0 0);
depl pla12 tour 90 (0 0.025 0) (0 0.025 0.025);
depl pla12 plus (0.175 -0.025 0);
*
plate = pla1 et pla2 et pla3 et pla4 et pla5 et pla6 et
        pla7 et pla8 et pla9 et pla10 et pla11 et pla12;
elim tol plate;
orie plate dire (0 0 1);
trac qual cach plate;
*
platet = plate elem tri3;
orie platet dire (0 0 1);
plateq = plate elem qua4;
orie plateq dire (0 0 1);
*
* Lower frame
*
blf = (plate diff pla1) plus (0 0 -0.0154);
lframe = blf volu tran (0 0 0.015);
trac qual cach lframe;
lfra8 = lframe elem cub8;
lfra6 = lframe elem pri6;
*
* Upper frame (without the bolts)
*
uf2 = pla2 plus (0 0 0.0004);
uf3 = pla3 plus (0 0 0.0004);
uf4 = pla4 plus (0 0 0.0004);
uf5 = pla5 plus (0 0 0.0004);
uf6 = pla6 plus (0 0 0.0004);
uf11 = pla11 plus (0 0 0.0004);
uf12 = pla12 plus (0 0 0.0004);
couf = p0 d (0.025 0 0) d (0.025 0.0190 0) c 4 (0.025 0.025 0)
       (0.0190 0.025 0) d (0 0.025 0) d p0;
quf1 = couf surf plan;
quf2 = quf1 syme plan (0.025 0 0) (0.025 0.025 0) (0.025 0 0.025);
quf12 = quf1 et quf2;
quf34 = quf12 syme plan (0 0.025 0) (0.025 0.025 0) (0 0.025 0.025);
u1 = quf12 et quf34;
elim tol u1;
orie u1 dire (0 0 1);
*
uf7 =  u1 plus (0.0375 0.15   0.0004);
uf8 =  u1 plus (0.10   0.15   0.0004);
uf9 =  u1 plus (0.15   0.10   0.0004);
uf10 = u1 plus (0.15   0.0375 0.0004);
*
buf = uf2 et uf3 et uf4 et uf5 et uf6 et
      uf7 et uf8 et uf9 et uf10 et uf11 et uf12;
elim tol buf;
uframe = buf volu tran (0 0 0.015);
trac qual cach uframe;
ufra8 = uframe elem cub8;
ufra6 = uframe elem pri6;
*
* Bolts
*
n = 4;
pz = 0 0 1;
b1 ier = px4cir3d (0.006 0 0) (0 0.006 0) p0 pz n tol;
b2 = b1 tour 90 p0 pz;
b3 = b2 tour 90 p0 pz;
b4 = b3 tour 90 p0 pz;
basu = b1 et b2 et b3 et b4;
elim tol basu;
basl = basu plus p0;
depl basu plus (0 0 0.0004);
bolu = basu volu tran 6 (0 0 0.0150);
boll = basl volu tran 6 (0 0 0.0158);
depl boll plus (0 0 -0.0154);
bolt = boll et bolu;
elim tol bolt;
trac qual cach bolt;
*
bolt1 = bolt plus (0.0625 0.1750 0);
bolt2 = bolt plus (0.1250 0.1750 0);
bolt3 = bolt plus (0.1750 0.1250 0);
bolt4 = bolt plus (0.1750 0.0625 0);
bolts = bolt1 et bolt2 et bolt3 et bolt4;
*
uframeb = uframe et bolts;
elim tol uframeb;
*
* Pressurized surfaces
*
pairb = pla1 coul noir;
ppimp = blf coul noir;
ppimp3 = ppimp elem tri3;
ppimp4 = ppimp elem qua4;
trac qual cach (pairb et ppimp);
*
mesh = plate et lframe et uframeb et pairb et ppimp;
tass mesh noop;
sauv form mesh;
trac qual cach mesh;
*
fin;
```

osto04.epx

```
OSTO04
ECHO
!CONV WIN
CAST mesh
EROS 1.0
TRID LAGR
GEOM CUB8 lfra8 ufra8 bolts
     PR6  lfra6 ufra6
     Q4GS plateq
     T3GS platet
     CL3D pairb ppimp4
     CL3T ppimp3
TERM
COMP ORIE OBJE LECT plate TERM POIN 0 0 1
     EPAI 0.0008 LECT plate TERM
     GROU 7 'pbol1'  LECT ppimp TERM
                     COND SPHE XC 0.0625 YC 0.175  ZC -0.0154 R 10.E-3
            'pbol2'  LECT ppimp TERM
                     COND SPHE XC 0.125  YC 0.175  ZC -0.0154 R 10.E-3
            'pbol3'  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.125  ZC -0.0154 R 10.E-3
            'pbol4'  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.0625 ZC -0.0154 R 10.E-3
            'pbolts' LECT pbol1 pbol2 pbol3 pbol4 TERM
            'PrCen' LECT pairb TERM
                     COND NEAR POIN 0.0 0.0 0.0
            'PrSen6' LECT pairb TERM
                   COND NEAR POIN 0.175 0.0 0.0
     NGRO 7 'bloc'   LECT uframe TERM COND Z GT 0.01539
            'symx'   LECT mesh TERM COND X LT 0.0001
            'symy'   LECT mesh TERM COND Y LT 0.0001
            'cen'    LECT plate TERM COND NEAR POIN 0.0 0.0 0.0
            'axis1'   LECT plate TERM COND X LT 0.0001
            'axis2'   LECT plate TERM COND Y LT 0.0001
            'plateN'   LECT plate diff axis1 axis2 TERM
     COUL VERT LECT plate  TERM
          ROSE LECT lframe TERM
          TURQ LECT uframe bolts TERM
          ROUG LECT pairb TERM
          JAUN LECT pbolts TERM
          GR50 LECT ppimp DIFF pbolts TERM
MATE VPJC RO 2700.0 YOUN 70.0E9 NU 0.3 ELAS 80.0E6 mxit 500
          QR1 49.3E6 CR1 1457.1 QR2 5.2E6 CR2 121.5
          PDOT 5.E-4 C 0.014 TQ 0.9 CP 910.0
          TM 893.0 M 0.0 DC 1.0 WC 65.0E6
     LECT plate TERM
     VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 3.7E8 mxit 500
          QR1 2.364E8 CR1 39.3 QR2 4.081E8 CR2 4.5
          PDOT 5.E-4 C 1.E-3 TQ 0.9 CP 452.0
          TM 1800.0 M 0.0 DC 0.9 WC 473.0E6
          LECT lframe uframe bolts TERM
     IMPE PIMP RO 7850.0 PRES 0 PREF 0
          LECT ppimp diff pbol1 pbol2 pbol3 pbol4 TERM
     IMPE PIMP RO 7850.0 PRES 527.e6 PREF 0
          LECT pbol1 pbol2 pbol3 pbol4 TERM
     IMPE AIRB X 0.0 Y 0.0 Z -0.625
          MASS 0.0402 TAUT ANGL COEF 1.0
          LECT pairb TERM
LINK COUP SPLT NONE SOLV PARD
     BLOQ 123 LECT bloc TERM
     CONT SPLA NX 1 NY 0 NZ 0 LECT symx TERM
     CONT SPLA NX 0 NY 1 NZ 0 LECT symy TERM
     GLIS 2 FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #1
            PGAP 0.4E-3
            MAIT LECT lframe TERM
            PESC LECT plateN  TERM
            FROT MUST 0.15 MUDY 0.1 GAMM 0 ! Contact surface #2
            PGAP 0.4E-3
            MAIT LECT uframeb TERM
            PESC LECT plateN   TERM
ECRI VITE TFRE 1.E-3
          POIN LECT 1 TERM
          NOEL
     FICH ALIT TFREQ 0.1E-4
          POIN LECT cen axis1 axis2 TERM
          ELEM LECT PrCen PrSen6 TERM
     FICH SPLI ALIC TFRE 1.0E-4
OPTI NOTE
     JAUM
     CSTA 0.5
     GLIS NORM ELEM
     LOG 1
     LMST
CALC TINI 0.0 TEND 10.E-3
FIN
```

osto04a.epx

```
OSTO04A
ECHO
!CONV WIN
RESU SPLI ALIC 'osto04.ali' GARD PSCR
SORT VISU NSTO 1
PLAY
CAME   1 EYE  -3.18123E-01 -3.98270E-01  3.11803E-01
!        Q     7.92529E-01  5.04896E-01 -1.83767E-01 -2.88457E-01
         VIEW  5.82563E-01  6.94272E-01 -4.22618E-01
         RIGH  7.66044E-01 -6.42788E-01  0.00000E+00
         UP    2.71654E-01  3.23744E-01  9.06308E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.00092E-01  1.00140E-01  8.40999E-03
!RSPHERE:  1.43578E-01
!RADIUS :  7.17889E-01
!ASPECT :  1.00000E+00
!NEAR   :  5.59953E-01
!FAR    :  1.00504E+00
SCEN GEOM NAVI FREE
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 101 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 99 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
FIN
```

osto04b.epx

```
OSTO04B
ECHO
RESU ALIC TEMP 'osto04.alt' GARD PSCR
COMP GROU 7 'pbol1'  LECT ppimp TERM
                     COND SPHE XC 0.0625 YC 0.175  ZC -0.0154 R 10.E-3
            'pbol2'  LECT ppimp TERM
                     COND SPHE XC 0.125  YC 0.175  ZC -0.0154 R 10.E-3
            'pbol3'  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.125  ZC -0.0154 R 10.E-3
            'pbol4'  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.0625 ZC -0.0154 R 10.E-3
            'pbolts' LECT pbol1 pbol2 pbol3 pbol4 TERM
            'PrCen' LECT pairb TERM
                     COND NEAR POIN 0.0 0.0 0.0
            'PrSen6' LECT pairb TERM
                   COND NEAR POIN 0.175 0.0 0.0
     NGRO 7 'bloc'   LECT uframe TERM COND Z GT 0.01539
            'symx'   LECT mesh TERM COND X LT 0.0001
            'symy'   LECT mesh TERM COND Y LT 0.0001
            'cen'    LECT plate TERM COND NEAR POIN 0.0 0.0 0.0
            'axis1'   LECT plate TERM COND X LT 0.0001
            'axis2'   LECT plate TERM COND Y LT 0.0001
            'plateN'   LECT plate diff axis1 axis2 TERM
SORT GRAP
AXTE 1.0 'Time [s]'
COUR 1 'DisCen' DEPL COMP 3 POIN LECT cen TERM
TRAC 1 AXES 1.0 'DisCen' YZER
     COLO bleu
     THIC 0.8
LIST 1 AXES 1.0 'DisCen' YZER
COUR 2 'VitCen' VITE COMP 3 POIN LECT cen TERM
TRAC 2 AXES 1.0 'VITCen' YZER
     COLO bleu
     THIC 0.8
LIST 2 AXES 1.0 'VitCen' YZER
FIN
```

osto04c.epx

```
OSTO04C
ECHO
!CONV WIN
RESU SPLI ALIC 'osto04.ali' GARD PSCR
SORT VISU NSTO 1
PLAY
CAME   1 EYE  -3.18123E-01 -3.98270E-01  3.11803E-01
!        Q     7.92529E-01  5.04896E-01 -1.83767E-01 -2.88457E-01
         VIEW  5.82563E-01  6.94272E-01 -4.22618E-01
         RIGH  7.66044E-01 -6.42788E-01  0.00000E+00
         UP    2.71654E-01  3.23744E-01  9.06308E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.00092E-01  1.00140E-01  8.40999E-03
!RSPHERE:  1.43578E-01
!RADIUS :  7.17889E-01
!ASPECT :  1.00000E+00
!NEAR   :  5.59953E-01
!FAR    :  1.00504E+00
SCEN GEOM NAVI FREE
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 101 FPS 15 KFRE 10 COMP -1 NFAI
     OBJE LECT plate TERM REND
GOTR LOOP 99 OFFS FICH AVI CONT NOCL NFAI
     OBJE LECT plate TERM REND
GO
TRAC OFFS FICH AVI CONT NFAI
     OBJE LECT plate TERM REND
ENDPLAY
FIN
```

osto05.dgibi

```
’DEBPROC’ pxbox3d x0*’FLOTTANT’ y0*’FLOTTANT’ z0*’FLOTTANT’
                  lx*’FLOTTANT’ ly*’FLOTTANT’ lz*’FLOTTANT’
                  dd*’FLOTTANT’;
*
*--------------------------------------------------
* Generates a parallelepiped mesh with origin in point
* (x0,y0,z0), sides of length (lx,ly,lz) and density (mesh size) dd.
* The mesh consists of CUB8 hexahedral elements and is oriented
* along the global axes.
*
* Input :
* -----
*     x0,y0,z0 : coordinates of ’origin’ of the box
*     lx,ly,lz : length of the box sides
*     dd : "density" (size) of the mesh (the same in all directions)
* Output :
* ------
*     box : mesh consisting of CUB8 hexahedra
*--------------------------------------------------
*
dens dd;
p1 = x0 y0 z0;
p2 = (x0 + lx) y0 z0;
p3 = (x0 + lx) (y0 + ly) z0;
p4 = x0 (y0 + ly) z0;
*
c1 = p1 d p2;
c2 = p2 d p3;
c3 = p3 d p4;
c4 = p4 d p1;
base = dall c1 c2 c3 c4 plan;
*
box = base volu tran (0 0 lz);
*
finproc box;
*-----------------------------------------------------------------------
’DEBPROC’ pxrec3d x0*’FLOTTANT’ y0*’FLOTTANT’ z0*’FLOTTANT’
                  lx*’FLOTTANT’ ly*’FLOTTANT’
                  dd*’FLOTTANT’;
*
*--------------------------------------------------
* Generates a rectangle mesh with origin in point
* (x0,y0,z0), sides of length (lx,ly) and density (mesh size) dd.
* The mesh consists of QUA4 quadrilateral elements and is oriented
* along the global axes.
*
* Input :
* -----
*     x0,y0,z0 : coordinates of ’origin’ of the box
*     lx,ly    : length of the box sides
*     dd : "density" (size) of the mesh (the same in all directions)
* Output :
* ------
*     box : mesh consisting of CUB8 hexahedra
*--------------------------------------------------
*
dens dd;
p1 = x0 y0 z0;
p2 = (x0 + lx) y0 z0;
p3 = (x0 + lx) (y0 + ly) z0;
p4 = x0 (y0 + ly) z0;
*
c1 = p1 d p2;
c2 = p2 d p3;
c3 = p3 d p4;
c4 = p4 d p1;
rect = dall c1 c2 c3 c4 plan;
*
finproc rect;
*-----------------------------------------------------------------------
*$$$$ PX4CIR3D
*
* Pour generer le maillage 3D (plan) d’un quart de cercle
* avec seulement des quadrilateres a 4 noeuds.
* Le quart de cercle est defini par les deux extremes
* d’un arc (de 90 degrees), par le centre du cercle
* et par un autre point qui definit l’axe de rotation
* (axe perpendiculaire au plan du cercle, passant pour son centre).
*
* Input:
* =====
* P1 = premiere extremite de l’arc
* P2 = deuxieme extremite de l’arc
* PC = centre de l’arc
* PZ = autre point de l’axe
* N  = nombre de mailles a generer sur chaque cote (doit etre pair)
* TOL= tolerance pour l’elimination des noeuds doubles
*
* Output:
* ======
* SUR = objet MAILLAGE d’elements de type QUA4
* IER = 0: pas d’erreur, .NE.0: erreur dans la generation de SUR
*
’DEBPROC’ PX4CIR3D P1*’POINT’ P2*’POINT’ PC*’POINT’ PZ*’POINT’
                   N*’ENTIER’ TOL*’FLOTTANT’;
*------------------------------------------------------------------
*
ier=0;
n2 = n / 2;
p0 = 0 0 0;
pm1 = p1 plus p0;
depl pm1 tour 45 pc pz;
pm2 = 0.5*(pc plus p2);
pm3 = 0.5*(pc plus p1);
pm = 0.5*(pc plus pm1);
c1a = cerc n2 p1 pc pm1;
c1b = cerc n2 pm1 pc p2;
c2a = droi n2 p2 pm2;
c2b = droi n2 pm2 pc;
c3a = droi n2 pc pm3;
c3b = droi n2 pm3 p1;
c4a = droi n2 pm pm1;
c4b = droi n2 pm pm2;
c4c = droi n2 pm pm3;
sur1 = dall plan c4c c3b c1a (inve c4a);
sur2 = dall plan c4a c1b c2a (inve c4b);
sur3 = dall plan c2b c3a (inve c4c) c4b;
sur = sur1 et sur2 et sur3;
*
elim tol sur;
*
’FINPROC’ sur ier;
*-----------------------------------------------------------------------
*
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra ’osto05_mesh.ps’;
opti sauv form ’osto05.msh’;
*
*dd = 0.0025;
dd = 0.00625;
dens dd;
tol = 0.1*dd;
*
* Plate
*
pla1 = pxrec3d 0.0    0.0    0.0 0.15   0.15   dd;
pla2 = pxrec3d 0.15   0.15   0.0 0.05   0.05   dd;
pla3 = pxrec3d 0.0875 0.15   0.0 0.0125 0.05   dd;
pla4 = pxrec3d 0.03   0.15   0.0 0.0075 0.05   dd;
pla5 = pxrec3d 0.15   0.0875 0.0 0.05   0.0125 dd;
pla6 = pxrec3d 0.15   0.03   0.0 0.05   0.0075 dd;
trac qual cach (pla3 ET pla4);
*
p0 = 0 0 0;
***/*/*/*/*/cont = p0 d (0.025 0 0) d (0.025 0.0185 0) c 4 (0.025 0.025 0)
cont = p0 d (0.025 0 0) d (0.025 0.0185 0) c 2 (0.025 0.025 0)
       (0.0185 0.025 0) d (0 0.025 0) d p0;
trac qual cach cont;
qua1 = cont surf plan;
trac qual cach qua1;
qua2 = qua1 syme plan (0.025 0 0) (0.025 0.025 0) (0.025 0 0.025);
qua12 = qua1 et qua2;
qua34 = qua12 syme plan (0 0.025 0) (0.025 0.025 0) (0 0.025 0.025);
q1 = qua12 et qua34;
elim tol q1;
trac qual cach q1;
orie q1 dire (0 0 1);
*
pla7 =  q1 plus (0.0375 0.15   0);
pla8 =  q1 plus (0.10   0.15   0);
pla9 =  q1 plus (0.15   0.10   0);
depl pla9 tour 90 (0.175 0.125 0) (0.175 0.125 1);
pla10 = q1 plus (0.15   0.0375 0);
depl pla10 tour 90 (0.175 0.0625 0) (0.175 0.0625 1);
*
dens dd;
*con5 = p0 d (0.030 0 0) d (0.030 0.025 0) d (0.012 0.025 0)
*       c (0 0.025 0) (0 0.013 0) d p0;
con5 = p0 d (0.0375 0 0) d (0.0375 0.025 0) d (0.012 0.025 0)
       c 2 (0 0.025 0) (0 0.013 0) d p0;
trac qual cach con5;
qua5 = con5 surf plan;
trac qual cach qua5;
orie qua5 dire (0 0 1);
qua6 = qua5 syme plan (0 0.025 0) (0.030 0.025 0) (0 0.025 0.025);
orie qua6 dire (0 0 1);
q56 = qua5 et qua6;
elim tol q56;
trac qual cach q56;
orie q56 dire (0 0 1);
*
pla11 = q56 plus (0 0.15 0);
pla12 = q56 plus (0 0 0);
depl pla12 tour 90 (0 0.025 0) (0 0.025 0.025);
depl pla12 plus (0.175 -0.025 0);
*
*plate = pla1 et pla2 et pla3 et pla4 et pla5 et pla6 et
*        pla7 et pla8 et pla9 et pla10 et pla11 et pla12;
plate = pla1 et pla2 et pla3 et pla5 et
        pla7 et pla8 et pla9 et pla10 et pla11 et pla12;
elim tol plate;
orie plate dire (0 0 1);
trac qual cach plate;
*
*platet = plate elem tri3;
*orie platet dire (0 0 1);
plateq = plate elem qua4;
orie plateq dire (0 0 1);
nplate = chan POI1 plate;
*
* Lower frame
*
blf = (plate diff pla1) plus (0 0 -0.0154);
```

```
lframe = blf volu tran (0 0 0.015);
trac qual cach lframe;
lfra8 = lframe elem cub8;
*lfra6 = lframe elem pri6;
*
* Upper frame (without the bolts)
*
uf2 = pla2 plus (0 0 0.0004);
uf3 = pla3 plus (0 0 0.0004);
uf4 = pla4 plus (0 0 0.0004);
uf5 = pla5 plus (0 0 0.0004);
uf6 = pla6 plus (0 0 0.0004);
uf11 = pla11 plus (0 0 0.0004);
uf12 = pla12 plus (0 0 0.0004);
couf = p0 d (0.025 0 0) d (0.025 0.0190 0) c 2 (0.025 0.025 0)
       (0.0190 0.025 0) d (0 0.025 0) d p0;
quf1 = couf surf plan;
quf2 = quf1 syme plan (0.025 0 0) (0.025 0.025 0) (0.025 0 0.025);
quf12 = quf1 et quf2;
quf34 = quf12 syme plan (0 0.025 0) (0.025 0.025 0) (0 0.025 0.025);
u1 = quf12 et quf34;
elim tol u1;
orie u1 dire (0 0 1);
*
uf7 =  u1 plus (0.0375 0.15   0.0004);
uf8 =  u1 plus (0.10   0.15   0.0004);
uf9 =  u1 plus (0.15   0.10   0.0004);
uf10 = u1 plus (0.15   0.0375 0.0004);
*
*buf = uf2 et uf3 et uf4 et uf5 et uf6 et
*      uf7 et uf8 et uf9 et uf10 et uf11 et uf12;
buf = uf2 et uf3 et uf5 et
      uf7 et uf8 et uf9 et uf10 et uf11 et uf12;
elim tol buf;
uframe = buf volu tran (0 0 0.015);
trac qual cach uframe;
ufra8 = uframe elem cub8;
*ufra6 = uframe elem pri6;
*
* Bolts
*
*n = 4;
n = 2;
pz = 0 0 1;
b1 ier = px4cir3d (0.006 0 0) (0 0.006 0) p0 pz n tol;
b2 = b1 tour 90 p0 pz;
b3 = b2 tour 90 p0 pz;
b4 = b3 tour 90 p0 pz;
basu = b1 et b2 et b3 et b4;
elim tol basu;
basl = basu plus p0;
depl basu plus (0 0 0.0004);
bolu = basu volu tran 2 (0 0 0.0150);
boll = basl volu tran 2 (0 0 0.0158);
depl boll plus (0 0 -0.0154);
bolt = boll et bolu;
elim tol bolt;
trac qual cach bolt;
*
bolt1 = bolt plus (0.0625 0.1750 0);
bolt2 = bolt plus (0.1250 0.1750 0);
bolt3 = bolt plus (0.1750 0.1250 0);
bolt4 = bolt plus (0.1750 0.0625 0);
bolts = bolt1 et bolt2 et bolt3 et bolt4;
*
uframeb = uframe et bolts;
elim tol uframeb;
*
* Pressurized surfaces
*
pairb = pla1 coul noir;
ppimp = blf coul noir;
*ppimp3 = ppimp elem tri3;
ppimp4 = ppimp elem qua4;
trac qual cach (pairb et ppimp);
*
mesh = plate et lframe et uframeb et pairb et ppimp et nplate;
tass mesh noop;
sauv form mesh;
trac qual cach mesh;
*
fin;
```

osto05.epx

```
OSTO05
ECHO
!CONV WIN
CAST mesh
EROS 1.0
TRID LAGR
DIME
  ADAP NPOI 7000 Q4GS 7000 CL3D 7000 !NPIN 7000
  ENDA
TERM
GEOM CUB8 lfra8 ufra8 bolts
     Q4GS plateq
     PMAT nplate
     CL3D pairb ppimp4
TERM
COMP ORIE OBJE LECT plate TERM POIN 0 0 1
     EPAI 0.0008 LECT plate nplate _q4gs TERM
     GROU 9 ’pbol1’  LECT ppimp TERM
                     COND SPHE XC 0.0625 YC 0.175  ZC -0.0154 R 10.E-3
            ’pbol2’  LECT ppimp TERM
                     COND SPHE XC 0.125  YC 0.175  ZC -0.0154 R 10.E-3
            ’pbol3’  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.125  ZC -0.0154 R 10.E-3
            ’pbol4’  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.0625 ZC -0.0154 R 10.E-3
            ’pbolts’ LECT pbol1 pbol2 pbol3 pbol4 TERM
            ’PrCen’ LECT pairb TERM
                     COND NEAR POIN 0.0 0.0 0.0
            ’PrSen6’ LECT pairb TERM
                   COND NEAR POIN 0.175 0.0 0.0
            ’pinada’ LECT plate TERM
                   COND BOX X0 0.0 Y0 0.0  Z0 -0.1
                         DX 0.15 DY 0.15 DZ 0.2
            ’pinade’ LECT plate TERM
                   COND BOX X0 0.0 Y0 0.0  Z0 -0.1
                         DX 0.18 DY 0.18 DZ 0.2
     NGRO 9 ’bloc’   LECT uframe TERM COND Z GT 0.01539
            ’symx’   LECT mesh TERM COND X LT 0.0001
            ’symy’   LECT mesh TERM COND Y LT 0.0001
            ’symxP’   LECT plate TERM COND X LT 0.0001
            ’symyP’   LECT plate TERM COND Y LT 0.0001
            ’cen’    LECT plate TERM COND NEAR POIN 0.0 0.0 0.0
            ’axis1’   LECT plate TERM COND X LT 0.0001
            ’axis2’   LECT plate TERM COND Y LT 0.0001
            ’plateN’   LECT plate diff axis1 axis2 TERM
     COUL VERT LECT plate  TERM
          ROSE LECT lframe TERM
          TURQ LECT uframe bolts TERM
          ROUG LECT pairb nplate TERM
          JAUN LECT pbolts TERM
          GR50 LECT ppimp DIFF pbolts TERM
 ADAP THRS ECRO 11 TMIN 0.05 TMAX 0.25 MAXL 3
           LECT pinade TERM
MATE VPJC RO 2700.0 YOUN 70.0E9 NU 0.3 ELAS 80.0E6 mxit 500
          QR1 49.3E6 CR1 1457.1 QR2 5.2E6 CR2 121.5
          PDOT 5.E-4 C 0.014 TQ 0.9 CP 910.0
          TM 893.0 M 0.0 DC 1.0 WC 65.0E6
     LECT plate _q4gs TERM
     VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 3.7E8 mxit 500
          QR1 2.364E8 CR1 39.3 QR2 4.081E8 CR2 4.5
          PDOT 5.E-4 C 1.E-3 TQ 0.9 CP 452.0
          TM 1800.0 M 0.0 DC 0.9 WC 473.0E6
          LECT lframe uframe bolts TERM
     MASS 0.0 YOUN 70.0E9 NU 0.3
          LECT nplate TERM
     IMPE PIMP RO 7850.0 PRES 0 PREF 0
          LECT ppimp diff pbol1 pbol2 pbol3 pbol4 TERM
     IMPE PIMP RO 7850.0 PRES 340.e6 PREF 0
          LECT pbol1 pbol2 pbol3 pbol4 TERM
     IMPE AIRB X 0.0 Y 0.0 Z -0.375
          MASS 0.0402 TAUT ANGL COEF 1.0
          LECT pairb _cl3d TERM
OPTI PINS ASN
LINK COUP SPLT NONE SOLV PARD
     BLOQ 123 LECT bloc TERM
     BLOQ 1 LECT symx TERM
     BLOQ 2 LECT symy TERM
     BLOQ 56 LECT symxP TERM
     BLOQ 46 LECT symyP TERM
!    GLIS 2 FROT MUST 0.1 MUDY 0.1 GAMM 0 ! Contact surface #1
!           PGAP 0.4E-3
!           MAIT LECT lframe TERM
!           PESC LECT plateN  TERM
!           FROT MUST 0.1 MUDY 0.1 GAMM 0 ! Contact surface #2
!           PGAP 0.4E-3
!           MAIT LECT uframeb TERM
!           PESC LECT plateN   TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY FROT MUST 0.1 MUDY 0.1 GAMM 0 MLEV 3
          LECT lframe TERM
          BODY FROT MUST 0.1 MUDY 0.1 GAMM 0 MLEV 3
          LECT uframeb TERM
          BODY FROT MUST 0.1 MUDY 0.1 GAMM 0 DIAM 0.0008
          LECT nplate TERM
ECRI VITE TFRE 1.E-3
          POIN LECT 1 TERM
          NOEL
     FICH ALIT TFREQ 0.1E-4
          POIN LECT cen axis1 axis2 TERM
          ELEM LECT PrCen PrSen6 TERM
     FICH SPLI ALIC TFRE 1.0E-4
OPTI NOTE
     JAUM
     CSTA 0.5
     GLIS NORM ELEM
     LOG 1
     LMST
     ADAP RCON
MEAS ! Measurement commands (batch)
  OBJE LECT pbolts TERM
  EMIN LECT pbolts TERM
  EMAX LECT pbolts TERM
CALC TINI 0.0 TEND 10.E-3
FIN
```

osto05a.epx

```
OSTO05A
ECHO
!CONV WIN
RESU SPLI ALIC ’osto05.ali’ GARD PSCR
SORT VISU NSTO 1
PLAY
CAME   1 EYE  -3.18123E-01 -3.98270E-01  3.11803E-01
!        Q     7.92529E-01  5.04896E-01 -1.83767E-01 -2.88457E-01
         VIEW  5.82563E-01  6.94272E-01 -4.22618E-01
         RIGH  7.66044E-01 -6.42788E-01  0.00000E+00
         UP    2.71654E-01  3.23744E-01  9.06308E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.00092E-01  1.00140E-01  8.40999E-03
!RSPHERE:  1.43578E-01
!RADIUS :  7.17889E-01
!ASPECT :  1.00000E+00
!NEAR   :  5.59953E-01
!FAR    :  1.00504E+00
SCEN GEOM NAVI FREE
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 76 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 74 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
FIN
```

osto05b.epx

```
OSTO05B
ECHO
RESU ALIC TEMP ’osto05.alt’ GARD PSCR
COMP GROU 9 ’pbol1’  LECT ppimp TERM
                     COND SPHE XC 0.0625 YC 0.175  ZC -0.0154 R 10.E-3
            ’pbol2’  LECT ppimp TERM
                     COND SPHE XC 0.125  YC 0.175  ZC -0.0154 R 10.E-3
            ’pbol3’  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.125  ZC -0.0154 R 10.E-3
            ’pbol4’  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.0625 ZC -0.0154 R 10.E-3
            ’pbolts’ LECT pbol1 pbol2 pbol3 pbol4 TERM
            ’PrCen’ LECT pairb TERM
                     COND NEAR POIN 0.0 0.0 0.0
            ’PrSen6’ LECT pairb TERM
                   COND NEAR POIN 0.175 0.0 0.0
            ’pinada’ LECT plate TERM
                   COND BOX X0 0.0 Y0 0.0  Z0 -0.1
                            DX 0.15 DY 0.15 DZ 0.2
            ’pinade’ LECT plate TERM
                   COND BOX X0 0.0 Y0 0.0  Z0 -0.1
                            DX 0.18 DY 0.18 DZ 0.2
     NGRO 9 ’bloc’   LECT uframe TERM COND Z GT 0.01539
            ’symx’   LECT mesh TERM COND X LT 0.0001
            ’symy’   LECT mesh TERM COND Y LT 0.0001
            ’symxP’   LECT plate TERM COND X LT 0.0001
            ’symyP’   LECT plate TERM COND Y LT 0.0001
            ’cen’    LECT plate TERM COND NEAR POIN 0.0 0.0 0.0
            ’axis1’   LECT plate TERM COND X LT 0.0001
            ’axis2’   LECT plate TERM COND Y LT 0.0001
            ’plateN’   LECT plate diff axis1 axis2 TERM
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR 1 ’DisCen’ DEPL COMP 3 POIN LECT cen TERM
TRAC 1 AXES 1.0 ’DisCen’ YZER
     COLO bleu
     THIC 0.8
LIST 1 AXES 1.0 ’DisCen’ YZER
COUR 2 ’VitCen’ VITE COMP 3 POIN LECT cen TERM
TRAC 2 AXES 1.0 ’VITCen’ YZER
     COLO bleu
     THIC 0.8
LIST 2 AXES 1.0 ’VitCen’ YZER
FIN
```

osto05c.epx

```
OSTO05C
ECHO
!CONV WIN
RESU SPLI ALIC ’osto05.ali’ GARD PSCR
SORT VISU NSTO 1
PLAY
CAME   1 EYE  -3.18123E-01 -3.98270E-01  3.11803E-01
!        Q     7.92529E-01  5.04896E-01 -1.83767E-01 -2.88457E-01
         VIEW  5.82563E-01  6.94272E-01 -4.22618E-01
         RIGH  7.66044E-01 -6.42788E-01  0.00000E+00
         UP    2.71654E-01  3.23744E-01  9.06308E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.00092E-01  1.00140E-01  8.40999E-03
!RSPHERE:  1.43578E-01
!RADIUS :  7.17889E-01
!ASPECT :  1.00000E+00
!NEAR   :  5.59953E-01
!FAR    :  1.00504E+00
SCEN GEOM NAVI FREE
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 76 FPS 15 KFRE 10 COMP -1 NFAI
     OBJE LECT plate TERM REND
GOTR LOOP 74 OFFS FICH AVI CONT NOCL NFAI
     OBJE LECT plate TERM REND
GO
TRAC OFFS FICH AVI CONT NFAI
     OBJE LECT plate TERM REND
ENDPLAY
FIN
```

osto06.dgibi

```
’DEBPROC’ pxbox3d x0*’FLOTTANT’ y0*’FLOTTANT’ z0*’FLOTTANT’
                  lx*’FLOTTANT’ ly*’FLOTTANT’ lz*’FLOTTANT’
                  dd*’FLOTTANT’;
*
*--------------------------------------------------
* Generates a parallelepiped mesh with origin in point
* (x0,y0,z0), sides of length (lx,ly,lz) and density (mesh size) dd.
* The mesh consists of CUB8 hexahedral elements and is oriented
* along the global axes.
*
* Input :
* -----
*     x0,y0,z0 : coordinates of ’origin’ of the box
*     lx,ly,lz : length of the box sides
*     dd : "density" (size) of the mesh (the same in all directions)
* Output :
* ------
*     box : mesh consisting of CUB8 hexahedra
*--------------------------------------------------
*
dens dd;
p1 = x0 y0 z0;
p2 = (x0 + lx) y0 z0;
p3 = (x0 + lx) (y0 + ly) z0;
p4 = x0 (y0 + ly) z0;
*
c1 = p1 d p2;
c2 = p2 d p3;
c3 = p3 d p4;
c4 = p4 d p1;
base = dall c1 c2 c3 c4 plan;
*
box = base volu tran (0 0 lz);
*
finproc box;
*-----------------------------------------------------------------------
’DEBPROC’ pxrec3d x0*’FLOTTANT’ y0*’FLOTTANT’ z0*’FLOTTANT’
                  lx*’FLOTTANT’ ly*’FLOTTANT’
                  dd*’FLOTTANT’;
*
*--------------------------------------------------
* Generates a rectangle mesh with origin in point
* (x0,y0,z0), sides of length (lx,ly) and density (mesh size) dd.
* The mesh consists of QUA4 quadrilateral elements and is oriented
* along the global axes.
*
* Input :
* -----
*     x0,y0,z0 : coordinates of ’origin’ of the box
*     lx,ly    : length of the box sides
*     dd : "density" (size) of the mesh (the same in all directions)
* Output :
* ------
*     box : mesh consisting of CUB8 hexahedra
*--------------------------------------------------
*
dens dd;
p1 = x0 y0 z0;
p2 = (x0 + lx) y0 z0;
p3 = (x0 + lx) (y0 + ly) z0;
p4 = x0 (y0 + ly) z0;
*
c1 = p1 d p2;
c2 = p2 d p3;
c3 = p3 d p4;
c4 = p4 d p1;
rect = dall c1 c2 c3 c4 plan;
*
finproc rect;
*-----------------------------------------------------------------------
*$$$$ PX4CIR3D
*
* Pour generer le maillage 3D (plan) d’un quart de cercle
* avec seulement des quadrilateres a 4 noeuds.
* Le quart de cercle est defini par les deux extremes
* d’un arc (de 90 degrees), par le centre du cercle
* et par un autre point qui definit l’axe de rotation
* (axe perpendiculaire au plan du cercle, passant pour son centre).
*
* Input:
* =====
* P1 = premiere extremite de l’arc
* P2 = deuxieme extremite de l’arc
* PC = centre de l’arc
```

```
* PZ = autre point de l'axe
* N  = nombre de mailles a generer sur chaque cote (doit etre pair)
* TOL= tolerance pour l'elimination des noeuds doubles
*
* Output:
* ======
* SUR = objet MAILLAGE d'elements de type QUA4
* IER = 0: pas d'erreur, .NE.0: erreur dans la generation de SUR
*
'DEBPROC' PX4CIR3D P1*'POINT' P2*'POINT' PC*'POINT' PZ*'POINT'
                   N*'ENTIER' TOL*'FLOTTANT';
*------------------------------------------------------------------
*
ier=0;
n2 = n / 2;
p0 = 0 0 0;
pm1 = p1 plus p0;
depl pm1 tour 45 pc pz;
pm2 = 0.5*(pc plus p2);
pm3 = 0.5*(pc plus p1);
pm = 0.5*(pc plus pm1);
c1a = cerc n2 p1 pc pm1;
c1b = cerc n2 pm1 pc p2;
c2a = droi n2 p2 pm2;
c2b = droi n2 pm2 pc;
c3a = droi n2 pc pm3;
c3b = droi n2 pm3 p1;
c4a = droi n2 pm pm1;
c4b = droi n2 pm pm2;
c4c = droi n2 pm pm3;
sur1 = dall plan c4c c3b c1a (inve c4a);
sur2 = dall plan c4a c1b c2a (inve c4b);
sur3 = dall plan c2b c3a (inve c4c) c4b;
sur = sur1 et sur2 et sur3;
*
elim tol sur;
*
'FINPROC' sur ier;
*-----------------------------------------------------------------------
*
opti echo 1;
opti dime 3 elem cub8;
opti trac psc ftra 'osto06_mesh.ps';
opti sauv form 'osto06.msh';
*
*dd = 0.0025;
dd = 0.00625;
dens dd;
tol = 0.1*dd;
*
* Plate
*
pla1 = pxrec3d 0.0    0.0    0.0 0.15   0.15   dd;
pla2 = pxrec3d 0.15   0.15   0.0 0.05   0.05   dd;
pla3 = pxrec3d 0.0875 0.15   0.0 0.0125 0.05   dd;
pla4 = pxrec3d 0.03   0.15   0.0 0.0075 0.05   dd;
pla5 = pxrec3d 0.15   0.0875 0.0 0.05   0.0125 dd;
pla6 = pxrec3d 0.15   0.03   0.0 0.05   0.0075 dd;
trac qual cach (pla3 ET pla4);
*
p0 = 0 0 0;
***/*/*/*/*/cont = p0 d (0.025 0 0) d (0.025 0.0185 0) c 4 (0.025 0.025 0)
cont = p0 d (0.025 0 0) d (0.025 0.0185 0) c 2 (0.025 0.025 0)
       (0.0185 0.025 0) d (0 0.025 0) d p0;
trac qual cach cont;
qua1 = cont surf plan;
trac qual cach qua1;
qua2 = qua1 syme plan (0.025 0 0) (0.025 0.025 0) (0.025 0 0.025);
qua12 = qua1 et qua2;
qua34 = qua12 syme plan (0 0.025 0) (0.025 0.025 0) (0 0.025 0.025);
q1 = qua12 et qua34;
elim tol q1;
trac qual cach q1;
orie q1 dire (0 0 1);
*
pla7 =  q1 plus (0.0375 0.15   0);
pla8 =  q1 plus (0.10   0.15   0);
pla9 =  q1 plus (0.15   0.10   0);
depl pla9 tour 90 (0.175 0.125 0) (0.175 0.125 1);
pla10 = q1 plus (0.15   0.0375 0);
depl pla10 tour 90 (0.175 0.0625 0) (0.175 0.0625 1);
*
dens dd;
*con5 = p0 d (0.030 0 0) d (0.030 0.025 0) d (0.012 0.025 0)
*       c (0 0.025 0) (0 0.013 0) d p0;
con5 = p0 d (0.0375 0 0) d (0.0375 0.025 0) d (0.012 0.025 0)
       c 2 (0 0.025 0) (0 0.013 0) d p0;
trac qual cach con5;
qua5 = con5 surf plan;
trac qual cach qua5;
orie qua5 dire (0 0 1);
qua6 = qua5 syme plan (0 0.025 0) (0.030 0.025 0) (0 0.025 0.025);
orie qua6 dire (0 0 1);
q56 = qua5 et qua6;
elim tol q56;
trac qual cach q56;
orie q56 dire (0 0 1);
*
pla11 = q56 plus (0 0.15 0);
pla12 = q56 plus (0 0 0);
depl pla12 tour 90 (0 0.025 0) (0 0.025 0.025);
depl pla12 plus (0.175 -0.025 0);
*
*plate = pla1 et pla2 et pla3 et pla4 et pla5 et pla6 et
*        pla7 et pla8 et pla9 et pla10 et pla11 et pla12;
plate = pla1 et pla2 et pla3 et pla5 et
        pla7 et pla8 et pla9 et pla10 et pla11 et pla12;
elim tol plate;
orie plate dire (0 0 1);
trac qual cach plate;
*
*platet = plate elem tri3;
*orie platet dire (0 0 1);
plateq = plate elem qua4;
orie plateq dire (0 0 1);
nplate = chan POI1 plate;
*
* Lower frame
*
blf = (plate diff pla1) plus (0 0 -0.0154);
lframe = blf volu tran (0 0 0.015);
trac qual cach lframe;
lfra8 = lframe elem cub8;
*lfra6 = lframe elem pri6;
*
* Upper frame (without the bolts)
*
uf2 = pla2 plus (0 0 0.0004);
uf3 = pla3 plus (0 0 0.0004);
uf4 = pla4 plus (0 0 0.0004);
uf5 = pla5 plus (0 0 0.0004);
uf6 = pla6 plus (0 0 0.0004);
uf11 = pla11 plus (0 0 0.0004);
uf12 = pla12 plus (0 0 0.0004);
couf = p0 d (0.025 0 0) d (0.025 0.0190 0) c 2 (0.025 0.025 0)
       (0.0190 0.025 0) d (0 0.025 0) d p0;
quf1 = couf surf plan;
quf2 = quf1 syme plan (0.025 0 0) (0.025 0.025 0) (0.025 0 0.025);
quf12 = quf1 et quf2;
quf34 = quf12 syme plan (0 0.025 0) (0.025 0.025 0) (0 0.025 0.025);
u1 = quf12 et quf34;
elim tol u1;
orie u1 dire (0 0 1);
*
uf7 =  u1 plus (0.0375 0.15   0.0004);
uf8 =  u1 plus (0.10   0.15   0.0004);
uf9 =  u1 plus (0.15   0.10   0.0004);
uf10 = u1 plus (0.15   0.0375 0.0004);
*
*buf = uf2 et uf3 et uf4 et uf5 et uf6 et
*      uf7 et uf8 et uf9 et uf10 et uf11 et uf12;
buf = uf2 et uf3 et uf5 et
      uf7 et uf8 et uf9 et uf10 et uf11 et uf12;
elim tol buf;
uframe = buf volu tran (0 0 0.015);
trac qual cach uframe;
ufra8 = uframe elem cub8;
*ufra6 = uframe elem pri6;
*
* Bolts
*
*n = 4;
n = 2;
pz = 0 0 1;
b1 ier = px4cir3d (0.006 0 0) (0 0.006 0) p0 pz n tol;
b2 = b1 tour 90 p0 pz;
b3 = b2 tour 90 p0 pz;
b4 = b3 tour 90 p0 pz;
basu = b1 et b2 et b3 et b4;
elim tol basu;
basl = basu plus p0;
depl basu plus (0 0 0.0004);
bolu = basu volu tran 2 (0 0 0.0150);
boll = basl volu tran 2 (0 0 0.0158);
depl boll plus (0 0 -0.0154);
bolt = boll et bolu;
elim tol bolt;
trac qual cach bolt;
*
bolt1 = bolt plus (0.0625 0.1750 0);
bolt2 = bolt plus (0.1250 0.1750 0);
bolt3 = bolt plus (0.1750 0.1250 0);
bolt4 = bolt plus (0.1750 0.0625 0);
bolts = bolt1 et bolt2 et bolt3 et bolt4;
*
uframeb = uframe et bolts;
elim tol uframeb;
*
* Pressurized surfaces
*
pairb = pla1 coul noir;
ppimp = blf coul noir;
*ppimp3 = ppimp elem tri3;
ppimp4 = ppimp elem qua4;
trac qual cach (pairb et ppimp);
*
mesh = plate et lframe et uframeb et pairb et ppimp et nplate;
tass mesh noop;
sauv form mesh;
trac qual cach mesh;
*
fin;
```

osto06.epx

```
OSTO06
ECHO
!CONV WIN
CAST mesh
EROS 1.0
TRID LAGR
DIME
  ADAP NPOI 7000 Q4GS 7000 CL3D 7000 !NPIN 7000
  ENDA
TERM
GEOM CUB8 lfra8 ufra8 bolts
     Q4GS plateq
     PMAT nplate
     CL3D pairb ppimp4
TERM
COMP ORIE OBJE LECT plate TERM POIN 0 0 1
     EPAI 0.0008 LECT plate nplate _q4gs TERM
     GROU 9 ’pbol1’  LECT ppimp TERM
                     COND SPHE XC 0.0625 YC 0.175  ZC -0.0154 R 10.E-3
            ’pbol2’  LECT ppimp TERM
                     COND SPHE XC 0.125  YC 0.175  ZC -0.0154 R 10.E-3
            ’pbol3’  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.125  ZC -0.0154 R 10.E-3
            ’pbol4’  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.0625 ZC -0.0154 R 10.E-3
            ’pbolts’ LECT pbol1 pbol2 pbol3 pbol4 TERM
            ’PrCen’ LECT pairb TERM
                     COND NEAR POIN 0.0 0.0 0.0
            ’PrSen6’ LECT pairb TERM
                   COND NEAR POIN 0.175 0.0 0.0
            ’pinada’ LECT plate TERM
                   COND BOX X0 0.0 Y0 0.0  Z0 -0.1
                            DX 0.15 DY 0.15 DZ 0.2
            ’pinade’ LECT plate TERM
                   COND BOX X0 0.0 Y0 0.0  Z0 -0.1
                            DX 0.18 DY 0.18 DZ 0.2
     NGRO 9 ’bloc’   LECT uframe TERM COND Z GT 0.01539
            ’symx’   LECT mesh TERM COND X LT 0.0001
            ’symy’   LECT mesh TERM COND Y LT 0.0001
            ’symxP’   LECT plate TERM COND X LT 0.0001
            ’symyP’   LECT plate TERM COND Y LT 0.0001
            ’cen’    LECT plate TERM COND NEAR POIN 0.0 0.0 0.0
            ’axis1’   LECT plate TERM COND X LT 0.0001
            ’axis2’   LECT plate TERM COND Y LT 0.0001
            ’plateN’   LECT plate diff axis1 axis2 TERM
     COUL VERT LECT plate  TERM
          ROSE LECT lframe TERM
          TURQ LECT uframe bolts TERM
          ROUG LECT pairb nplate TERM
          JAUN LECT pbolts TERM
          GR50 LECT ppimp DIFF pbolts TERM
 ADAP THRS ECRO 11 TMIN 0.05 TMAX 0.25 MAXL 3
           LECT pinade TERM
MATE VPJC RO 2700.0 YOUN 70.0E9 NU 0.3 ELAS 80.0E6 mxit 500
          QR1 49.3E6 CR1 1457.1 QR2 5.2E6 CR2 121.5
          PDOT 5.E-4 C 0.014 TQ 0.9 CP 910.0
          TM 893.0 M 0.0 DC 1.0 WC 65.0E6
     LECT plate _q4gs TERM
     VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 3.7E8 mxit 500
          QR1 2.364E8 CR1 39.3 QR2 4.081E8 CR2 4.5
          PDOT 5.E-4 C 1.E-3 TQ 0.9 CP 452.0
          TM 1800.0 M 0.0 DC 0.9 WC 473.0E6
          LECT lframe uframe bolts TERM
     MASS 0.0 YOUN 70.0E9 NU 0.3
          LECT nplate TERM
     IMPE PIMP RO 7850.0 PRES 0 PREF 0
          LECT ppimp diff pbol1 pbol2 pbol3 pbol4 TERM
     IMPE PIMP RO 7850.0 PRES 340.e6 PREF 0
          LECT pbol1 pbol2 pbol3 pbol4 TERM
     IMPE AIRB X 0.0 Y 0.0 Z -0.375
          MASS 0.0402 TAUT ANGL COEF 1.0
          LECT pairb _cl3d TERM
OPTI PINS ASN
LINK COUP SPLT NONE SOLV PARD
     BLOQ 123 LECT bloc TERM
     BLOQ 1 LECT symx TERM
     BLOQ 2 LECT symy TERM
     BLOQ 56 LECT symxP TERM
     BLOQ 46 LECT symyP TERM
!    GLIS 2 FROT MUST 0.1 MUDY 0.1 GAMM 0 ! Contact surface #1
!           PGAP 0.4E-3
!           MAIT LECT lframe TERM
!           PESC LECT plateN  TERM
!           FROT MUST 0.1 MUDY 0.1 GAMM 0 ! Contact surface #2
!           PGAP 0.4E-3
!           MAIT LECT uframeb TERM
!           PESC LECT plateN   TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY FROT MUST 0.1 MUDY 0.1 GAMM 0 MLEV 4
          LECT lframe TERM
          BODY FROT MUST 0.1 MUDY 0.1 GAMM 0 MLEV 4
          LECT uframeb TERM
          BODY FROT MUST 0.1 MUDY 0.1 GAMM 0 DIAM 0.0008
          LECT nplate TERM
ECRI VITE TFRE 1.E-3
          POIN LECT 1 TERM
          NOEL
     FICH ALIT TFREQ 0.1E-4
          POIN LECT cen axis1 axis2 TERM
          ELEM LECT PrCen PrSen6 TERM
     FICH SPLI ALIC TFRE 1.0E-4
OPTI NOTE
     JAUM
     CSTA 0.5
     GLIS NORM ELEM
     LOG 1
     LMST
     ADAP RCON
MEAS ! Measurement commands (batch)
  OBJE LECT pbolts TERM
  EMIN LECT pbolts TERM
  EMAX LECT pbolts TERM
CALC TINI 0.0 TEND 10.E-3
FIN
```

osto06a.epx

```
OSTO06A
ECHO
!CONV WIN
RESU SPLI ALIC ’osto06.ali’ GARD PSCR
SORT VISU NSTO 1
PLAY
CAME   1 EYE  -3.18123E-01 -3.98270E-01  3.11803E-01
!        Q     7.92529E-01  5.04896E-01 -1.83767E-01 -2.88457E-01
         VIEW  5.82563E-01  6.94272E-01 -4.22618E-01
         RIGH  7.66044E-01 -6.42788E-01  0.00000E+00
         UP    2.71654E-01  3.23744E-01  9.06308E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.00092E-01  1.00140E-01  8.40999E-03
!RSPHERE:  1.43578E-01
!RADIUS :  7.17889E-01
!ASPECT :  1.00000E+00
!NEAR   :  5.59953E-01
!FAR    :  1.00504E+00
SCEN GEOM NAVI FREE
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 64 FPS 15 KFRE 10 COMP -1 NFAI REND
GOTR LOOP 62 OFFS FICH AVI CONT NOCL NFAI REND
GO
TRAC OFFS FICH AVI CONT NFAI REND
ENDPLAY
FIN
```

osto06b.epx

```
OSTO06B
ECHO
RESU ALIC TEMP ’osto06.alt’ GARD PSCR
COMP GROU 9 ’pbol1’  LECT ppimp TERM
                     COND SPHE XC 0.0625 YC 0.175  ZC -0.0154 R 10.E-3
            ’pbol2’  LECT ppimp TERM
                     COND SPHE XC 0.125  YC 0.175  ZC -0.0154 R 10.E-3
            ’pbol3’  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.125  ZC -0.0154 R 10.E-3
            ’pbol4’  LECT ppimp TERM
                     COND SPHE XC 0.175  YC 0.0625 ZC -0.0154 R 10.E-3
            ’pbolts’ LECT pbol1 pbol2 pbol3 pbol4 TERM
            ’PrCen’ LECT pairb TERM
                     COND NEAR POIN 0.0 0.0 0.0
            ’PrSen6’ LECT pairb TERM
                   COND NEAR POIN 0.175 0.0 0.0
            ’pinada’ LECT plate TERM
                   COND BOX X0 0.0 Y0 0.0  Z0 -0.1
                            DX 0.15 DY 0.15 DZ 0.2
            ’pinade’ LECT plate TERM
                   COND BOX X0 0.0 Y0 0.0  Z0 -0.1
                            DX 0.18 DY 0.18 DZ 0.2
     NGRO 9 ’bloc’   LECT uframe TERM COND Z GT 0.01539
            ’symx’   LECT mesh TERM COND X LT 0.0001
            ’symy’   LECT mesh TERM COND Y LT 0.0001
            ’symxP’   LECT plate TERM COND X LT 0.0001
            ’symyP’   LECT plate TERM COND Y LT 0.0001
            ’cen’    LECT plate TERM COND NEAR POIN 0.0 0.0 0.0
            ’axis1’   LECT plate TERM COND X LT 0.0001
            ’axis2’   LECT plate TERM COND Y LT 0.0001
            ’plateN’   LECT plate diff axis1 axis2 TERM
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR 1 ’DisCen’ DEPL COMP 3 POIN LECT cen TERM
TRAC 1 AXES 1.0 ’DisCen’ YZER
     COLO bleu
     THIC 0.8
LIST 1 AXES 1.0 ’DisCen’ YZER
COUR 2 ’VitCen’ VITE COMP 3 POIN LECT cen TERM
TRAC 2 AXES 1.0 ’VITCen’ YZER
     COLO bleu
     THIC 0.8
LIST 2 AXES 1.0 ’VitCen’ YZER
RCOU 31 ’DisCen’ FICH ’osto03b.pun’ RENA ’DisCen03’
RCOU 41 ’DisCen’ FICH ’osto04b.pun’ RENA ’DisCen04’
RCOU 51 ’DisCen’ FICH ’osto05b.pun’ RENA ’DisCen05’
TRAC 31 41 51 1 AXES 1.0 ’DisCen’ YZER
COLO NOIR TURQ VERT ROUG
FIN
```

osto06c.epx

```
OSTO06C
ECHO
!CONV WIN
RESU SPLI ALIC 'osto06.ali' GARD PSCR
SORT VISU NSTO 1
PLAY
CAME   1 EYE  -3.18123E-01 -3.98270E-01  3.11803E-01
!        Q     7.92529E-01  5.04896E-01 -1.83767E-01 -2.88457E-01
         VIEW  5.82563E-01  6.94272E-01 -4.22618E-01
         RIGH  7.66044E-01 -6.42788E-01  0.00000E+00
         UP    2.71654E-01  3.23744E-01  9.06308E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.00092E-01  1.00140E-01  8.40999E-03
!RSPHERE:  1.43578E-01
!RADIUS :  7.17889E-01
!ASPECT :  1.00000E+00
!NEAR   :  5.59953E-01
!FAR    :  1.00504E+00
SCEN GEOM NAVI FREE
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 64 FPS 15 KFRE 10 COMP -1 NFAI
     OBJE LECT plate TERM REND
GOTR LOOP 62 OFFS FICH AVI CONT NOCL NFAI
     OBJE LECT plate TERM REND
GO
TRAC OFFS FICH AVI CONT NFAI
     OBJE LECT plate TERM REND
ENDPLAY
FIN
```

pepi01.epx

```
PEPI01
ECHO
 CONV WIN
LAGR CPLA
opti dump
GEOM LIBR POIN 8 Q42L 2 TERM
 0 -1 1 -1 0 0 1 0
 0 0 1 0 0 1 1 1
 1 2 4 3
 5 6 8 7
COMP EPAI 1.0 LECT 1 2 TERM
MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 2.D8
          TRAC 3 2.D8 2.D-3 3.D8 1. 3.1D8 2.
          LECT 1 2 TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY LECT 1 TERM
          BODY LECT 2 TERM
INIT VITE 1  50 LECT 5 6 7 8 TERM
     VITE 2 -50 LECT 5 6 7 8 TERM
ECRI COOR DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO FREQ 1
     FICH ALIC FREQ 1
OPTI PAS AUTO NOTE LOG 1
     LNKS STAT DIAG DUMP
CALC TINI 0. TEND 1.0 NMAX 5
FIN
```

pepi02.epx

```
PEPI02
ECHO
 CONV WIN
LAGR CPLA
opti dump
GEOM LIBR POIN 8 Q42L 2 TERM
 0 -1 1 -1 0 0 1 0
 0 0 1 0 0 1 1 1
 1 2 4 3
 5 6 8 7
COMP EPAI 1.0 LECT 1 2 TERM
MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 2.D8
          TRAC 3 2.D8 2.D-3 3.D8 1. 3.1D8 2.
          LECT 1 2 TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY FROT MUST 0.3 MUDY 0.1 GAMM 2.3025
          LECT 1 TERM
          BODY FROT MUST 0.3 MUDY 0.1 GAMM 2.3025
          LECT 2 TERM
INIT VITE 1  50 LECT 5 6 7 8 TERM
     VITE 2 -50 LECT 5 6 7 8 TERM
ECRI COOR DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO FREQ 1
     FICH ALIC FREQ 1
OPTI PAS AUTO NOTE LOG 1
     LNKS STAT DIAG DUMP
CALC TINI 0. TEND 1.0 NMAX 5
FIN
```

pepi03.epx

```
PEPI03
ECHO
 CONV WIN
LAGR CPLA
opti dump
GEOM LIBR POIN 8 Q42L 2 TERM
 0 -1 1 -1 0 0 1 0
 0 0 1 0 0 1 1 1
 1 2 4 3
 5 6 8 7
COMP EPAI 1.0 LECT 1 2 TERM
MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 2.D8
          TRAC 3 2.D8 2.D-3 3.D8 1. 3.1D8 2.
          LECT 1 2 TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY MLEV 1 LECT 1 TERM
          BODY MLEV 1 LECT 2 TERM
INIT VITE 1  50 LECT 5 6 7 8 TERM
     VITE 2 -50 LECT 5 6 7 8 TERM
ECRI COOR DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO FREQ 1
     FICH ALIC FREQ 1
OPTI PAS AUTO NOTE LOG 1
     LNKS STAT DIAG DUMP
CALC TINI 0. TEND 1.0 NMAX 5
FIN
```

pepi04.epx

```
PEPI04
ECHO
 CONV WIN
LAGR CPLA
opti dump
GEOM LIBR POIN 8 Q42L 2 TERM
 0 -1 1 -1 0 0 1 0
 0 0 1 0 0 1 1 1
 1 2 4 3
 5 6 8 7
COMP EPAI 1.0 LECT 1 2 TERM
MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 2.D8
          TRAC 3 2.D8 2.D-3 3.D8 1. 3.1D8 2.
          LECT 1 2 TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY FROT MUST 0.3 MUDY 0.1 GAMM 2.3025 MLEV 1
          LECT 1 TERM
          BODY FROT MUST 0.3 MUDY 0.1 GAMM 2.3025 MLEV 1
          LECT 2 TERM
INIT VITE 1  50 LECT 5 6 7 8 TERM
     VITE 2 -50 LECT 5 6 7 8 TERM
ECRI COOR DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO FREQ 1
     FICH ALIC FREQ 1
OPTI PAS AUTO NOTE LOG 1
     LNKS STAT DIAG DUMP
CALC TINI 0. TEND 1.0 NMAX 5
FIN
```

pepi05.epx

```
PEPI05
ECHO
 CONV WIN
LAGR CPLA
opti dump
GEOM LIBR POIN 8 Q42L 2 TERM
 0 -1 1 -1 0 0 1 0
 0 0 1 0 0 1 1 1
 1 2 4 3
 5 6 8 7
COMP EPAI 1.0 LECT 1 2 TERM
MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 2.D8
          TRAC 3 2.D8 2.D-3 3.D8 1. 3.1D8 2.
          LECT 1 2 TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY MLEV 2 LECT 1 TERM
          BODY MLEV 2 LECT 2 TERM
INIT VITE 1  50 LECT 5 6 7 8 TERM
     VITE 2 -50 LECT 5 6 7 8 TERM
ECRI COOR DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO FREQ 1
     FICH ALIC FREQ 1
OPTI PAS AUTO NOTE LOG 1
     LNKS STAT DIAG DUMP
CALC TINI 0. TEND 1.0 NMAX 5
FIN
```

pepi06.epx

```
PEPI06
ECHO
 CONV WIN
LAGR CPLA
opti dump
GEOM LIBR POIN 8 Q42L 2 TERM
 0 -1 1 -1 0 0 1 0
 0 0 1 0 0 1 1 1
 1 2 4 3
 5 6 8 7
COMP EPAI 1.0 LECT 1 2 TERM
```

```
MATE VM23 RO 8000. YOUN 1.D11 NU 0.3 ELAS 2.D8
          TRAC 3 2.D8 2.D-3 3.D8 1. 3.1D8 2.
          LECT 1 2 TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY FROT MUST 0.3 MUDY 0.1 GAMM 2.3025 MLEV 2
          LECT 1 TERM
          BODY FROT MUST 0.3 MUDY 0.1 GAMM 2.3025 MLEV 2
          LECT 2 TERM
INIT VITE 1  50 LECT 5 6 7 8 TERM
     VITE 2 -50 LECT 5 6 7 8 TERM
ECRI COOR DEPL VITE ACCE FINT FEXT FLIA FDEC CONT ECRO FREQ 1
     FICH ALIC FREQ 1
OPTI PAS AUTO NOTE LOG 1
     LNKS STAT DIAG DUMP
CALC TINI 0. TEND 1.0 NMAX 5
FIN
```

px4cir2d.proc

```
*$$$$ PX4CIR2D
*
* Pour generer le maillage 2D d'un quart de cercle
* avec seulement des quadrilateres a 4 noeuds.
* Le quart de cercle est defini par les deux extremes
* d'un arc (de 90 degrees) et par le centre du cercle.
*
* Input:
* =====
* P1 = premiere extremite de l'arc
* P2 = deuxieme extremite de l'arc
* PC = centre de l'arc
* N  = nombre de mailles a generer sur chaque cote (doit etre pair)
* TOL= tolerance pour l'elimination des noeuds doubles
*
* Output:
* ======
* SUR = objet MAILLAGE d'elements de type QUA4
* IER = 0: pas d'erreur, .NE.0: erreur dans la generation de SUR
*
'DEBPROC' PX4CIR2D P1*'POINT' P2*'POINT' PC*'POINT'
                   N*'ENTIER' TOL*'FLOTTANT';
*-------------------------------------------------------------------
*
ier=0;
n2 = n / 2;
p0 = 0 0;
pm1 = p1 plus p0;
depl pm1 tour 45 pc;
pm2 = 0.5*(pc plus p2);
pm3 = 0.5*(pc plus p1);
pm = 0.5*(pc plus pm1);
c1a = cerc n2 p1 pc pm1;
c1b = cerc n2 pm1 pc p2;
c2a = droi n2 p2 pm2;
c2b = droi n2 pm2 pc;
c3a = droi n2 pc pm3;
c3b = droi n2 pm3 p1;
c4a = droi n2 pm pm1;
c4b = droi n2 pm pm2;
c4c = droi n2 pm pm3;
sur1 = dall plan c4c c3b c1a (inve c4a);
sur2 = dall plan c4a c1b c2a (inve c4b);
sur3 = dall plan c2b c3a (inve c4c) c4b;
sur = sur1 et sur2 et sur3;
*
elim tol sur;
*
'FINPROC' sur ier;
```

px4cir3d.proc

```
*$$$$ PX4CIR3D
*
* Pour generer le maillage 3D (plan) d'un quart de cercle
* avec seulement des quadrilateres a 4 noeuds.
* Le quart de cercle est defini par les deux extremes
* d'un arc (de 90 degrees), par le centre du cercle
* et par un autre point qui definit l'axe de rotation
* (axe perpendiculaire au plan du cercle, passant pour son centre).
*
* Input:
* =====
* P1 = premiere extremite de l'arc
* P2 = deuxieme extremite de l'arc
* PC = centre de l'arc
* PZ = autre point de l'axe
* N  = nombre de mailles a generer sur chaque cote (doit etre pair)
* TOL= tolerance pour l'elimination des noeuds doubles
*
* Output:
* ======
* SUR = objet MAILLAGE d'elements de type QUA4
* IER = 0: pas d'erreur, .NE.0: erreur dans la generation de SUR
*
'DEBPROC' PX4CIR3D P1*'POINT' P2*'POINT' PC*'POINT' PZ*'POINT'
                   N*'ENTIER' TOL*'FLOTTANT';
*-------------------------------------------------------------------
*
ier=0;
n2 = n / 2;
p0 = 0 0 0;
pm1 = p1 plus p0;
depl pm1 tour 45 pc pz;
pm2 = 0.5*(pc plus p2);
pm3 = 0.5*(pc plus p1);
pm = 0.5*(pc plus pm1);
c1a = cerc n2 p1 pc pm1;
c1b = cerc n2 pm1 pc p2;
c2a = droi n2 p2 pm2;
c2b = droi n2 pm2 pc;
c3a = droi n2 pc pm3;
c3b = droi n2 pm3 p1;
c4a = droi n2 pm pm1;
c4b = droi n2 pm pm2;
c4c = droi n2 pm pm3;
sur1 = dall plan c4c c3b c1a (inve c4a);
sur2 = dall plan c4a c1b c2a (inve c4b);
sur3 = dall plan c2b c3a (inve c4c) c4b;
sur = sur1 et sur2 et sur3;
*
elim tol sur;
*
'FINPROC' sur ier;
```

pxbox2d.proc

```
'DEBPROC' pxbox2d x0*'FLOTTANT' y0*'FLOTTANT'
                  lx*'FLOTTANT' ly*'FLOTTANT'
                  dd*'FLOTTANT';
*
*--------------------------------------------------
* Generates a rectangular mesh with origin in point
* (x0,y0), sides of length (lx,ly) and density (mesh size) dd.
* The mesh consists of QUA4 quadrangular elements and is oriented
* along the global axes.
*
* Input :
* -----
*     x0,y0 : coordinates of 'origin' of the box
*     lx,ly : length of the box sides
*     dd : "density" (size) of the mesh (the same in all directions)
* Output :
* ------
*     box : mesh consisting of CUB8 hexahedra
*--------------------------------------------------
*
dens dd;
p1 = x0 y0;
p2 = (x0 + lx) y0;
*
c1 = p1 d p2;
box = c1 tran (0 ly);
*
finproc box;
```

pxbox3d.proc

```
'DEBPROC' pxbox3d x0*'FLOTTANT' y0*'FLOTTANT' z0*'FLOTTANT'
                  lx*'FLOTTANT' ly*'FLOTTANT' lz*'FLOTTANT'
                  dd*'FLOTTANT';
*
*--------------------------------------------------
* Generates a parallelepiped mesh with origin in point
* (x0,y0,z0), sides of length (lx,ly,lz) and density (mesh size) dd.
* The mesh consists of CUB8 hexahedral elements and is oriented
* along the global axes.
*
* Input :
* -----
*     x0,y0,z0 : coordinates of 'origin' of the box
*     lx,ly,lz : length of the box sides
*     dd : "density" (size) of the mesh (the same in all directions)
* Output :
* ------
*     box : mesh consisting of CUB8 hexahedra
*--------------------------------------------------
*
dens dd;
p1 = x0 y0 z0;
p2 = (x0 + lx) y0 z0;
p3 = (x0 + lx) (y0 + ly) z0;
p4 = x0 (y0 + ly) z0;
*
c1 = p1 d p2;
c2 = p2 d p3;
c3 = p3 d p4;
c4 = p4 d p1;
base = dall c1 c2 c3 c4 plan;
*
box = base volu tran (0 0 lz);
*
finproc box;
```

pxextb2d.proc

```
’DEBPROC’ pxextb2d m*’MAILLAGE’ x1*’FLOTTANT’ x2*’FLOTTANT’
                   y1*’FLOTTANT’ y2*’FLOTTANT’;
*
*--------------------------------------------------
* Extracts from the 2D mesh m the elements whose centroid is
* located in the box [x1-x2,y1-y2].
*
* Input :
* -----
*     m              : 2D mesh
*     x1, x2, y1, y2 : extremes of the box
* Output :
* ------
*     box : mesh contained in the box
*--------------------------------------------------
*
nel = nbel m;
n = 0;
i = 0;
repe loop1 nel;
  i = i + 1;
  iel = m elem i;
  x y = coor (bary (iel));
  si (>eg x x1);
    si (<eg x x2);
      si (>eg y y1);
        si (<eg y y2);
          n = n + 1;
          si (ega n 1);
            box = iel;
          sinon;
            box = box et iel;
          finsi;
        finsi;
      finsi;
    finsi;
  finsi;
fin loop1;
*
finproc box;
```

pxordpoi.proc

```
*$$$$ PXORDPOI
*
* pour ordonner une serie de points PLIN en partant de P1
*
* Input:
* =====
* PLIN = objet MAILLAGE de type POI1 (ligne de points)
* P1   = premier point de la ligne (typ POINT)
*
* Output:
* ======
* PORDO = objet MAILLAGE de type POI1 (ligne de points) contenant
*         les points ordonnes a partir de P1
*
’DEBPROC’ PXORDPOI PLIN*’MAILLAGE’ P1*’POINT’ ;
*-----------------------------------------------------------------
*
PORDO=P1;
PPA=P1;
NE=’NBEL’ PLIN;
*
I=0;
’REPETER’ LAB1 (NE-1);
  I=I + 1;
* mess I;
  PLIN= ’DIFF’ ((PPA ’ET’ PPA) ’ELEM’ 1) PLIN;
  PPA=PLIN ’POIN’ ’PROC’ PPA;
  PORDO=PORDO ’ET’ PPA;
’FIN’ LAB1;
*
’FINPROC’ PORDO;
```

pxq42q92.proc

```
’DEBPROC’ pxq42q92 q4*’MAILLAGE’;
*
*--------------------------------------------------
* Transforms a q4 (in 2d !!!) into a q9 by creating:
* - 4 new mid-side nodes
* - 1 new central  node
*
* Input :
* -----
*     q4 : a mesh containing just one q4 (in 2d)
* Output :
* ------
*     q9 : mesh containing one q9 (in 3d: meshed as qua9)
*--------------------------------------------------
*
hh = chan poi1 q4;
*
p01 = hh poin 1;
p02 = hh poin 2;
p03 = hh poin 3;
p04 = hh poin 4;
*
n01 = noeu p01;
n02 = noeu p02;
n03 = noeu p03;
n04 = noeu p04;
*
x01 y01 = coor p01;
x02 y02 = coor p02;
x03 y03 = coor p03;
x04 y04 = coor p04;
*
x05 = (x01 + x02) / 2.0;
y05 = (y01 + y02) / 2.0;
p05 = x05 y05;
n05 = noeu p05;
*
x06 = (x02 + x03) / 2.0;
y06 = (y02 + y03) / 2.0;
p06 = x06 y06;
n06 = noeu p06;
*
x07 = (x03 + x04) / 2.0;
y07 = (y03 + y04) / 2.0;
p07 = x07 y07;
n07 = noeu p07;
*
x08 = (x01 + x04) / 2.0;
y08 = (y01 + y04) / 2.0;
p08 = x08 y08;
n08 = noeu p08;
*
x09 = (x01 + x02 + x03 + x04) / 4.0;
y09 = (y01 + y02 + y03 + y04) / 4.0;
p09 = x09 y09;
n09 = noeu p09;
*
* Q9 element in 2D (meshed as a "qua9")
*
*q9 = manu qua9 p01 p02 p03 p04 p05 p06 p07 p08 p09;
q9 = manu qua9 p01 p05 p02 p06 p03 p07 p04 p08 p09;
*
finproc q9;
```

pxsph8.proc

```
’DEBPROC’ PXSPH8 cent*’POINT’ rcub*’FLOTTANT’ rsph*’FLOTTANT’
                 ncub*’ENTIER’;
*
*--------------------------------------------------
* Generates a mesh of hexahedra consisting of 1/8 of a cube
* surrounded by 1/8 of a sphere.
* The origin of the cube is at point cent and the radius of the cube
* is rcub. The radius of the sphere is rsph (> rcub).
* The number of subdivisions along the cube radius is ncub.
* The cube is oriented along the global axes X, Y, Z and
* the resulting mesh (cube end sphere portion) lies in the first
* octant.
*
* Input :
* -----
*     cent : center of the cube/sphere
*     rcub : radius (half-side) of the cube
*     rsph : radius of the sphere
*     ncub : number of subdivisions along the cube radius
*
* Output :
* ------
*     hexa : mesh of hexahedra in 3d
*--------------------------------------------------
*
* Longueurs caracteristiques du maillage
*
rsph_c = ((2**0.5)/2)*rsph;
rsph_s = ((3**0.5)/3)*rsph;
P = 0.5*rsph*3.14159;
*
* Points
*
dc = rcub / ncub;
OPTI DENS dc;
*
tol = 0.01*dc;
*
O = 0. 0. 0.;
Cx = rcub 0. 0.;
Cy = 0. rcub 0.;
Cz = 0. 0. rcub;
Cdb = rcub rcub 0.;
Cdh = rcub rcub rcub;
Cxz = rcub 0. rcub;
Cyz = 0. rcub rcub;
*
* Lignes cube
*
OCx = O DROIT Cx;
OCy = O DROIT Cy;
OCz = O DROIT Cz;
CxCxz = Cxz DROIT Cx;
CyCyz = Cy DROIT Cyz;
CdbCdh = Cdb DROIT Cdh;
```

```
CxCdb = Cx DROIT Cdb;
CyCdb = Cy DROIT Cdb;
CxzCdh = Cxz DROIT Cdh;
CyzCdh = Cyz DROIT Cdh;
CzCxz = Cz DROIT Cxz;
CzCyz = Cz DROIT Cyz;
*
* Surfaces cube
*
carre_y1 = OCx CxCxz CzCxz OCz DALLER PLAN;
carre_x1 = OCy CyCyz CzCyz OCz DALLER PLAN;
carre_z1 = OCx CxCdb CyCdb OCy DALLER PLAN;
carre_y2 = CyCdb CdbCdh CyzCdh CyCyz DALLER PLAN;
carre_x2 = CxCdb CdbCdh CxzCdh CxCxz DALLER PLAN;
carre_z2 = CzCxz CxzCdh CyzCdh CzCyz DALLER PLAN;
*
* Cube
*
cube = carre_y1 carre_x1 carre_z1 carre_y2 carre_x2 carre_z2 pave;
*
* Points sphere
*
ds = P / (2*ncub);
OPTI DENS ds;
*
ps1 = rsph 0 0;
ps2 = rsph_c rsph_c 0;
ps3 = rsph_s rsph_s rsph_s;
ps4 = rsph_c 0 rsph_c;
ps5 = 0 rsph 0;
ps6 = 0 rsph_c rsph_c;
ps7 = 0 0 rsph;
*
* Lignes sphere
*
ls1 = CERC ncub ps1 O ps2;
ls2 = CERC ncub ps2 O ps3;
ls3 = CERC ncub ps3 O ps4;
ls4 = CERC ncub ps4 O ps1;
*
ls5 = CERC ncub ps5 O ps2;
ls6 = CERC ncub ps3 O ps6;
ls7 = CERC ncub ps6 O ps5;
ls8 = CERC ncub ps7 O ps4;
ls9 = CERC ncub ps6 O ps7;
*
* Surfaces sphere
*
ss1 = DALL ls1 ls2 ls3 ls4 'SPHE' O;
ss2 = DALL ls5 ls2 ls6 ls7 'SPHE' O;
ss3 = DALL ls8 ls3 ls6 ls9 'SPHE' O;
*
* Volumes sphere
*
vs1 = carre_x2 VOLU ss1;
vs2 = carre_y2 VOLU ss2;
vs3 = carre_z2 VOLU ss3;
*
hexa = cube et vs1 et vs2 et vs3;
elim tol hexa;
DEPL hexa 'PLUS' cent;
*
'FINPROC' hexa;
```

pxsphere.proc

```
'DEBPROC' PXSPHERE cent*'POINT' rcub*'FLOTTANT' rsph*'FLOTTANT'
                  ncub*'ENTIER';
*
*------------------------------------------------
* Generates a mesh of hexahedra consisting of a cube
* surrounded by a sphere.
* The origin of the cube is at point cent and the radius of the cube
* is rcub. The radius of the sphere is rsph (> rcub).
* The number of subdivisions along the cube radius is ncub.
*
* Input :
* -----
*     cent : center of the cube/sphere
*     rcub : radius (half-side) of the cube
*     rsph : radius of the sphere
*     ncub : number of subdivisions along the cube radius
*
* Output :
* ------
*     hexa : mesh of hexahedra in 3d
*
* Attention: this procedure uses procedure PXSPH8, which must
*           be read in first in the calling Gibiane data set
*---------------------------------------------------------
*cent = 0 0 0;
*rcub = 1.0;
*rsph = 3.0;
*ncub = 4;
px = 1 0 0;
py = 0 1 0;
pz = 0 0 1;
tol = 0.01 * rcub;
tol = tol / ncub;
sp1 = pxsph8 cent rcub rsph ncub;
sp2 = sp1 syme plan cent py pz;
sp12 = sp1 et sp2;
sp3 = sp12 syme plan cent px pz;
sp123 = sp12 et sp3;
sp4 = sp123 syme plan cent px py;
sp = sp123 et sp4;
elim tol sp;
'FINPROC' sp;
```

stpm01.dgibi

```
*$$$$ PX4CIR3D
*
* Pour generer le maillage 3D (plan) d'un quart de cercle
* avec seulement des quadrilateres a 4 noeuds.
* Le quart de cercle est defini par les deux extremes
* d'un arc (de 90 degrees), par le centre du cercle
* et par un autre point qui definit l'axe de rotation
* (axe perpendiculaire au plan du cercle, passant pour son centre).
*
* Input:
* =====
* P1 = premiere extremite de l'arc
* P2 = deuxieme extremite de l'arc
* PC = centre de l'arc
* PZ = autre point de l'axe
* N  = nombre de mailles a generer sur chaque cote (doit etre pair)
* TOL= tolerance pour l'elimination des noeuds doubles
*
* Output:
* ======
* SUR = objet MAILLAGE d'elements de type QUA4
* IER = 0: pas d'erreur, .NE.0: erreur dans la generation de SUR
*
'DEBPROC' PX4CIR3D P1*'POINT' P2*'POINT' PC*'POINT' PZ*'POINT'
                  N*'ENTIER' TOL*'FLOTTANT';
*---------------------------------------------------------------
*
ier=0;
n2 = n / 2;
p0 = 0 0 0;
pm1 = p1 plus p0;
depl pm1 tour 45 pc pz;
pm2 = 0.5*(pc plus p2);
pm3 = 0.5*(pc plus p1);
pm = 0.5*(pc plus pm1);
c1a = cerc n2 p1 pc pm1;
c1b = cerc n2 pm1 pc p2;
c2a = droi n2 p2 pm2;
c2b = droi n2 pm2 pc;
c3a = droi n2 pc pm3;
c3b = droi n2 pm3 p1;
c4a = droi n2 pm pm1;
c4b = droi n2 pm pm2;
c4c = droi n2 pm pm3;
sur1 = dall plan c4c c3b c1a (inve c4a);
sur2 = dall plan c4a c1b c2a (inve c4b);
sur3 = dall plan c2b c3a (inve c4c) c4b;
sur = sur1 et sur2 et sur3;
*
elim tol sur;
*
'FINPROC' sur ier;
*********************************************************************
opti echo 1;
opti dime 3 elem cub8;
opti sauv form 'stpm01.msh';
opti trac psc ftra 'stpm01_mesh.ps';
*
* upper frame
*
tol = 1.E-5;
p0 = 0 0 0;
pz = 0 0 1;
p1 = 0.150 0 0;
dhole = 0.025;
rhole = dhole / 2.0;
dhext = 0.042;
rhext = dhext / 2.0;
p2 = (0.260 - rhext) 0 0;
p3 = (0.260 + rhext) 0 0;
p4 = 0.325 0 0;
p5 = p4 tour 45.0 p0 pz;
p6 = p3 tour 45.0 p0 pz;
p7 = p2 tour 45.0 p0 pz;
p8 = 0.150 0.150 0;
*
c1 = p1 d 9 p2;
pc1 = 0.260 0 0;
pz1 = pc1 plus pz;
p2b = p3 tour 90.0 pc1 pz1;
c2 = p2 c 2 pc1 p2b c 2 pc1 p3;
c3 = p3 d 4 p4;
c4 = p4 c 13 p0 p5;
c5 = p5 d 3 p6;
pc2 = pc1 tour 45.0 p0 pz;
pz2 = pc2 plus pz;
p6b = p7 tour 90.0 pc2 pz2;
c6 = p6 c 2 pc2 p6b c 2 pc2 p7;
c7 = p7 d 2 p8;
c8 = p8 d 15 p1;
*
cc0 = c1 et c2 et c3 et c4 et c5 et c6 et c7 et c8;
```

```
elim tol cc0;
*
c9a = c2 tour 15.0 p0 pz;
c9b = c6 tour (0 - 30.0) p0 pz;
c9 = elim tol (c9a et c9b);
c10 = c9 tour 15.0 p0 pz;
cc = cc0 et c9 et c10;
*
bas1 = surf cc plan;
bas1 = orie bas1 dire pz;
*
c11 = ((0.260 + rhole) 0 0) d 1 p3;
pre1 = c11 rota 4 180.0 pc1 pz1;
pre2a = pre1 tour 15.0 p0 pz;
pc3 = pc1 tour 15.0 p0 pz;
pre2b = pre2a tour 180.0 pc3 (pc3 plus pz);
pre2 = pre2a et pre2b;
pre3 = pre2 tour 15.0 p0 pz;
pre4 = pre2b tour 30.0 p0 pz;
baspr = pre1 et pre2 et pre3 et pre4;
baspr = orie baspr dire pz;
elim tol (bas1 et baspr);
*
bas2 = bas1 syme plan p8 p5 (p8 plus pz);
baspr2 = baspr syme plan p8 p5 (p8 plus pz);
bas = bas1 et baspr et bas2 et baspr2;
elim tol bas;
bas = orie bas dire pz;
trac bas;
*
gap = 0.0008;
gap2 = gap / 2.0;
ubas = bas plus (0 0 gap2);
thu = 0.025;
uframe = ubas volu tran 3 (0 0 thu);
trac cach uframe;
*
* plate
*
c12 = p0 d 15 p1;
placen = c12 tran 15 (0 0.15 0);
*
p4p = 0.3125 0 0;
c3p = p3 d 4 p4p;
p5p = 0.3125 0.3125 0;
c4p = p4p d 16 p5p;
c5p = p5p d 9 p6;
cc0p = c1 et c2 et c3p et c4p et c5p et c6 et c7 et c8;
*
ccp = cc0p et c9 et c10;
elim tol ccp;
*
bas1p = surf ccp plan;
bas1p = orie bas1p dire pz;
bas2p = bas1p syme plan p8 p5 (p8 plus pz);
plate = bas1p et baspr et bas2p et baspr2 et placen;
elim tol plate;
plate = orie plate dire pz;
trac plate;
trac cach (uframe et plate);
nplate = chan poi1 plate;
*
* lower frame with bolts
*
dbolt = 0.021;
rbolt = dbolt / 2.0;
*
lpb = (0.260 + rbolt) 0 0;
c11 = lpb d 1 p3;
lpre1 = c11 rota 4 180.0 pc1 pz1;
lpre2a = lpre1 tour 15.0 p0 pz;
lpre2b = lpre2a tour 180.0 pc3 (pc3 plus pz);
lpre2 = lpre2a et lpre2b;
lpre3 = lpre2 tour 15.0 p0 pz;
lpre4 = lpre2b tour 30.0 p0 pz;
lbaspr = lpre1 et lpre2 et lpre3 et lpre4;
lbaspr = orie lbaspr dire pz;
elim tol (bas1 et lbaspr);
*
lbaspr2 = lbaspr syme plan p8 p5 (p8 plus pz);
lbas = bas1 et lbaspr et bas2 et lbaspr2;
elim tol lbas;
lbas = orie lbas dire pz;
trac lbas;
*
thl = 0.025;
lobas = lbas plus (0 0 (0 - gap2 - thl));
lframe = lobas volu tran 3 (0 0 thl);
trac cach lframe;
trac cach (uframe et lframe);
trac cach (uframe et plate et lframe);
*
lpb2 = lpb tour 90.0 pc1 pz1;
sur11 ier = PX4CIR3D lpb lpb2 pc1 pz1 2 1.E-3;
sur12 = sur11 tour 90.0 pc1 pz1;
sur1 = sur11 et sur12;
elim tol sur1;
sur1 = orie sur1 dire pz;
*
sur2a = sur1 tour 30.0 p0 pz;
sur1s = sur1 syme plan pc1 lpb pz1;
sur1s = orie sur1s dire pz;
sur2b = sur1s tour 30.0 p0 pz;
sur2 = sur2a et sur2b;
elim tol sur2;
sur2 = orie sur2 dire pz;
sur3 = sur2 tour 30.0 p0 pz;
sur4 = sur1s tour 90.0 p0 pz;
sur1 = sur1 plus (0 0 (0 - gap2 - thl));
sur2 = sur2 plus (0 0 (0 - gap2 - thl));
sur3 = sur3 plus (0 0 (0 - gap2 - thl));
sur4 = sur4 plus (0 0 (0 - gap2 - thl));
*
hbolt = 0.06;
hboltu = hbolt - thl;
*
bolt1a = sur1 volu tran 3 (0 0 thl);
bolt1b = (sur1 plus (0 0 thl)) volu tran 4 (0 0 hboltu);
bolt1 = bolt1a et bolt1b;
elim tol (bolt1 et lframe);
bolt2a = sur2 volu tran 3 (0 0 thl);
bolt2b = (sur2 plus (0 0 thl)) volu tran 4 (0 0 hboltu);
bolt2 = bolt2a et bolt2b;
elim tol (bolt2 et lframe);
bolt3a = sur3 volu tran 3 (0 0 thl);
bolt3b = (sur3 plus (0 0 thl)) volu tran 4 (0 0 hboltu);
bolt3 = bolt3a et bolt3b;
elim tol (bolt3 et lframe);
bolt4a = sur4 volu tran 3 (0 0 thl);
bolt4b = (sur4 plus (0 0 thl)) volu tran 4 (0 0 hboltu);
bolt4 = bolt4a et bolt4b;
elim tol (bolt4 et lframe);
bolts = bolt1 et bolt2 et bolt3 et bolt4;
lframeb = lframe et bolts;
elim tol lframeb;
trac cach lframeb;
trac cach (lframeb et plate);
trac cach (lframeb et plate et uframe);
*
* pressurized clamping surfaces
*
presa = pre1 plus (0 0 (gap2 + thu));
presa = presa orie dire pz;
presb = pre3 plus (0 0 (gap2 + thu));
presb = presb orie dire pz;
presc = presb tour 30.0 p0 pz;
presd = (presa tour 90.0 p0 pz) syme plan p0 (0 1 0) pz;
presd = presd orie dire pz;
presur = presa et presb et presc et presd;
elim tol (presur et uframe);
trac cach (presur et uframe);
*
* pressurized (exposed) part of the plate
*
preplat = placen coul roug;
trac cach (plate et preplat);
*
ecub8 = (lframeb et uframe) elem cub8;
epri6 = (lframeb et uframe) elem pri6;
equa4 = plate elem qua4;
etri3 = plate elem tri3;
*
mesh = lframeb et plate et uframe et presur et preplat
      et nplate et ecub8 et epri6 et equa4 et etri3;
tass mesh noop;
sauv form mesh;
trac cach mesh;
*
fin;
```

stpm01.epx

```
STPM01
ECHO
!CONV win
CAST MESH
EROS 1.0
TRID LAGR
GEOM CUB8 ecub8
     PR6  epri6
     Q4GS equa4
     T3GS etri3
     PMAT nplate
     CL3D presur preplat
TERM
COMP EPAI 0.8e-3 LECT plate nplate TERM
     COUL TURQ LECT lframeb TERM
          BLEU LECT uframe TERM
          VERT LECT plate TERM
          ROSE LECT presur TERM
          ROUG LECT preplat nplate TERM
     NGRO 3 'bloz'  LECT lframeb TERM COND Z LT -0.0253
            'symx'  LECT mesh TERM    COND X LT 0.0001
            'symy'  LECT mesh TERM    COND Y LT 0.0001
     GROU 1 'prpl'  LECT preplat TERM COND NEAR POIN 0.0 0.0 0.0
!    NGRO 3 'centd' LECT PART 2 TERM COND NEAR POIN 0.0 0.0 0.0
!           'centl' LECT PART 3 TERM COND NEAR POIN 0.0 0.0 0.0
!           'axis'  LECT PART 2 TERM
!                   COND LINE X1 -0.2 Y1 -0.001 Z1 -0.001
!                             X2  0.2 Y2  0.001 Z2  0.001 TOL 1.E-5
     ORIE INVE LECT preplat TERM
INCLUDE 'p77_75.txt'
MATE VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 3.257E8 mxit 500
```

```
          QR1 2.348E8 CR1 56.2 QR2 4.457E8 CR2 4.7
          PDOT 5.E-4 C 1.E-2 TQ 0.9 CP 452.0
          TM 1800.0 M 1.0 DC 1.0 WC 555.0E6
          LECT lframeb plate uframe TERM
     MASS 0.0 YOUN 2.1E11 NU 0.33
          LECT nplate TERM
     IMPE PIMP RO 7850.0 PRES 43.1e6 PREF 1.0E5
          LECT presur TERM
     IMPE PIMP RO 7850 PRES -1.0 PREF 1.0E5 FONC 1
          LECT preplat TERM
OPTI PINS ASN
LINK COUP SPLT NONE
     BLOQ 3 LECT bloz TERM
     CONT SPLA NX 1 NY 0 NZ 0 LECT symx TERM
     CONT SPLA NX 0 NY 1 NZ 0 LECT symy TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY FROT MUST 0.15 MUDY 0.1 GAMM 0 MLEV 5
          LECT lframeb TERM
          BODY FROT MUST 0.15 MUDY 0.1 GAMM 0 MLEV 5
          LECT uframe TERM
          BODY FROT MUST 0.15 MUDY 0.1 GAMM 0 DIAM 0.0008
          LECT nplate TERM
ECRI DEPL VITE ECRO FAIL TFRE 1.E-3
          POIN LECT 1 TERM
          ELEM LECT 1 TERM
     FICH SPLI ALIC TFRE 0.1E-3
     FICH ALIC TEMP TFRE 0.01E-3
          POIN LECT p0 TERM
          ELEM LECT prpl TERM
OPTI NOTE CSTA 0.8
     LOG 1
     JAUM
     LMST
     PINS GRID DPIN 1.01
    !lnks stat
CALC TINI 0 TEND 10.0E-3
FIN
```

stpm01a.epx

```
STPM01A
ECHO
 CONV WIN
RESU SPLI ALIC 'stpm01.ali' GARD PSCR
SORT VISU NSTO 1
PLAY
CAME   1 EYE  -4.12376E-01 -4.12376E-01  6.82373E-01
!        Q     8.37319E-01  3.90448E-01 -1.61729E-01 -3.46829E-01
         VIEW  5.41675E-01  5.41675E-01 -6.42788E-01
         RIGH  7.07107E-01 -7.07107E-01  5.55112E-17
         UP    4.54519E-01  4.54519E-01  7.66044E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.62658E-01  1.62658E-01  0.00000E+00
!RSPHERE:  2.41269E-01
!RADIUS :  1.06158E+00
!ASPECT :  1.00000E+00
!NEAR   :  8.20315E-01
!FAR    :  1.54412E+00
SCEN GEOM NAVI FREE
          FACE SBAC
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 101 FPS 15 KFRE 10 COMP -1 NFAI
!    OBJE LECT lframeb plate uframe TERM REND
     OBJE LECT lframeb plate TERM REND
GOTR LOOP 99 OFFS FICH AVI CONT NOCL NFAI
!    OBJE LECT lframeb plate uframe TERM REND
     OBJE LECT lframeb plate TERM REND
GO
TRAC OFFS FICH AVI CONT NFAI
!    OBJE LECT lframeb plate uframe TERM REND
     OBJE LECT lframeb plate TERM REND
ENDPLAY
FIN
```

stpm01b.epx

```
STPM01B
ECHO
RESU ALIC TEMP 'stpm01.alt' GARD PSCR
SORT GRAP
AXTE 1.0 'Time [s]'
COUR 1 'DisCen' DEPL COMP 3 POIN LECT p0 TERM
TRAC 1 AXES 1.0 'DisCen' YZER
     COLO bleu
     THIC 0.8
LIST 1 AXES 1.0 'DisCen' YZER
RCOU 11 'DisCen' FICH 'clam01b.pun' RENA 'DisCenC01'
RCOU 21 'DisCen' FICH 'clam02b.pun' RENA 'DisCenC02'
TRAC 11 21 1 AXES 1.0 'DisCen' YZER
COLO VERT ROUG BLEU
COUR 31 'DisCenInv' MULC 1 -1.0
TRAC 11 21 31 AXES 1.0 'DisCen' YZER
COLO VERT ROUG BLEU
FIN
```

stpm02.dgibi

```
*$$$$ PX4CIR3D
*
* Pour generer le maillage 3D (plan) d'un quart de cercle
* avec seulement des quadrilateres a 4 noeuds.
* Le quart de cercle est defini par les deux extremes
* d'un arc (de 90 degrees), par le centre du cercle
* et par un autre point qui definit l'axe de rotation
* (axe perpendiculaire au plan du cercle, passant pour son centre).
*
* Input:
* =====
* P1 = premiere extremite de l'arc
* P2 = deuxieme extremite de l'arc
* PC = centre de l'arc
* PZ = autre point de l'axe
* N  = nombre de mailles a generer sur chaque cote (doit etre pair)
* TOL= tolerance pour l'elimination des noeuds doubles
*
* Output:
* ======
* SUR = objet MAILLAGE d'elements de type QUA4
* IER = 0: pas d'erreur, .NE.0: erreur dans la generation de SUR
*
'DEBPROC' PX4CIR3D P1*'POINT' P2*'POINT' PC*'POINT' PZ*'POINT'
                   N*'ENTIER' TOL*'FLOTTANT';
*------------------------------------------------------------------
*
ier=0;
n2 = n / 2;
p0 = 0 0 0;
pm1 = p1 plus p0;
depl pm1 tour 45 pc pz;
pm2 = 0.5*(pc plus p2);
pm3 = 0.5*(pc plus p1);
pm = 0.5*(pc plus pm1);
c1a = cerc n2 p1 pc pm1;
c1b = cerc n2 pm1 pc p2;
c2a = droi n2 p2 pm2;
c2b = droi n2 pm2 pc;
c3a = droi n2 pc pm3;
c3b = droi n2 pm3 p1;
c4a = droi n2 pm pm1;
c4b = droi n2 pm pm2;
c4c = droi n2 pm pm3;
sur1 = dall plan c4c c3b c1a (inve c4a);
sur2 = dall plan c4a c1b c2a (inve c4b);
sur3 = dall plan c2b c3a (inve c4c) c4b;
sur = sur1 et sur2 et sur3;
*
elim tol sur;
*
'FINPROC' sur ier;
************************************************************************
opti echo 1;
opti dime 3 elem cub8;
opti sauv form 'stpm02.msh';
opti trac psc ftra 'stpm02_mesh.ps';
*
* upper frame
*
tol = 1.E-5;
p0 = 0 0 0;
pz = 0 0 1;
p1 = 0.150 0 0;
dhole = 0.025;
rhole = dhole / 2.0;
dhext = 0.042;
rhext = dhext / 2.0;
p2 = (0.260 - rhext) 0 0;
p3 = (0.260 + rhext) 0 0;
p4 = 0.325 0 0;
p5 = p4 tour 45.0 p0 pz;
p6 = p3 tour 45.0 p0 pz;
p7 = p2 tour 45.0 p0 pz;
p8 = 0.150 0.150 0;
*
c1 = p1 d 9 p2;
pc1 = 0.260 0 0;
pz1 = pc1 plus pz;
p2b = p3 tour 90.0 pc1 pz1;
c2 = p2 c 2 pc1 p2b c 2 pc1 p3;
c3 = p3 d 4 p4;
c4 = p4 c 13 p0 p5;
c5 = p5 d 3 p6;
pc2 = pc1 tour 45.0 p0 pz;
pz2 = pc2 plus pz;
p6b = p7 tour 90.0 pc2 pz2;
c6 = p6 c 2 pc2 p6b c 2 pc2 p7;
c7 = p7 d 2 p8;
c8 = p8 d 30 p1;
*
cc0 = c1 et c2 et c3 et c4 et c5 et c6 et c7 et c8;
elim tol cc0;
*
c9a = c2 tour 15.0 p0 pz;
c9b = c6 tour (0 - 30.0) p0 pz;
c9 = elim tol (c9a et c9b);
c10 = c9 tour 15.0 p0 pz;
cc = cc0 et c9 et c10;
*
```

```
bas1 = surf cc plan;
bas1 = orie bas1 dire pz;
*
c11 = ((0.260 + rhole) 0 0) d 1 p3;
pre1 = c11 rota 4 180.0 pc1 pz1;
pre2a = pre1 tour 15.0 p0 pz;
pc3 = pc1 tour 15.0 p0 pz;
pre2b = pre2a tour 180.0 pc3 (pc3 plus pz);
pre2 = pre2a et pre2b;
pre3 = pre2 tour 15.0 p0 pz;
pre4 = pre2b tour 30.0 p0 pz;
baspr = pre1 et pre2 et pre3 et pre4;
baspr = orie baspr dire pz;
elim tol (bas1 et baspr);
*
bas2 = bas1 syme plan p8 p5 (p8 plus pz);
baspr2 = baspr syme plan p8 p5 (p8 plus pz);
bas = bas1 et baspr et bas2 et baspr2;
elim tol bas;
bas = orie bas dire pz;
trac bas;
*
gap = 0.0008;
gap2 = gap / 2.0;
ubas = bas plus (0 0 gap2);
thu = 0.025;
uframe = ubas volu tran 3 (0 0 thu);
trac cach uframe;
*
* plate
*
c12 = p0 d 30 p1;
placen = c12 tran 30 (0 0.15 0);
*
p4p = 0.3125 0 0;
c3p = p3 d 4 p4p;
p5p = 0.3125 0.3125 0;
c4p = p4p d 16 p5p;
c5p = p5p d 9 p6;
cc0p = c1 et c2 et c3p et c4p et c5p et c6 et c7 et c8;
*
ccp = cc0p et c9 et c10;
elim tol ccp;
*
bas1p = surf ccp plan;
bas1p = orie bas1p dire pz;
bas2p = bas1p syme plan p8 p5 (p8 plus pz);
plate = bas1p et baspr et bas2p et baspr2 et placen;
elim tol plate;
plate = orie plate dire pz;
trac plate;
trac cach (uframe et plate);
nplate = chan poi1 plate;
*
* lower frame with bolts
*
dbolt = 0.021;
rbolt = dbolt / 2.0;
*
lpb = (0.260 + rbolt) 0 0;
c11 = lpb d 1 p3;
lpre1 = c11 rota 4 180.0 pc1 pz1;
lpre2a = lpre1 tour 15.0 p0 pz;
lpre2b = lpre2a tour 180.0 pc3 (pc3 plus pz);
lpre2 = lpre2a et lpre2b;
lpre3 = lpre2 tour 15.0 p0 pz;
lpre4 = lpre2b tour 30.0 p0 pz;
lbaspr = lpre1 et lpre2 et lpre3 et lpre4;
lbaspr = orie lbaspr dire pz;
elim tol (bas1 et lbaspr);
*
lbaspr2 = lbaspr syme plan p8 p5 (p8 plus pz);
lbas = bas1 et lbaspr et bas2 et lbaspr2;
elim tol lbas;
lbas = orie lbas dire pz;
trac lbas;
*
thl = 0.025;
lobas = lbas plus (0 0 (0 - gap2 - thl));
lframe = lobas volu tran 3 (0 0 thl);
trac cach lframe;
trac cach (uframe et lframe);
trac cach (uframe et plate et lframe);
*
lpb2 = lpb tour 90.0 pc1 pz1;
sur11 ier = PX4CIR3D lpb lpb2 pc1 pz1 2 1.E-3;
sur12 = sur11 tour 90.0 pc1 pz1;
sur1 = sur11 et sur12;
elim tol sur1;
sur1 = orie sur1 dire pz;
*
sur2a = sur1 tour 30.0 p0 pz;
sur1s = sur1 syme plan pc1 lpb pz1;
sur1s = orie sur1s dire pz;
sur2b = sur1s tour 30.0 p0 pz;
sur2 = sur2a et sur2b;
elim tol sur2;
sur2 = orie sur2 dire pz;
sur3 = sur2 tour 30.0 p0 pz;
sur4 = sur1s tour 90.0 p0 pz;
sur1 = sur1 plus (0 0 (0 - gap2 - thl));
sur2 = sur2 plus (0 0 (0 - gap2 - thl));
sur3 = sur3 plus (0 0 (0 - gap2 - thl));
sur4 = sur4 plus (0 0 (0 - gap2 - thl));
*
hbolt = 0.06;
hboltu = hbolt - thl;
*
bolt1a = sur1 volu tran 3 (0 0 thl);
bolt1b = (sur1 plus (0 0 thl)) volu tran 4 (0 0 hboltu);
bolt1 = bolt1a et bolt1b;
elim tol (bolt1 et lframe);
bolt2a = sur2 volu tran 3 (0 0 thl);
bolt2b = (sur2 plus (0 0 thl)) volu tran 4 (0 0 hboltu);
bolt2 = bolt2a et bolt2b;
elim tol (bolt2 et lframe);
bolt3a = sur3 volu tran 3 (0 0 thl);
bolt3b = (sur3 plus (0 0 thl)) volu tran 4 (0 0 hboltu);
bolt3 = bolt3a et bolt3b;
elim tol (bolt3 et lframe);
bolt4a = sur4 volu tran 3 (0 0 thl);
bolt4b = (sur4 plus (0 0 thl)) volu tran 4 (0 0 hboltu);
bolt4 = bolt4a et bolt4b;
elim tol (bolt4 et lframe);
bolts = bolt1 et bolt2 et bolt3 et bolt4;
lframeb = lframe et bolts;
elim tol lframeb;
trac cach lframeb;
trac cach (lframeb et plate);
trac cach (lframeb et plate et uframe);
*
* pressurized clamping surfaces
*
presa = pre1 plus (0 0 (gap2 + thu));
presa = presa orie dire pz;
presb = pre3 plus (0 0 (gap2 + thu));
presb = presb orie dire pz;
presc = presb tour 30.0 p0 pz;
presd = (presa tour 90.0 p0 pz) syme plan p0 (0 1 0) pz;
presd = presd orie dire pz;
presur = presa et presb et presc et presd;
elim tol (presur et uframe);
trac cach (presur et uframe);
*
* pressurized (exposed) part of the plate
*
preplat = placen coul roug;
trac cach (plate et preplat);
*
ecub8 = (lframeb et uframe) elem cub8;
epri6 = (lframeb et uframe) elem pri6;
equa4 = plate elem qua4;
etri3 = plate elem tri3;
*
mesh = lframeb et plate et uframe et presur et preplat
       et nplate et ecub8 et epri6 et equa4 et etri3;
tass mesh noop;
sauv form mesh;
trac cach mesh;
*
fin;
```

stpm02.epx

```
STPM02
ECHO
!CONV win
CAST MESH
EROS 1.0
TRID LAGR
GEOM CUB8 ecub8
     PR6  epri6
     Q4GS equa4
     T3GS etri3
     PMAT nplate
     CL3D presur preplat
TERM
COMP EPAI 0.8e-3 LECT plate nplate TERM
     COUL TURQ LECT lframeb TERM
          BLEU LECT uframe TERM
          VERT LECT plate TERM
          ROSE LECT presur TERM
          ROUG LECT preplat nplate TERM
     NGRO 3 'bloz'  LECT lframeb TERM COND Z LT -0.0253
            'symx'  LECT mesh TERM    COND X LT 0.0001
            'symy'  LECT mesh TERM    COND Y LT 0.0001
     GROU 1 'prpl'  LECT preplat TERM COND NEAR POIN 0.0 0.0 0.0
!    NGRO 3 'centd' LECT PART 2 TERM COND NEAR POIN 0.0 0.0 0.0
!           'centl' LECT PART 3 TERM COND NEAR POIN 0.0 0.0 0.0
!           'axis'  LECT PART 2 TERM
!                   COND LINE X1 -0.2 Y1 -0.001 Z1 -0.001
!                             X2  0.2 Y2  0.001 Z2  0.001 TOL 1.E-5
     ORIE INVE LECT preplat TERM
INCLUDE 'p77_75.txt'
MATE VPJC RO 7850.0 YOUN 2.1E11 NU 0.33 ELAS 3.257E8 mxit 500
          QR1 2.348E8 CR1 56.2 QR2 4.457E8 CR2 4.7
          PDOT 5.E-4 C 1.E-2 TQ 0.9 CP 452.0
          TM 1800.0 M 1.0 DC 1.0 WC 555.0E6
          LECT lframeb plate uframe TERM
     MASS 0.0 YOUN 2.1E11 NU 0.33
          LECT nplate TERM
     IMPE PIMP RO 7850.0 PRES 43.1e6 PREF 1.0E5
          LECT presur TERM
```

```
     IMPE PIMP RO 7850 PRES -1.0 PREF 1.0E5 FONC 1
          LECT preplat TERM
OPTI PINS ASN
LINK COUP SPLT NONE
     BLOQ 3 LECT bloz TERM
     CONT SPLA NX 1 NY 0 NZ 0 LECT symx TERM
     CONT SPLA NX 0 NY 1 NZ 0 LECT symy TERM
LINK DECO
     PINB PENA SFAC 1.0
          BODY FROT MUST 0.15 MUDY 0.1 GAMM 0 MLEV 5
          LECT lframeb TERM
          BODY FROT MUST 0.15 MUDY 0.1 GAMM 0 MLEV 5
          LECT uframe TERM
          BODY FROT MUST 0.15 MUDY 0.1 GAMM 0 DIAM 0.0008
          LECT nplate TERM
ECRI DEPL VITE ECRO FAIL TFRE 1.E-3
          POIN LECT 1 TERM
          ELEM LECT 1 TERM
     FICH SPLI ALIC TFRE 0.1E-3
     FICH ALIC TEMP TFRE 0.01E-3
          POIN LECT p0 TERM
          ELEM LECT prpl TERM
OPTI NOTE CSTA 0.8
     LOG 1
     JAUM
     LMST
     PINS GRID DPIN 1.01
    !lnks stat
CALC TINI 0 TEND 10.0E-3
FIN
```

stpm02a.epx

```
STPM02A
ECHO
 CONV WIN
RESU SPLI ALIC ’stpm02.ali’ GARD PSCR
SORT VISU NSTO 1
PLAY
CAME   1 EYE  -4.12376E-01 -4.12376E-01  6.82373E-01
!        Q     8.37319E-01  3.90448E-01 -1.61729E-01 -3.46829E-01
         VIEW  5.41675E-01  5.41675E-01 -6.42788E-01
         RIGH  7.07107E-01 -7.07107E-01  5.55112E-17
         UP    4.54519E-01  4.54519E-01  7.66044E-01
         FOV   2.48819E+01
!NAVIGATION MODE: ROTATING CAMERA
!CENTER :  1.62658E-01  1.62658E-01  0.00000E+00
!RSPHERE:  2.41269E-01
!RADIUS :  1.06158E+00
!ASPECT :  1.00000E+00
!NEAR   :  8.20315E-01
!FAR    :  1.54412E+00
SCEN GEOM NAVI FREE
          FACE SBAC
     LIMA ON
SLER CAM1 1 NFRA 1
FREQ 1
TRAC OFFS FICH AVI NOCL NFTO 101 FPS 15 KFRE 10 COMP -1 NFAI
     OBJE LECT lframeb plate uframe TERM REND
!    OBJE LECT lframeb plate TERM REND
GOTR LOOP 99 OFFS FICH AVI CONT NOCL NFAI
     OBJE LECT lframeb plate uframe TERM REND
!    OBJE LECT lframeb plate TERM REND
GO
TRAC OFFS FICH AVI CONT NFAI
     OBJE LECT lframeb plate uframe TERM REND
!    OBJE LECT lframeb plate TERM REND
ENDPLAY
FIN
```

stpm02b.epx

```
STPM02B
ECHO
RESU ALIC TEMP ’stpm02.alt’ GARD PSCR
SORT GRAP
AXTE 1.0 ’Time [s]’
COUR 12 ’DisCen’ DEPL COMP 3 POIN LECT p0 TERM
TRAC 12 AXES 1.0 ’DisCen’ YZER
     COLO bleu
     THIC 0.8
LIST 12 AXES 1.0 ’DisCen’ YZER
RCOU  1 ’DisCen’ FICH ’clam01b.pun’ RENA ’DisCenC01’
RCOU  2 ’DisCen’ FICH ’clam02b.pun’ RENA ’DisCenC02’
RCOU  5 ’DisCen’ FICH ’clam05b.pun’ RENA ’DisCenC05’
RCOU  6 ’DisCen’ FICH ’clam06b.pun’ RENA ’DisCenC06’
RCOU 11 ’DisCen’ FICH ’stpm01b.pun’ RENA ’DisCenS01’
TRAC 1 2 5 6 11 12 AXES 1.0 ’DisCen’ YZER
COLO VERT ROUG BLEU ROSE TURQ NOIR
COUR 21 ’DisCenS1i’ MULC 11 -1.0
COUR 22 ’DisCenS2i’ MULC 12 -1.0
TRAC 1 2 5 6 21 22 AXES 1.0 ’DisCen’ YZER
COLO VERT ROUG BLEU ROSE TURQ NOIR
FIN
```

List of input files

GETTING IN TOUCH WITH THE EU

In person

All over the European Union there are hundreds of Europe Direct centres. You can find the address of the centre nearest you online (european-union.europa.eu/contact-eu/meet-us_en).

On the phone or in writing

Europe Direct is a service that answers your questions about the European Union. You can contact this service:

— by freephone: 00 800 6 7 8 9 10 11 (certain operators may charge for these calls),

— at the following standard number: +32 22999696,

— via the following form: european-union.europa.eu/contact-eu/write-us_en.

FINDING INFORMATION ABOUT THE EU

Online

Information about the European Union in all the official languages of the EU is available on the Europa website (european-union.europa.eu).

EU publications

You can view or order EU publications at op.europa.eu/en/publications. Multiple copies of free publications can be obtained by contacting Europe Direct or your local documentation centre (european-union.europa.eu/contact-eu/meet-us_en).

EU law and related documents

For access to legal information from the EU, including all EU law since 1951 in all the official language versions, go to EUR-Lex (eur-lex.europa.eu).

Open data from the EU

The portal data.europa.eu provides access to open datasets from the EU institutions, bodies and agencies. These can be downloaded and reused for free, for both commercial and non-commercial purposes. The portal also provides access to a wealth of datasets from European countries.

Science for policy

The Joint Research Centre (JRC) provides independent, evidence-based knowledge and science, supporting EU policies to positively impact society

EU Science Hub
joint-research-centre.ec.europa.eu

@EU_ScienceHub
EU Science Hub - Joint Research Centre
EU Science, Research and Innovation
EU Science Hub
@eu_science

Publications Office
of the European Union