PLEXIS-3C: A Computer Code for Fast Dynamic Problems in Structures and Fluids

H. Bung
CEA-CEN Saclay, Gif sur Yvette, France
F. Casadei, J. P. Halleux
Commission of the European Communities, Ispra, Italy
M. Lepareux
CEA-CEN Saclay, Gif sur Yvette, France

INTRODUCTION

In 1986 the Commissariat à l'Energie Atomique (CEA), CEN Saclay (F) and the Commission of the European Communities (CEC), JRC Ispra (I), started a collaborative project aiming at developing a new computer code, now called PLEXIS-3C. The program, based on finite elements, treats transient dynamic problems in structures and accounts also for the presence of compressible fluids. Both institutions had previous experience in the field of fast dynamics, which had led to the development of CEA's PLEXUS (part of the CASTEM system of codes) and of JRC's EURDYN series of codes.

While CEA had been involved in many different practical applications and had from the very beginning set up a suitable general-purpose informatics framework built around a well-defined (though elementary) data-structure and resulting in a rather large computer code (PLEXUS), JRC had concentrated its efforts mainly on the development of effective and at times innovative models and computational techniques, which had been implemented and validated separately in a number of specialised software modules (EURDYN-01,-02,-03,-1D,-1M). The main idea of the collaboration was therefore to merge the data structure and programming environment, including general modelling features, that existed on one side (in practice, an informatics skeleton based on PLEXUS) with some more specific models available from the other side, thus producing a common integrated software tool that would also provide an excellent basis for future developments. Indeed, new models have already been developed within the collaboration. The joint product, PLEXIS-3C (the name is an acronym for PLEXUS (CEN Saclay) - EURDYN (JRC Ispra) Collaboration CEA-CEC), is of course available to all EEC member countries. On the other hand, since all joint developments have been maintained compatible with (and included into) PLEXUS, they are readily available within CEA's proprietary product also.

At present, PLEXIS-3C contains models for 1D, 2D and 3D dynamic analysis of structures. Compressible fluids can be treated in 2D while 3D models are currently being completed and validated. Nonlinearities of material (plasticity, viscoplasticity) and of geometry (large displacements and rotations, large strains) are accounted for. Models for fluid-structure interaction phenomena are also available. The development of the program will be continued with the introduction of new material models (possibly concrete) and the improvement of present capabilities, especially in the 3D domain of applications.

FORMULATION

For reasons of space, the following discussion of the formulation adopted in PLEXIS-3C is reduced to only a few essential points. Readers interested in more details should refer to the publications quoted hereafter. The formulation adopted for the structural modelling is discussed first. This formulation is based on finite element spatial discretisation of the equilibrium equations and assumes, as usual, a Lagrangian representation, i.e. the finite element mesh follows the motion of material particles. By

expressing equilibrium (in a dynamic sense) in the current configuration, the following set of discrete differential equations in time can be obtained (see, e.g., Zienkiewicz, 1977) :

$$\mathbf{M}\ddot{\mathbf{u}} = \mathbf{f}^{ext} - \sum_e \int_{V_e} \mathbf{B}^T \boldsymbol{\sigma}\, dV, \tag{1}$$

where : $\boldsymbol{\sigma}$ is a matrix representing the approximated true (Cauchy) stresses acting on the structure, $\mathbf{M}$ is a mass matrix, $\mathbf{u}$ is the nodal acceleration vector, $\mathbf{f}^{ext}$ is a vector of externally applied loads, $\mathbf{B}$ is the matrix of shape functions derivatives and V_e represents the element (e) volume in the current configuration. The set of equations (1) is decoupled because the matrix $\mathbf{M}$ can be diagonalised by a suitable lumping process, so that the accelerations $\ddot{\mathbf{u}}$ are directly obtained, without any need for system solutions.

Time integration of (1) is achieved via a central difference scheme. By assuming that all discretized quantities (displacements $\mathbf{u}$, velocities $\dot{\mathbf{u}}$, accelerations $\ddot{\mathbf{u}}$, stresses $\boldsymbol{\sigma}$ and related variables) are known at time t^n, and denoted by the upper suffix ^{n}, first a modified velocity is introduced :

$$\mathbf{v}^{n+1/2} = \dot{\mathbf{u}}^n + \frac{\Delta t}{2}\ddot{\mathbf{u}}^n, \tag{2}$$

which is denoted $\mathbf{v}$ in order to stress the difference with $\mathbf{u}$. This is the constant velocity that would transform configuration n into configuration $n+1$ over a time interval Δt in the discretization process. The new displacements are thus given by :

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t\, \mathbf{v}^{n+1/2}. \tag{3}$$

On this new configuration, the stress $\boldsymbol{\sigma}^{n+1}$ can now be evaluated by application of the constitutive relations. Then, the new field of accelerations $\ddot{\mathbf{u}}^{n+1}$ can be directly computed via the discretized equilibrium equations (1) and finally the new velocities are obtained as :

$$\dot{\mathbf{u}}^{n+1} = \dot{\mathbf{u}}^n + \frac{\Delta t}{2}\left(\ddot{\mathbf{u}}^n + \ddot{\mathbf{u}}^{n+1}\right). \tag{4}$$

It is important to note that in the time integration process the new configuration, induced by the displacements $\mathbf{u}^{n+1}$ is obtained first (at the initial time, the configuration is known by definition), then equilibrium is applied on the current configuration, while the velocities $\dot{\mathbf{u}}^{n+1}$ which correspond to this new configuration are computed only at the end of the time stepping procedure. A flowchart of the time integration procedure is given in Table 1. This time integration scheme is explicit in that all quantities in the right-hand side terms are known when the equations are applied : no system solver is needed. Moreover, the central difference operator introduces no damping in the solution. An analysis of spectral properties of time integrators and of mass representations (Key, 1980) reveals that explicit integration schemes can indeed be conveniently associated with lumped (diagonal) mass representations, while implicit schemes should rather be used in conjunction with consistent mass matrices. The method used in PLEXIS-3C to obtain the lumped mass matrix corresponds to adding all terms of each row.

A feature of explicit time integration schemes that is sometimes presented as a major drawback is their conditional stability, which requires the use of extremely small time steps in the discrete solution. It may be noted, however, that the class of problems of interest here would require, for accuracy reasons, very short steps even with implicit schemes, which are mainly unconditionally stable. In fact, as a general rule, the higher are the frequencies that have to be "captured" by the numerical solution, the smaller is the time interval to be adopted in implicit methods. In the presence of wave propagation phenomena (steep fronts) or whenever high frequencies become important, explicit integration is preferable, for a variety of reasons : a) there is no need for system solvers, b) the computational effort grows only proportionally to the number of degrees of freedom (no band-width problems), a useful property when solving very large problems, c) easier implementation and treatment of non-linearities : non-iterative treatment of plasticity and large deformations, simplified handling of fluid-structure coupling, of contact-impact problems, of friction boundary conditions, etc.

To model the fluid domain, an Arbitrary Lagrangian Eulerian (ALE) formulation is adopted (Donea et al., 1982). In this formulation the finite element mesh is not tied to material particles

as in the Lagrangian formulation but its velocity is prescribed by the program so as to maintain the distortions within acceptable bounds. As a consequence, fluid material continuously flows across element boundaries and transport terms have to be introduced in the governing equations which prescribe the conservation of mass, momentum and energy. Because of the presence of these terms and in order to overcome numerical problems (noisy solutions) arising in cases where the pressures and the velocities are low, a special treatment is adopted for the integration of the equations for the fluid domain. The time integration procedure is subdivided into three phases at each time step ; in the first phase the equations for the purely Lagrangian formulation are solved, in the middle phase an implicit pressure calculation is introduced which permits solutions to be obtained at all flow speeds and in the last phase the convective contributions are taken into account. With PLEXIS-3C it is of course also possible to perform a purely Eulerian calculation (fixed mesh), which is sometimes useful and adequate for cases involving fluid flow, for example when no free surfaces are present.

PROGRAM LAYOUT

As mentioned in the introductory remarks, PLEXIS-3C was conceived and designed from the very beginning as a general-purpose program for the treatment of transient dynamic problems. Its generality lies in the fact that it can accommodate many different models. The introduction of a new model (e.g., a new type of finite element or a new material model) is generally straightforward since the necessary programming structure is available. Therefore, the program is considered also an excellent environment for the set-up and testing of new models. There are, of course, limitations, but they are dictated only by the nature of the models : the most important requirement is compatibility with an explicit time integration, although it is possible to accommodate for some degree of impliciteness in the solution (consider for example the treatment of fluids in the ALE formulation as discussed in the previous section).

Being a joint project, PLEXIS-3C has also been designed so as to facilitate multiple developments performed by different groups (actually, on different sites) at the same time. The basic ingredient here is orthogonality : while most of the time the single models under development have an essentially orthogonal nature in that they do not (or only weakly) interact with each other, great care must be taken to avoid introducing couplings and side effects due to inappropriate programming and implementation. In practice a few simple rules have been established that allow for (almost) painless joint development. The data structure is of primary importance in this respect ; if a new model fully conforms to the data structure, there is no danger of unwanted interactions. Only those developments which require modifications or enhancements in the data structure are potentially dangerous and should therefore be considered with the maximum care since they may necessitate some kind of negotiation within the developing groups. In this respect sufficient information must be supplied to potential developers and appropriate checks need to be performed. Thus one remains free to concentrate the main effort on the actual models. These apparently obvious but fundamental rules have allowed for a successful joint development of PLEXIS-3C. A unique common version of the program is kept and continuously maintained and updated by using network communications where appropriate, an unusual but very effective way of organizing the work on the program, which allows the minimization of effort and almost eliminates the usual problems of compatibility and program maintenance related to the existence of many different versions.

The structure of the program which, by its several tens of thousands FORTRAN lines could at first sight appear as an obscure black box, is actually quite simple. The kernel is a single routine (first level) which contains the time loop (Table 1) and pilots the entire solution. First, the configuration is updated in a loop over the nodes, then in order to solve the equilibrium equations the new internal forces have to be calculated. This operation is piloted by a second-level routine that computes the new internal forces (via the updated stresses) in the elements. This routine consists in practice of a loop on all the elements and it calls, for each finite element type, the corresponding model routine (third level). Finally, since each element type can be associated with various different material types (so as to achieve full generality and orthogonality), each element routine contains a branching structure which calls the appropriate constitutive model routine (fourth level) for that element, which in turn returns the updated stresses. Control is then returned to the first-level routine which assembles the

forces, computes the new accelerations (by simply dividing by the lumped masses) and finally updates the velocities (two separate loops on the nodes), before starting a new step. The treatment of essential boundary conditions is accomplished via Lagrange multipliers ; a small linear system involving only the degrees of freedom tied by the constraints is obtained which, upon solution, yields the corresponding reaction forces. Within the flowchart presented in Table 1 this fits into step 6 and does not modify the general solution procedure. The choice of the time increments to be used for the solution can either be prescribed by the user or (more often) be left to the program, which by default evaluates the global stability time step as the minimum of the estimated stable steps for each element. An interesting option offered by the program is that of time-step partitioning in space, i.e. the use of time steps varying not only in time but also over the different finite elements in the mesh. An explicit-explicit partitioning is adopted (see e.g. Key, 1986). This feature is considered essential for the solution of realistic 3D problems using explicit techniques, in that it allows for important savings in CPU time when localised mesh refinements are required for accuracy reasons. The solution algorithm used for the fluid domain (see previous section) has been successfully embedded within the general frame sketched in Table 1 ; the treatment of the three separate phases has been achieved using a double loop over the elements (not shown in Table 1, for simplicity) for the calculation of internal forces.

MODELLING

A few, particularly significant modelling aspects of PLEXIS-3C will be reviewed briefly : a) large-strain continuum and structural models and elasto-plastic material behaviour for the structural domain and b) mesh re-zoning and fluid-structure sliding models for the fluid domain. The treatment of problems involving arbitrarily large strains in solids, also in the presence of shear, is achieved by a particularly effective 9-node parabolic element (in 2D) using the Cauchy stress formulation discussed above and a careful implementation of the Zaremba-Jaumann-Noll stress rate preserving objectivity also in the presence of finite rotation increments (see Halleux and Casadei, 1984). In order to obtain maximum accuracy the stress update is performed in various stages : first, the stress is rotated by half the angle of rigid rotation, then the constitutive equation is applied and finally the second half of the rotation is applied. In structural elements (beams, plates and shells) large displacements and rotations are taken into account by a co-rotational formulation while the effect of large membrane strains is modelled by suitable variations in the element thickness. Elasto-plastic material behaviour is modelled by the radial return algorithm. Basically, a trial stress is first estimated by adding to the old stress an elastically induced stress increment based on the given strain increment and then, if the new value lies outside the yield surface, by projecting the trial stress onto the yield surface along the radial direction towards the centre of the yield surface in deviatoric space. An excellent description of this computationally attractive method which has also proved to be very accurate in practice, can be found in Hughes (1984).

The main advantage of the ALE formulation (see the previous sections) is that it allows for a free choice of mesh velocity in the bulk of the fluid domain. This velocity can therefore be assigned (a feature sometimes called re-zoning) in such a way that mesh distortions be minimized and that the accuracy of finite element representation be retained. In PLEXIS-3C several re-zoning techniques, both 'automatic' and 'manual' (as defined below) are available. An automatic algorithm, in the sense that it requires no input effort by the user, is normally adopted when tackling a new problem and should be adequate for the majority of the cases (see, e.g., the calculation described in the companion paper by Casadei et al., 1989). If re-zoning difficulties are encountered during the solution, the user can always resort to prescribing the motion of certain (usually a few) nodes by one of the manual options available in the code ; these facilities, based on specialised operators, ensure the regularity of the mesh inside a closed contour (in 2D or 3D) by imposing homeomorphic deformation.

The automatic re-zoning algorithm was originally developed by Giuliani (1982). To each node that has to be re-zoned an 'influence domain', based on topological connections (element sides containing that node), is associated. This domain is subdivided into triangles (in 2D) or tetrahedra (in 3D). The algorithm tries to keep each influence domain as regular as possible by minimizing both the 'squeeze' and the 'distortion' (see below, for an intuitive definition) of each triangle (or tetrahedron). In 2D the squeeze of a triangle is defined as the difference between its height and the mean height of all

the triangles belonging to the influence domain, while the distortion is the distance between the node position before re-zoning and the position it would occupy if the triangle became isosceles (by keeping the same area).

While the bulk of the fluid domain is usually treated with the ALE formulation and free surfaces or fluid/fluid interfaces (without mixing) are conveniently represented as Lagrangian, suitable models have to be introduced for the description of the interaction between the fluid and the structures. To this end, the models originally developed in the EURDYN-1M code (see Donea et al., 1982) have been introduced in PLEXIS-3C for the 2D case. A generalization of the same models for the 3D case (Donea and Giuliani, 1988) is being implemented in the code and validated. Two distinct situations are considered : a) interaction along permanently submerged portions of the structure and b) fluid-structure (F-S) interfaces in the vicinity of free surfaces or other Lagrangian surfaces. In the first case, the freedom allowed by the ALE formulation is exploited to keep the fluid mesh (ALE nodes) permanently aligned with the structural mesh (Lagrangian nodes) along the interface. Fluid sliding along the interface is thus modelled by standard mass transfer between neighbouring ALE elements (ALE sliding). The motions of the fluid and of the structure are coupled in the direction normal to the interface while each component is free to move on its own in the tangential direction. In the presence of curved interfaces (represented by corners in the finite element discretization) the direction of the tangential (unconstrained) fluid velocity can be determined, for the linear elements used in PLEXIS-3C, by the condition that no fluid should be lost or gained at corners, i.e. the fluid flowing out of one element should counterbalance the fluid entering into the next element along the interface.

The second situation occurs when, because of large F-S relative displacements and/or the vicinity of a Lagrangian interface, it is no longer possible to keep the fluid nodes permanently aligned with the structural nodes. In this case the fluid nodes have to be treated as Lagrangian and a Lagrangian sliding model is employed to describe their motion relative to the structure. First, a virtual fluid surface, having nodes aligned with those of the structure, is considered ; the relevant quantities at the virtual nodes are obtained by proper interpolation of the corresponding values at actual fluid nodes. Then interaction forces are computed similarly to the case of ALE sliding. Finally, interpolation is again used to calculate the interaction forces at the actual nodes. These models can be used also to describe the sliding between two fluids along a Lagrangian interface.

REFERENCES

CASADEI, F., DANERI, A. and TOSELLI, G. :" Use of PLEXUS as a LMFBR primary containment code for the CONT benchmark problem", Paper E-174, SMiRT-10, Anaheim, Aug. 14–18, 1989.

CASADEI, F. and HALLEUX, J.P. : " Validation of the PLEXIS-3C computer code on academic and real test cases in structural dynamics", Paper B-83, SMiRT-10, Anaheim, Aug. 14–18, 1989.

DONEA, J., GIULIANI, S. and HALLEUX, J.P. : "An Arbitrary Lagrangian Eulerian finite element method for transient dynamic fluid-structure interactions", *Comp. Meth. Appl. Mech. Eng., 33*, pp. 689–723, 1982.

DONEA, J., and GIULIANI, S. :"An explicit ALE finite element formulation for 3D transient dynamic fluid-structure interaction problems", EUR Report No. 11936/EN, 1988.

GIULIANI, S. :"An algorithm for continuous re-zoning of the hydrodynamic grid in Arbitrary Lagrangian Eulerian codes", *Nucl. Eng. Des., 72*, pp. 205–212, 1982.

HALLEUX, J.P. and CASADEI, F. : "Transient large-strain finite element analysis of solids", in Computational Methods for Non-Linear Problems (Eds. C. Taylor, D.R.J. Owen and E. Hinton), Pineridge Press, Swansea, 1987.

HUGHES, T.J.R. : "Numerical implementation of constitutive models : rate independent deviatoric plasticity", in Proc. of the Workshop on Theoretical Foundation for Large-Scale Computations of Nonlinear Material Behaviour, Northwestern Univ., Evanston, Illinois, Oct. 24–26, 1983 (Eds. S. Nemat-Nasser, R.J. Asaro and G.A. Hegemier), Martinus Nijhoff Publ., pp. 29–57, Dordrecht 1984.

KEY, S.W. : "Transient response by time integration : review of implicit and explicit operators", in Advanced Structural Dynamics (Ed. J. Donea), pp. 71–95, Applied Science, London, 1980.

KEY, S.W. : "Improvements in transient dynamic time integration with application to spent fuel

shipping cask impact analyses", in Proc. of the Eur.-US Symposium on Finite Element Methods for Non-linear problems, Trondheim, Aug. 12–16, 1985 (Eds. P. G. Bergan, K.J. Bathe and W. Wunderlich), Springer, Berlin, 1986.
ZIENKIEWICZ, O.C. : "The finite element method", 3rd Edition, McGraw-Hill, 1977.

Table 1 - Schematic flowchart of the PLEXIS-3C program.

1. Set initial conditions : $n = 0$, $\Delta t = 0$, $t^n = t_0$, $\mathbf{x}^n = \mathbf{x}_0$, $\boldsymbol{\sigma}^n = \boldsymbol{\sigma}_0$, $\dot{\mathbf{u}}^n = \dot{\mathbf{u}}_0$, $W^{int} = 0$, $W^{ext} = W^{kin}$.
2. GO TO 5.
3. (Note : $\mathbf{x}^n$, $\boldsymbol{\sigma}^n$, $\ddot{\mathbf{u}}^n$ and $\mathbf{v}^{n+1/2} = \dot{\mathbf{u}}^n + \frac{\Delta t}{2}\ddot{\mathbf{u}}^n$ are known quantities);
   $n = n + 1$, $\Delta t = \Delta t^{new}$ and $t^n = t^n + \Delta t$.
4. Update configuration : $\mathbf{x}^n = \mathbf{x}^{n-1} + \Delta t\ \mathbf{v}^{n+1/2}$ (loop over nodes).
5. Compute internal forces : $\mathbf{f}^{int} = \sum_e \int_{V_e} \mathbf{B}^T \boldsymbol{\sigma}^n dV$ (loop over elements);
   V^e and $\mathbf{B}^T$ are evaluated on the new configuration (n);
   $\boldsymbol{\sigma}^n = \boldsymbol{\sigma}^{n-1} + \Delta\boldsymbol{\sigma}$;
   While computing element stresses, evaluate Δt^{new} and add internal energy increment to W^{int}.
6. Assemble internal forces, evaluate and assemble external forces $\mathbf{f}^{ext}$ and add external work increment to W^{ext}.
7. Compute accelerations and update velocities :
   $\ddot{\mathbf{u}}^n = \mathbf{M}^{-1}\left(\mathbf{f}^{ext,n} - \mathbf{f}^{int,n}\right)$ (loop over nodes) ;
   $\mathbf{v}^{n+1/2} = \mathbf{v}^{n-1/2} + \frac{\Delta t + \Delta t^{new}}{2}\ddot{\mathbf{u}}^n$ (loop over nodes).
8. If no output required, GO TO 3.
9. Compute kinetic energy and check energy balance : $W^{ext} \approx W^{int} + W^{kin}$. Print required output.
10. If final time not reached, GO TO 3.

LEGEND of SYMBOLS			
$\mathbf{x}$	Current coordinate	W^{ext}	External work
$\mathbf{u}$	Displacement	W^{kin}	Kinetic energy
$\mathbf{v}$	Velocity	V	Volume
$\boldsymbol{\sigma}$	Cauchy stress	e	Finite element
$\mathbf{f}$	Force	n	Step counter
$\mathbf{B}$	Matrix of shape function derivatives	t	Time
		Δt	Time step
$\mathbf{M}$	Mass matrix	$\cdot$	Time derivative
W^{int}	Internal energy	^{T}	Matrix transpose