General information on the combined gradient: GCPC *Aster* and use of PETSc

Summary:

Within the framework of thermomechanical simulations, the main part of the costs calculation often comes from the construction and the resolution of the linear systems. The choice of the good linear solvor is thus paramount, on the one hand for its speed, but also for its robustness and the place memory which it requires. Each time, a compromise is to be operated between these constraints.
For 60 years two types of solveurs have disputed supremacy in the field, direct and those iterative. From where, by precaution, a diversified offer of the codes of mechanics on the matter. *Code_acefor the third time* do not escape the rule. He proposes three solveurs direct (Gauss [R6.02.01], multifrontale native [R6.02.02] and external product MUMPS [R6.02.03]), of solveurs iterative of type Krylov (GCPC Aster and all those of the PETSc bookstore).

In it document, **one details** from a theoretical point of view, algorithmic and modelings *Code_hasster*, **the fundamental ones of the GCPC and other solveurs of Krylov established in PETSc**. In particular, one clarifies their links with the other linear solveurs of the code and the methods of continuous optimization. One continues by their difficulties of establishment, their parameter settings, their perimeters like some advices of use.

For more details and advices on the employment of the linear solveurs one will be able to consult the specific notes of use [U4.50.01]/[U2.08.03]. The related problems of improvement of performances (RAM/CPU) of a calculation and, use of parallelism, are also the object of detailed notes: [U1.03.03] and [U2.08.06].

Contents

1 Problems

In digital simulation of physical phenomena, **a cost important calculation often comes from the construction and the resolution of linear systems**. The mechanics of the structures does not escape the rule! The cost of the construction of the system depends amongst points on integration and complexity on the laws on behavior, while that of the resolution is related on the number of unknown factors, modelings selected and topology (bandwidth, conditioning). When the number of unknown factors explodes, the second stage becomes dominating[1] and it is thus the latter which mainly will interest us here. Moreover, when it is possible to be more performing on this phase of resolution, thanks to the access to a parallel machine, this asset can be propagated with the phase of constitution of the system itself (elementary calculations and assemblies, cf [U2.08.06]).
These **linear inversions of systems in fact omnipresent in the codes and are often hidden with deepest of other digital algorithms**: non-linear diagram, integration in time, analyze modal…. One seeks, for example, the vector of nodal displacements (or of the increments of displacement) u checking a linear system of the type

$$Ku = f \tag{1-1}$$

with K a matrix of rigidity and f a vector representing the application of forces generalized to the mechanical system.

In a general way, **resolution of this kind of problem requires one (more) broad questioning** that it does not appear to with it:

- Does one have access to the matrix or knows one simply his action on a vector?
- Is this matrix it digs or dense?
- Which are its digital properties (symmetry, definite positive…) and structural (real/complex, by bands, blocks.)?
- Please one solve only one system (1-1), several into simultaneous[2] or in a consecutive way[3] ? Even several different and successive systems to which the matrices are very close[4] ?
- In the case of successive resolutions, can one re-use preceding results in order to facilitate the next resolutions (cf technique of restarting, partial factorization)?
- Which is the order of magnitude of the size of the problem, the matrix and of its factorized compared to capacities for treatment of the CPU and the associated memories (RAM, disc)?
- Does one want a solution very precise or simply an estimate (cf encased solveurs)?
- One has access to bookstores of linear algebra (and with their pre-necessary MPI, BLAS, LAPACK…) or does one have to call on products “house”?

In *Code_Aster* , one explicitly builds the matrix and one stores it with the format MORSE[5] . With most modelings, the matrix hollow (because of discretization by finite elements), is potentially badly conditioned[6] and often real, symmetrical and indefinite[7] . Into nonlinear, modal or during thermomechanical chainings, one often deals with problems of the type “multiple second members”. The discrete methods of contact-friction benefit from faculties of partial factorization. In addition, *Code_Aster* use also scenarios of simultaneous resolutions (complements of Schur of the contact and under-structuring…).

1 For *Code_Aster*, to see the study of ‘profiling’ led by N.Anfaoui [Anf03].
2 Even matrix but several second independent members; Cf construction of a complement of Schur.
3 Problem of the multiples type second members: even matrix but several second successive and interdependent members; Cf algorithm of Newton without reactualization of the tangent matrix.
4 Problem of the multiples type first members: several matrices and second members successive and interdependent, but with close matrices “spectralement”; Cf algorithm of Newton with reactualization of the tangent matrix.
5 Known as still SCC for ‘Symmetric Compressed Column storage’.
6 In mechanics of the structures conditioning $\eta(K)$ is known to be rather bad. It can vary, typically, of 10^5 to 10^{12} and the fact of refining the grid, of using stretched elements or structural elements has dramatic consequences on this figure (cf B.Smith. *With parallel iterative implementation of year substructuring algorithm for problems in 3D.* SIAM J.Sci.Comput., **14** (1992), pp406-423. §3.1 or I.FRIED. *Condition of finite element matrices generated from nonuniform meshes*. AIAA J., **10** (1972), pp219-221.)
7 The indefinite character rather than definite positive is with the addition of additional variables (known as “of Lagrange”) to impose simple or generalized limiting conditions of Dirichlet [R3.03.01].

As for the sizes of the problems, even if they swell year by year, they remain modest compared to the CFD: about the million unknown factors but for hundreds of step of time or iterations of Newton.
In addition, of one **point of view "middleware and hardware"** , the code is pressed from now on on many optimized and perennial bookstores (MPI, BLAS, LAPACK, MONGREL, SCOTCH TAPE, PETSc, MUMPS…) and is used mainly on clusters of SMP (fast networks, great RAM storage capacity and disc). One thus seeks especially to optimize the use of the linear solveurs accordingly.

For 60 years, two types of techniques have disputed supremacy in the field, the direct solveurs and those iterative (cf [Che05] [Dav06] [Duf06] [Gol96] [LT98] [Liu89] [Meu99] [Saa03]).
First are robust and lead in a finished number of operations (theoretically) known by advance. Their theory is relatively well completed and their variation according to grinds standard matrices and software architectures is very complete. In particular, their algorithmic multiniveau is well adapted to the hierarchies memories of the current machines. However, they require storage capacities which grow quickly with the size of the problem what limits the extensibility of their parallelism[8] . Even if this parallelism can break up into several independent layers, thus gearing down the performances.
On the other hand, them **iterative methods** are more "scalables" when one increases the number of processors. Their theory abounds in many "opened problems", especially into arithmetic finished. In practice, their convergence in a "reasonable" number of iterations, is not always acquired, it depends on the structure of the matrix, the starting point, the criterion of stop… This kind of solveurs has more difficulty boring in mechanics of the industrial structures where one often cumulates heterogeneities, non-linearities and junctions of models which cause to become gangrenous the conditioning of the operator of work. In addition, they are not adapted to solve the problems of the type effectively "multiple second members". Out those are very frequent in algorithmic mechanical simulations.
Contrary to their direct counterparts, it is not possible to propose the iterative solvor who will solve any linear system. The adequacy of the type of algorithm to a class of problems is done on a case-by-case basis. They present, nevertheless, other advantages which historically gave them established among for certain applications. With management equivalent memory, they require some less than the direct solveurs, because one has right need to know the action of the matrix on an unspecified vector, without having truly to store it. In addition, one is not subjected to the "diktat" of the phenomenon of filling which deteriorates the profile of the matrices, one can effectively exploit the hollow character of the operators and control the precision of the results[9] .
In the literature in digital analysis, one often grants a dominating place to the iterative solveurs in general, and, with the alternatives of the gradient combined in particular. **The most senior authors agree to say that, even if its use gains over the years and of the alternatives, of the "market shares", certain fields remain still refractory. Among those, the mechanics of the structures, the electronic simulation of circuit…**
To paraphrase these authors, the use of direct solveurs concerns the area of the technology whereas the choice of the good couple iterative method/preconditionnor is rather an art! In spite of its biblical simplicity on paper, the resolution of a system linear, even symmetrical definite positive, is not "a long quiet river". Between two evils, filling/swivelling and pre-packaging, it is necessary to choose!

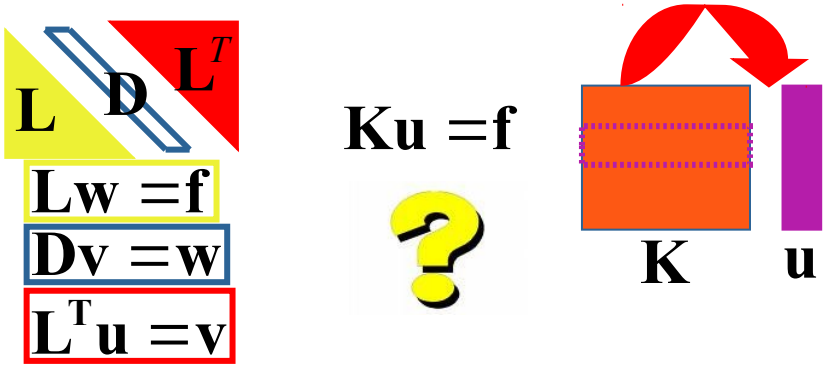


Figure 1-1. _Two classes of methods to solve

8 One also speaks about "scalability" or passage on the scale.
9 What can be very interesting within the framework of encased solveurs (e.g. Newton + GCPC), cf V.Frayssé. *The power of backward error analysis*. HDR of the Institut National Polytechnique of Toulouse (2000).

a linear system: the direct ones and the iterative ones.

Lately the arrival of the hybrid solveurs [Boi08], which mixes the two approaches to draw part of their respective advantages, upsets a little gives it. On “simulations borders” comprising of tens of million unknown factors, the combined gradient can be even competed with by these new approaches (especially for problems of the multiple type second-members).

But it makes some is generally present as a solvor of interface of these hybrids. Those can be seen like particular préconditionneurs (multilevel, multigrille, DD) of a combined gradient or all other methods of Krylov (GMRES, BiCG-Stab…). In addition, taking into account its simplicity of implementation (into sequential as in parallel) and of its weak occupation memory, there remains often it only alternate to tackle the very large problems[10] .

Note:

- *The two large families of solveurs must more be seen like complementary that like competitors. One often seeks to mix them: methods DD [Boi08], preconditionnor by incomplete factorization (cf. § 4.2) or of type multigrille [Tar09], procedure of iterative refinement at the end of the direct solvor [Boi09]…*
- *For more details on the bookstores of linear algebra implementing these methods and some results of benchmarks confronting them, one will be able to consult documentation [Boi09]. But for really drawing part of these products, they should be exploited in parallel mode. Documentation [Boi08] illustrates some aspects of parallel calculation (hardware, methodology, languages, indicators of performances) and specifies the various exploitable levels of parallelism in a code such as Code_Aster.*

Now let us approach the whole of the chapters around of which will articulate this note. After having clarified them **various formulations of the problem** (linear system, minimizations of functional calculus) and in order to better do to feel with the reader implicit subtleties of the methods of type gradient, one proposes a fast overflight the their fundamental ones: classic “steepest-descent” then them **general methods of descent** like their links with the GC. These recalls being made, it **algorithmic unfolding of the GC** becomes clear (at least it is hoped for!) and **its theoretical properties** of convergence, from orthogonality and complexity result from this. Complements are quickly brushed to put in prospect the GC with recurring concepts in digital analysis: projection of Petrov-Galerkin and space of Krylov, problem of orthogonalisation, equivalence with the method of Lanczos and properties spectral, encased solveurs and parallelism.
Then one details the “necessary evil” which constitutes it **pre-packaging of the operator of work**, some often used strategies and those retained for the native GCPC of *Code_Aster* and for the solveurs of PETSc.
One concludes by **particular difficulties of establishment in the code**, questions of **parameter setting** and of **perimeter** like by some **advices of use**.

Foot-note:

- *The effective establishment and the maintenance of the iterative solveurs in Code_Aster are the fruit of a team work: J.P.Grégoire, J.Pellet, T.DeSoza, N.Tardieu, O.Boiteau…*
- *In this document, a good amount of figures were borrowed from the introductory paper of J.R. Shewchuk [Sh94] with its pleasant authorization: ©1994 by Jonathan Richard Shewchuk.*

10 All is relative. The current limits are rather of a few million unknown factors for a PC and several hundreds of million for a machine of computer centre. Great codes of CFD (for EDF: *Code_Saturne*, TELEMAC…) are thus completely leaned with iterative solveurs of standard combined gradient. They their make it possible to distribute floods of data/treatments on thousands of processors and to thus manage, *via* parallelism, of the problems of very big size.

2 Methods of descent

2.1 Positioning of the problem

That is to say $\mathbf{K}$ the matrix of rigidity (of size N) to reverse, if it has the "good taste" to be symmetrical definite positive (SPD in the Anglo-Saxon literature), which is often[11] the case in mechanics of the structures, one shows by simple derivation which the following problems are equivalent:

- Resolution of the usual linear system with $\mathbf{u}$ vector solution (of displacements or increments of displacements, resp. in temperature….) and f vector representing the application of forces generalized to the thermomechanical system

$$\left(P_1\right) \quad \mathbf{Ku}=f \tag{2.1-1}$$

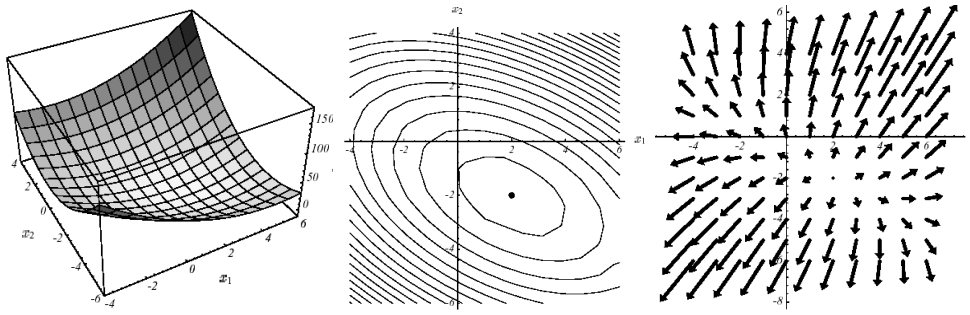
- Minimization of the quadratic functional calculus representing the energy of the system, with $<,>$ the usual Euclidean scalar product,

$$\left(P_2\right) \quad u=Arg\min_{v\in\Re^N} J\left(v\right)$$
$$\text{avec } J\left(v\right):=\frac{1}{2}\left\langle \mathbf{v},\mathbf{Kv}\right\rangle-\left\langle f,v\right\rangle=\frac{1}{2}v^T\mathbf{Kv}-f^T v \tag{2.1-2}$$

Because of "definite-positivity" of the matrix which returns J strictly convex, the cancelling vector[12] ∇J corresponds to the single one[13] total minimum $\mathbf{u}$. That is illustrated by the following, valid relation whatever $\mathbf{K}$ symmetrical,

$$J\left(\mathbf{v}\right)=J\left(\mathbf{u}\right)+\frac{1}{2}\left(\mathbf{v}-\mathbf{u}\right)^T\mathbf{K}\left(\mathbf{v}-\mathbf{u}\right) \tag{2.1-3}$$

Thus, for any vector $\mathbf{v}$ different from the solution $\mathbf{u}$, the positive definite character of the operator makes strictly positive the second term and thus $\mathbf{u}$ is also a total minimum.



(A) (b) (c)

Figure 2.1-1. Example of J quadratic in $N=2$ dimensions with $K:=\begin{bmatrix}3 & 2\\ 2 & 6\end{bmatrix}$ and $f:=\begin{bmatrix}2\\ -8\end{bmatrix}$:
graph (A), datum lines (b) and vectors gradient (c).
The spectrum of the operator is $(\lambda_1;v_1)=(7;[1,2]^T)$ and $(\lambda_2;v_2)=(2;[-2,1]^T)$.

This very important result in practice is based entirely on this famous definite-positivity "ethereal" property a little of the matrix of work. On a problem with two dimensions ($N=2$!) one can be done one of them limpid

11 For a simulation *via Code_Aster*, the matrix is often indefinite because of addition of additional variables (known as "of Lagranges") to impose limiting conditions [Pel01].

12 The cancellation of the derivative is a typical case with the convex and unconstrained case of the famous relations of Kuhn-Tucker characterizing the optimum of a differentiable problem of optimization. It is called "equation of Euler".

13 Without this convexity, one is not ensured of unicity. It is then necessary to compose with local minima!

representation (cf figure 2.1-1): the form paraboloid which focuses the single minimum at the point $[2,-2]^T$ of worthless slope.

- Minimization of the standard in energy of the error $e(v)=v-u$, more speaking for the mechanics,

$$(P_3)\quad u=Arg\min_{v\in\Re^N} E(v)$$
$$\text{avec } E(v):=\langle \mathbf{K}e(v), e(v)\rangle=(\mathbf{K}e(v))^T e(v)=\|e(v)\|_K^2 \tag{2.1-4}$$

From a mathematical point of view, it is anything else only one matric standard (licit since $\mathbf{K}$ is SPD). One often prefers to express it *via* a residue $r(v)=f-\mathbf{K}v$

$$E(v)=\langle r(v), K^{-1}r(v)\rangle=r(v)^T K^{-1} r(v) \tag{2.1-5}$$

Note:

- *The perimeter of use of the combined gradient can in fact of extending to any operator, inevitably symmetrical or not defined positive and even square! With this intention one defines the solution of (P_1) as being that, within the meaning of least squares, of the problem of minimization*

$$(P_2)'\quad u=Arg\min_{v\in\Re^N}\|\mathbf{K}v-f\|^2 \tag{2.1-6}$$

By derivation one leads to the "normal" equations known as whose operator is square, symmetrical and positive

$$(P_2)''\quad \underbrace{\mathbf{K}^T\mathbf{K}}_{\tilde{\mathbf{K}}}\mathbf{u}=\mathbf{K}^T\mathbf{f} \tag{2.1-7}$$

One can thus apply a GC to him or one steepest-descent without too much of encumber.

- *The solution of the problem (P_1) is with the intersection of N hyperplanes of dimension $N-1$. For example, in the case of figure 2.1-1, it is expressed crudely in the form of intersections of right-hand sides:*

$$\begin{cases} 3x_1+2x_2=2 \\ 2x_1+6x_2=-8 \end{cases} \tag{2.1-8}$$

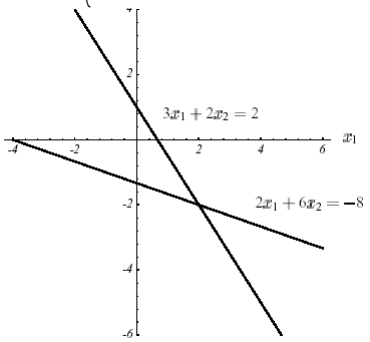


Figure 2.1-2. Resolution of the n°1 example by intersection of right-hand sides.

- *The methods of type gradient dissociate this philosophy, they lie naturally within the scope of minimization of a quadratic functional calculus, in which they were developed and intuitively understood.*

2.2 Steepest Descent

2.2.1 Principle

This last remark precedes the philosophy of the method known as "of the strongest slope", more known under its Anglo-Saxon denomination of 'Steepest Descent': one builds the continuation of reiterated $\mathbf{u}^i$ while following

the direction whereby J decrease more, at least locally, i.e. $d^i=-\nabla J^i=r^i$ with $J^i:=J\left(u^i\right)$ et $r^i:=f-\mathbf{K}\mathbf{u}^i$. With $i^{\text{ème}}$ iteration, one will thus seek to build $\mathbf{u}^{i+1}$ such as:

$$u^{i+1}:=u^i+\alpha^i r^i \tag{2.2-1}$$

and

$$J^{i+1}<J^i \tag{2.2-2}$$

Thanks to this formulation, one thus transformed a quadratic problem of minimization of size N (in J and $\mathbf{u}$) in a unidimensional minimization (in G and α)

$$\begin{aligned}&\text{Trouver } \alpha^i \text{ tel que } \alpha^i=\underset{\alpha\in[\alpha_m,\alpha_M]}{Arg\min}\, G^i(\alpha)\\&\text{avec } G^i(\alpha):=J\left(u^i+\alpha\, r^i\right)\end{aligned} \tag{2.2-3}$$

The following figures illustrate the operation of this procedure on the n°1 example: on the basis of the point $u^0=[-2,-2]^T$ (cf (A)) one seeks the optimal parameter of descent, α^0 , according to the line of greater slope r^0 ; what amounts seeking a point pertaining to the intersection of a vertical plan and a paraboloid (b), meant by the parabola (c). Crudely this point cancels the derivative of the parabola (d)

$$\frac{\partial G^0\left(\alpha^0\right)}{\partial\alpha}=0\Leftrightarrow\left\langle\nabla J\left(u^1\right),r^0\right\rangle=0\Leftrightarrow\left\langle r^1,r^0\right\rangle=0\Leftrightarrow\alpha^0:=\frac{\|r^0\|^2}{\left\langle r^0,\mathbf{K}\mathbf{r}^0\right\rangle} \tag{2.2-4}$$

This orthogonality between two successive residues (i.e successive gradients) produced a advance characteristic, known as in “zigzag”, towards the solution (E). Thus, in the case of a badly conditioned system producing narrow and lengthened ellipses[14], the iteration count necessary can be considerable (cf figure 2.2-1).

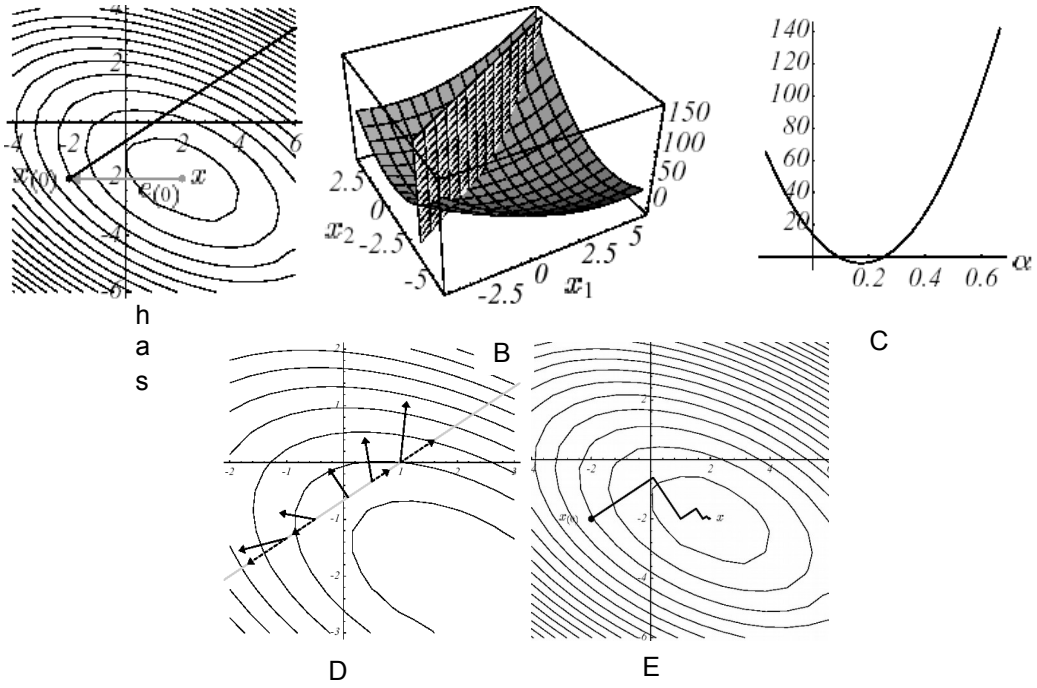


Figure 2.2-1. Illustration of Steepest Descent on the n°1 example: direction of initial descent (A), intersection of surfaces (b), corresponding parabola (c), vectors gradient and their projection along the direction of descent initial (d) and total process until convergence (E).

2.2.2 Algorithm

14 The conditioning of the operator K is written like the report of its extreme eigenvalues $\eta(K):=\frac{\lambda_{\max}}{\lambda_{\min}}$ who are themselves proportional to the axes of the ellipses. From where a direct link and visual between bad matric conditioning and narrow and tortuous valley where minimization is abused.

From where unfolding

Initialisation u^0 donné
Boucle en i
(1) $r^i = f - \mathbf{K}\mathbf{u}^i$ (nouveau résidu)
(2) $\alpha^i = \frac{\|r^i\|^2}{\langle r^i, \mathbf{K}\mathbf{r}^i \rangle}$ (paramètre optimal de descente)
(3) $u^{i+1} = u^i + \alpha^i r^i$ (nouvel itéré)
(4) Test d'arrêt via $J^{i+1} - J^i$ (par exemple)

Algorithm 1: Steepest Descent.

To save a product matrix-vector it is preferable to substitute at the stage (1), the update of the residue following

$$r^{i+1} = r^i - \alpha^i \mathbf{K}\mathbf{r}^i \tag{2.2-5}$$

However, so avoiding inopportune accumulations of round-offs, one periodically recomputes the residue with the initial formula (1).

2.2.3 Elements of theory

One shows that this algorithm converges, for any starting point u^0, at the speed[15]

$$\|e(u^{i+1})\|_K^2 = \omega^i \|e(u^i)\|_K^2$$
$$\text{avec } \omega^i := 1 - \frac{\|r^i\|^4}{\langle K^{-1} r^i, r^i \rangle \langle \mathbf{K}\mathbf{r}^i, r^i \rangle} \tag{2.2-6}$$

By developing the error on the basis of clean mode $(\lambda_j ; v_j)$ matrix $\mathbf{K}$

$$e(u^i) = \sum_j \xi_j v_j \tag{2.2-7}$$

the attenuation factor of the error in energy becomes

$$\omega^i = 1 - \frac{\left(\sum_j \xi_j^2 \lambda_j^2\right)^2}{\left(\sum_j \xi_j^2 \lambda_j^3\right)\left(\sum_j \xi_j^2 \lambda_j\right)} \tag{2.2-8}$$

In (2.2-8), the fact that components ξ_j are squared ensures the priority ousting of the dominant eigenvalues. One finds here one of the characteristics of the modal methods of type Krylov (Lanczos, Arnoldi cf [Boi01] §5/§6) which privilege the extreme clean modes. For this reason, Steepest Descent and the combined gradient known as “coarse” are compared to the classical iterative solveurs (Jacobi, Gauss-Seidel, SOR…) more “smooth” because eliminating without discrimination all the components with each iteration.
Finally, thanks to the inequality of Kantorovitch[16], one largely improves the legibility of the attenuation factor. At the end of *I* iterations, in the worst case, the decrease is expressed in the form

$$\|e(u^i)\|_K \leq \left(\frac{\eta(K) - 1}{\eta(K) + 1}\right)^i \|e(u^0)\|_K \tag{2.2-9}$$

It ensures linear convergence[17] process in an iteration count proportional to conditioning of the operator. Thus, to obtain

15 The definite positivity of the operator ensures us of the “kindly person” of the parameter ω^i. It is well an attenuation factor because $0 \leq \omega^i \leq 1$.

16 Some is K matrix SPD and u vector not no one: $1 \leq \frac{\langle \mathbf{K}\mathbf{u}, u \rangle \langle K^{-1} u, u \rangle}{\|u\|_2^4} \leq \frac{\left(\eta(K)^{1/2} + \eta(K)^{-1/2}\right)^2}{4}$).

$$\frac{\|e(u^i)\|_K}{\|e(u^0)\|_K} \leq \varepsilon \quad (\text{petit}) \tag{2.2-10}$$

one needs an iteration count about

$$i \approx \frac{\eta(K)}{4} \ln \frac{1}{\varepsilon} \tag{2.2-11}$$

A badly conditioned problem will thus slow down the convergence of the process, which one “visually” had already noted with the phenomenon of “narrow and tortuous valley”.

For better apprehending the implication of the spectrum of the operator and starting point in the unfolding of the algorithm, let us simplify the formula (2.2-8) while placing itself in the commonplace case where $N=2$. While noting $\kappa=\frac{\lambda_1}{\lambda_2}$ the matric conditioning of the operator and $\mu=\frac{\xi_2}{\xi_1}$ the slope of the error with $i^{\text{ème}}$ iteration (in the frame of reference of the two clean vectors), one obtains a more readable expression of the attenuation factor of the error (cf figure 2.2-2)

$$\omega^i = 1 - \frac{\left(\kappa^2+\mu^2\right)^2}{\left(\kappa+\mu^2\right)\left(\kappa^3+\mu^2\right)} \tag{2.2-12}$$

As for the modal solveurs, one notes that the importance of the conditioning of the operator is balanced by the choice of a good starting point: in spite of a bad conditioning, the cases (A) and (b) are very different; In the first, the starting point generates almost a clean space of the operator and one converges in two iterations, if they are not the “sempiternal zigzags”.

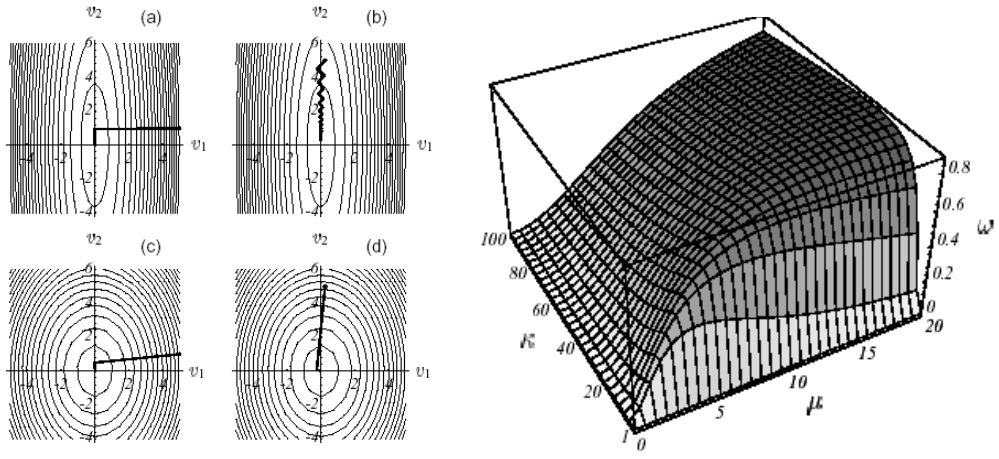


Figure 2.2-2. Convergence of Steepest Descent on the n°1 example according to the values of conditioning and the starting point: κ large and μ small (A), κ and μ large (b), κ and μ small (c), κ small and μ large (d) and total form of $\omega=\omega(\kappa,\mu)$ (E).

Note:

- *This method of the strongest slope was initiated by Cauchy (1847) and was given to the last style by Curry (1944). Within the framework of operators SPD, it is also sometimes called method of the gradient with optimal parameter (cf [LT98] §8.2.1). In spite of its weak properties of convergence, she knew her “heyday” to minimize functions J quasi-unspecified, makes some only derivable. In this case plus general, it then makes it possible to reach only stationary points, at best local minima.*

17 I.e. $\lim_{i\to\infty} \frac{J(u^{i+1})-J(u)}{J(u^i)-J(u)} \leq \alpha := \left(\frac{\eta(K)-1}{\eta(K)+1}\right)^2 < 1$. The asymptotic rate of convergence α report of Kantorovitch is called

- *To avoid zigzag the proceeded various effects of acceleration were proposed (cf Forsythe 1968, Luenberger 1973 or [Min08] §2.3) which have a speed of convergence similar to that of the gradient combined but for a quite higher calculation complexity. They thus fell gradually in disuse.*
- *Let us note that alternatives of algorithm 1 were introduced to treat cases not SPD: Iteration of the minimum residue and Steepest Descent with standard of the residue (cf [Saa03]).*
- *One finds through Steepest Descent a very widespread key concept in digital analysis: that of the resolution of a problem, by projection of reiterated on an approximating subspace, here* $K_i := vect(r^i)$ *, perpendicular to another subspace, here* $L_i := K_i$ *. They are called, respectively, spaces of research or approximation and space of constraint. For Steepest Descent, they are equal and reduced to their more simple expression but we will see that for the combined gradient they take the shape of particular spaces, known as of Krylov.*

Formally, with each iteration i *algorithm, one seeks an increment thus* $\delta^i := \alpha^i r^i$ *such as:*

$$\begin{cases} u^i := u^0 + \delta^i \quad \delta^i := \alpha^i r^i \in K_i \\ \langle r^0 - \boldsymbol{K}\,\delta^i, w \rangle = 0 \quad \forall\, w \in L_i = K_i \end{cases} \tag{2.2-13}$$

This framework general constitutes the conditions of Petrov-Galerkin what is called.

2.2.4 Complexity and occupation memory

The major part of the cost calculation of algorithm 1 lies in the update (2.2-5) and, more particularly, in the product matrix-vector $K r^i$. In addition, it was already mentioned that its convergence was acquired and took place in an iteration count proportional to the conditioning of the matrix (cf (2.2-11)).
Calculation complexity of the algorithm is thus, if one takes account of the hollow character of the operator, about $\Theta(\eta(K)\, cN)$ where c is the median number of nonworthless terms per line of K.
Thus, with problems resulting from the discretization finite elements elliptic operators of the second order[18] (resp. fourth order), one has conditionings of operators in $\eta(K) = \Theta(N^{2/d})$ (resp. $\eta(K) = \Theta(N^{4/d})$) where d the dimension of space. Calculation complexity of Steepest Descent within this framework is written thus $\Theta\left(cN^{\frac{2}{d}+1}\right)$ (resp. $\Theta\left(cN^{\frac{4}{d}+1}\right)$).
As regards the occupation memory, only the storage of the matrix of work is possibly necessary[19]: $\Theta(cN)$. In practice, the data-processing installation of hollow storage imposes the management of additional vectors of entireties: for example, for storage MORSE used in *Code_Aster*, vectors of the indices of end of line and the indices of columns of the nonworthless elements of the profile. From where effective a memory complexity of $\Theta(cN)$ realities and $\Theta(cN+N)$ entireties.

2.3 Method of "general" descent

2.3.1 Principle

A crucial stage of these methods is the choice of their direction of descent[20]. Even if the gradient of the functional calculus remains the principal ingredient about it, one can choose another version completely of it d^i that necessary for Steepest Descent. With *i*ème iteration, one will thus seek to build u^{i+1} checking

$$u^{i+1} := u^i + \alpha^i d^i \tag{2.3-1}$$

with

18 Case generally encountered in mechanics of the structures.
19 In the absolute, Steepest Descent as the GC requires only the knowledge of the action of the matrix on an unspecified vector and not its storage *in extenso*. This facility can prove to be invaluable very for applications rebuke in degrees of freedom (CFD).
20 By definition a direction of descent (dd) with *i*ème stage, noted d^i, checks: $\left(\nabla J^i\right)^T d^i < 0$.

$$\alpha^i = \underset{\alpha\in[\alpha_m,\alpha_M]}{Arg\min}\, J\left(u^i+\alpha\, d^i\right) \tag{2.3-2}$$

It is of course only one generalization of 'Steepest Descent' seen previously and it is shown that its choices of the parameter of descent and its property of orthogonality spread

$$\alpha^i := \frac{\langle r^i, d^i\rangle}{\langle d^i, \mathbf{Kd^i}\rangle} \qquad \langle d^i, r^{i+1}\rangle = 0 \tag{2.3-3}$$

From where the same effect "zigzag" during unfolding of the process and a convergence similar to (2.2-6) with an attenuation factor of the error undervalued by:

$$\omega^i := 1-\frac{\langle r^i, d^i\rangle^2}{\langle K^{-1} r^i, r^i\rangle\langle \mathbf{Kd^i}, d^i\rangle} \geq \frac{1}{\eta(K)}\left\langle \frac{r^i}{\|r^i\|}, \frac{d^i}{\|d^i\|}\right\rangle^2 \tag{2.3-4}$$

This result one can then formulate two reports:

- The conditioning of the operator intervenes directly on the attenuation factor and thus on the speed of convergence,
- To make sure of convergence (sufficient condition) one needs, at the time of a given iteration, that the direction of descent is not orthogonal with the residue.

The sufficient condition of this last item is of course checked for Steepest Descent ($d^i = r^i$) and it will impose a choice of direction of descent for the combined gradient. To mitigate the issue raised by the first point, we will see that it is possible to constitute an operator of work $\tilde{K}$ whose conditioning is less. One speaks then about pre-packaging.
Always in the case of an operator SPD, the course of a method of "general" descent is thus written

Initialisation u^0, d^0 donnés, $r^0 = f - \mathbf{Ku^0}$
Boucle en i
(1) $z^i = \mathbf{Kd^i}$
(2) $\alpha^i = \dfrac{\langle r^i, d^i\rangle}{\langle d^i, z^i\rangle}$ (paramètre optimal de descente)
(3) $u^{i+1} = u^i + \alpha^i d^i$ (nouvel itéré)
(4) $r^{i+1} = r^i - \alpha^i z^i$ (nouvel résidu)
(5) Test d'arrêt via $J^{i+1} - J^i$, $\|d^i\|$ ou $\|r^{i+1}\|$ (par exemple)
(6) Calcul de la dd $d^{i+1} = d^{i+1}\left(\nabla J^k, \nabla^2 J^k, d^k \ldots\right)$

Algorithm 2: Method of descent in the case of a quadratic functional calculus.

This algorithm precedes already well that of the Gradient Combined (GC) which we will examine in the following chapter (cf algorithm 4). It shows well that the GC is not that a method of descent applied within the framework of quadratic functional calculuses and specific directions of descent. Finally, only the stage (6) will be some packed.

Note:

- *By successively posing like directions of descent the canonical vectors of the axes of coordinates of space with N dimensions ($d^i = e_i$), one obtains the method of Gauss-Seidel.*
- *To avoid the overcost calculation of the stage of unidimensional minimization (2) (produced matrix-vector) one can choose to fix the parameter of descent arbitrarily: it is the method of Richardson who converges as well as possible like Steepest Descent.*

2.3.2 Complements

With a functional calculus J continue unspecified (cf figure 2.3-1), one exceeds the strict framework of the linear inversion of system for that of unconstrained nonlinear continuous optimization (J then is often called function objective cost or function). Two simplifications which had runs up to now become illicit then:

- The reactualization of the residue : stage (4),
- The simplified calculation of the optimal parameter of descent : stage (2).

Their reasons to be being only to use all the assumptions of the problem to facilitate unidimensional minimization (2.3-2), one is then constrained to explicitly carry out this linear research on a functional calculus more kicked up a rumpus with this time of multiple local minima. Fortunately, there exists a whole panoply of methods according to the degree of necessary information on the function cost J :

- J (quadratic interpolation, dichotomy on the values of the function, of the gilded section, rule of Goldstein and Price…),
- J , ∇J (dichotomy on the values of the derivative, regulates of Wolfe, Armijo…),
- (J , ∇J , $\nabla^2 J$ Newton-Raphson…)
- …

As regards the direction of descent, there too of many solutions are proposed in the literature (combined gradient nonlinear, quasi-Newton, Newton, Levenberg-Marquardt[21]…). Of long time, methods known as of nonlinear combined gradient (Fletcher-Reeves (FR) 1964 and Polak-Ribière (PR) 1971) proved to be interesting: they converge superlinéairement towards a local minimum at a reduced cost calculation and obstruction memory (cf Figures 2.3-1).

They lead to the choice of an additional parameter β^i who manages the linear combination between the directions of descent

$$d^{i+1} = -\nabla J^{i+1} + \beta^{i+1} d^i \quad \text{avec } \beta^{i+1} := \begin{cases} \dfrac{\|\nabla J^{i+1}\|^2}{\|\nabla J^i\|^2} & (\text{FR}) \\ \dfrac{\langle \nabla J^{i+1}, \nabla J^{i+1} - \nabla J^i \rangle}{\|\nabla J^i\|^2} & (\text{PR}) \end{cases} \tag{2.3-5}$$

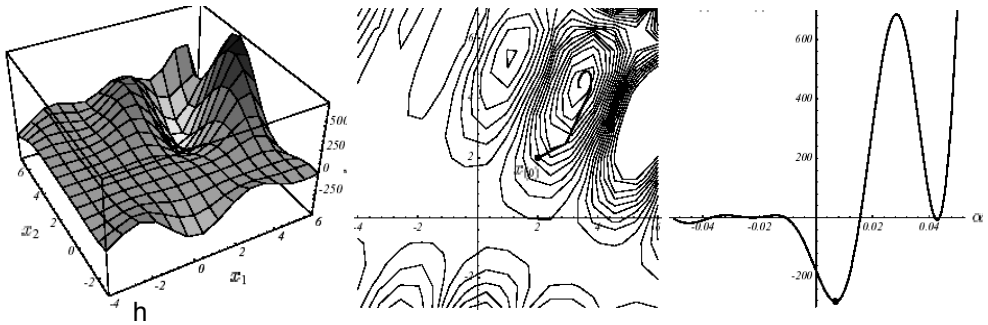


Figure 2.3-1. Example of J nonconvex in $N=2$: graph (A), convergence with a nonlinear GC of type Fletcher-Reeves (b), plan of cut the first unidimensional research (c).

From where the algorithm, while naming r^0 opposite of the gradient and either the residue (which does not take any more place to be here),

21 These two last methods are used by *Code_Aster* : the first in the nonlinear operators (STAT/DYNA/THER_NON_LINE), the second in the macro one of retiming (MACR_RECAL).

Initialisation u^0 donné, $r^0=-\nabla J^0$, $d^0=r^0$
Boucle en i

(1) Trouver $\alpha^i=Arg\min_{\alpha\in[\alpha_m,\alpha_M]} J\left(u^i+\alpha\, d^i\right)$ (paramètre de descente)

(2) $u^{i+1}=u^i+\alpha^i d^i$ (nouvel itéré)

(3) $r^{i+1}=-\nabla J^{i+1}$ (nouveau gradient)

(4) Test d'arrêt via $J^{i+1}-J^i$, $\|d^i\|$ ou $\|r^{i+1}\|$ (par exemple)

(5) $\beta^{i+1}=\begin{cases}\dfrac{\|r^{i+1}\|^2}{\|r^i\|^2} & (\mathrm{FR})\\ \dfrac{\langle r^{i+1}, r^{i+1}-r^i\rangle}{\|r^i\|^2} & (\mathrm{PR})\end{cases}$ (paramètre de conjugaison)

(6) $d^{i+1}=r^{i+1}+\beta^{i+1}d^i$ (nouvelle dd)

Algorithm 3: Methods of the nonlinear combined gradient (FR and PR).

Nonlinear designation combined gradient is rather here synonymous with “nonconvex”: there is no more dependence between the problem of minimization (P_2) and a linear system (P_1). With the sight of algorithm 4, the great similarities with the GC consequently appear completely clear. Within the framework of a quadratic function cost of type (2.1-2b), it is just enough to substitute at the stage (1) the actualization of the residue and the calculation of the optimal parameter of descent. The GC is only one method of Fletcher - Reeves steady to the case of a convex quadratic functional calculus.

Now that we started well the link between the methods of descent, the gradient combined with the linear direction solvor SPD and his version, sight of the end of the nonlinear spyglass continuous optimization, we (finally!) will pass to sharp from the subject and will argue the choice of a direction of descent of the type (2.3-5) for the standard GC. For more information on the methods of type descent, the reader will be able to refer to the excellent works of optimization (in French language) of Mr. Minoux [Min08], J.C. Culioli [Cu94] or J.F.Bonnans et al. [BGLS].

3 The Combined Gradient (GC)

3.1 General description

3.1.1 Principle

Now that the bases are set up we will be able to approach the algorithm of the Combined Gradient (GC) itself. It belongs to a subset of methods of descent which gathers the methods known as “of combined directions”[22]. Those recommend to build directions of descents gradually $d^0, d^1, d^2 \ldots$ linearly independent so as to avoid the zigzags of the method of descent classical.

Which linear combination then to recommend to build, at the stage i, new direction of descent? Knowing of course that it must take account of two crucial information: the value of the gradient $\nabla J^i = -r^i$ and those of the preceding directions $d^0, d^1 \ldots d^{i-1}$.

$$? \quad d^i = \alpha\, r^i + \sum_{j<i} \beta_j\, d^j \tag{3.1-1}$$

The trick consists in choosing a vectorial independence of type K - orthogonality (as the operator of work is SPD, it defines a scalar product well *via* which two vectors can be orthogonal, cf appears 3.1-1)

$$\left(d^i\right)^T K\, d^j = 0 \quad i \neq j \tag{3.1-2}$$

also called conjugation, from where the designation of the algorithm. It makes it possible to propagate orthogonality gradually and thus to calculate only one coefficient of Gram-Schmidt to each iteration. From where a profit in calculation complexity and very appreciable memory.

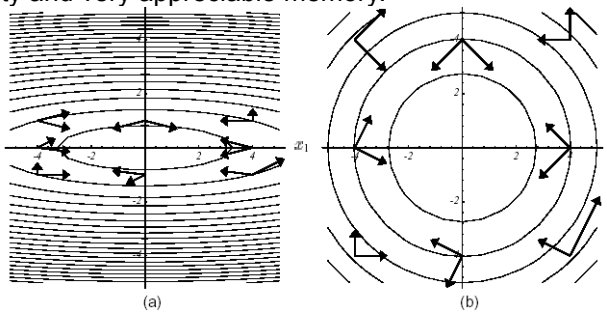


Figure 3.1-1. Example of pairs of vectors K - orthogonal in 2D: conditioning of K unspecified (A), perfect conditioning (i.e equal to the unit) = usual orthogonality (b).

One can thus be satisfied with a linear combination of the type

$$d^i := r^i + \beta^i d^{i-1} \tag{3.1-3}$$

And this, more especially as it checks the sufficient condition (2.3-4) and that because of orthogonality (2.3-3b), the unidimensional research (2.3-2) which follows takes place in an optimal space: the plan formed by the two orthogonal directions $\left(r^i, d^{i-1}\right)$.

It thus remains to determine the optimal value of the proportionality factor β^i. In the GC this choice takes place so as to maximize the attenuation factor of (2.3-4), i.e. the term

$$\frac{\langle \mathrm{r}^i, \mathrm{d}^i \rangle^2}{\langle \mathrm{K}^{-1} \mathrm{r}^i, \mathrm{r}^i \rangle \langle \mathrm{K} d^i, \mathrm{d}^i \rangle} \tag{3.1-4}$$

It leads to the expression

$$\beta^i := \frac{\|r^i\|^2}{\|r^{i-1}\|^2} \tag{3.1-5}$$

and the same property of orthogonality of the successive residues as for Steepest Descent induces (but without the zigzags!)

$$\langle r^i, r^{i-1} \rangle = 0 \tag{3.1-6}$$

A condition “residue-dd is added”

22 The methods of Fletcher-Reeves and Polak-Ribière (cf §2.3) are also part of this family of methods.

$$\langle r^i, d^i \rangle = \|r^i\|^2 \tag{3.1-7}$$

who forces to initialize the process *via* $d^0 = r^0$.

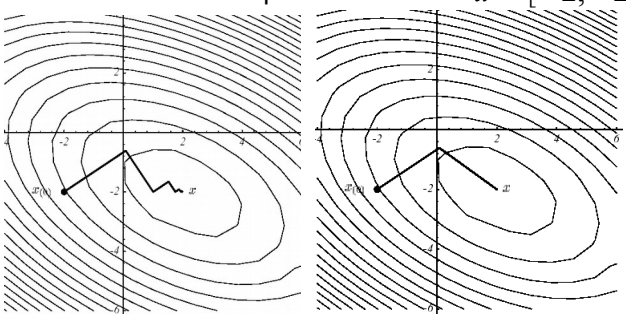
3.1.2 Algorithm

In short, by recapitulating the relations (2.2-5), (2.3-1), (2.3-3) and (3.1-3), (3.1-5), (3.1-7) it occurs the classical algorithm

Initialisation u^0 donné, $r^0 = \mathbf{f\text{-}Ku^0}$, $d^0 = r^0$
Boucle en i
(1) $z^i = \mathbf{Kd^i}$
(2) $\alpha^i = \dfrac{\|r^i\|^2}{\langle d^i, z^i \rangle}$ (paramètre optimal de descente)
(3) $u^{i+1} = u^i + \alpha^i d^i$ (nouvel itéré)
(4) $r^{i+1} = r^i - \alpha^i z^i$ (nouveau résidu)
(5) Test d'arrêt via $\langle r^{i+1}, r^{i+1} \rangle$ (par exemple)
(6) $\beta^{i+1} = \dfrac{\|r^{i+1}\|^2}{\|r^i\|^2}$ (paramètre de conjugaison optimal)
(7) $d^{i+1} = r^{i+1} + \beta^{i+1} d^i$ (nouvelle dd)

Algorithm 4: Standard combined gradient (GC).

On the n°1 example, the "supremacy" of the GC compared to Steepest Descent is manifesto (cf figure 3.1-2) and is explained by the difference between the result of convergence (2.2-9) and that of (3.2-9). In both cases, the same starting points and the same criteria of stop were selected: $u^0 = [-2, -2]^T$ and $\|r^i\|^2 < \varepsilon = 10^{-6}$.



has B

Figure 3.1-2. Convergences compared, on the n°1 example, of Steepest Descent (A) and GC (b).

Note:

- *This method of the GC was developed independently by M.R.Hestenes (1951) and E.Stiefel (1952) of the 'National Office of Standard' of Washington D.C. (seedbed of numericians with also C.Lanczos). The first theoretical results of convergence are due to work of S.Kaniel (1966) and of H.A.Van der Vorst (1986) and it was really popularized for the resolution of large hollow systems by J.K.Reid (1971). The interested reader will find a history commented on and an exhaustive bibliography on the subject in papers of G.H.Golub, H.A Van der Vorst and Y.Saad [Go89] [GV97] [SV00].*
- *For the little story, extremely of its very broad diffusion in the industrial world and academic, and, of his many alternatives, the GC was classified in third position of the "Top10" of the best algorithms of the XX ^{E} century [Ci00]. Just behind the methods of Monte Carlo and the simplex but in front of the algorithm **QR** , transforms of Fourier and the solveurs multipôles!*
- *One finds traces in the codes of EDF R & D of the GC as of the beginning of the Eighties with the first work on the subject of J.P.Grégoire. Since it was spread, with an unequal happiness, in many fields,*

_N3S, Code_Saturne, LADYBIRD, Code_Aster, TRIFOU, ESTET, TELEMAC _, and it much was optimized, vectorized, was paralleled [Greg]…

- *With the test of stop on the standard of the residue, theoretically licit but in practice sometimes difficult to gauge, one often prefers a criterion of adimensional stop, such as the residue relative to $I^{ème}$ iteration: $\delta^i := \frac{\|r^i\|}{\|f\|}$ (cf §5).*

3.2 Elements of theory

3.2.1 Space of Krylov

By carrying out a second analysis of Petrov-Galerkin already evoked for Steepest Descent, one can synthesize in a sentence the action of the GC. It carries out successive orthogonal projections on the space of Krylov $\kappa_i(K, r^0) := vect(r^0, Kr^0 \dots K^{i-1} r^0)$ where r^0 is the initial residue: with $I^{ème}$ iteration $K_i = L_i = \kappa_i(K, r^0)$. One solves the linear system (P_1) by seeking an approximate solution u^i in the subspace refines (space of search for dimension N)

$$A = r^0 + \kappa_i(K, r^0) \tag{3.2-1}$$

while imposing the constraint of orthogonality (space of the constraints of dimension N)

$$r^i := f - \mathbf{K}\mathbf{u}^i \perp \kappa_i(K, r^0) \tag{3.2-2}$$

This space of Krylov has the good property to facilitate the approximation of the solution, at the end of m iterations, in the form

$$K^{-1} f \approx u^m = r^0 + P_{m-1}(K) f \tag{3.2-3}$$

where P_{m-1} is a certain matric polynomial of order $m-1$. Indeed, it is shown that the residues and the directions of descent generate this space

$$\begin{aligned} vect(r^0, r^1 \dots r^{m-1}) &= \kappa_m(K, r^0) \\ vect(d^0, d^1 \dots d^{m-1}) &= \kappa_m(K, r^0) \end{aligned} \tag{3.2-4}$$

while allowing the approached solution, u^m , to minimize the standard in energy on all space closely connected *With*

$$\|u^m\|_K \le \|u\|_K \quad \forall u \in A \tag{3.2-5}$$

This joint result with the property (3.2-4b) illustrates all the optimality of the GC: contrary to the methods of descent, the minimum of energy is not carried out successively for each direction of descent, but jointly for all the directions of descent already obtained.

Note:

- *One distinguishes a large variety from methods of projection on spaces of the Krylov type, called more prosaically "methods of Krylov". To solve linear systems [Saa03] (GC, GMRES, FOM/IOM/DOM, GCR, ORTHODIR/MIN…) and/or of the modal problems [Boi01] (Lanczos, Arnoldi…). They differ by the choice from their space from constraint and by that from the pre-packaging applied to the initial operator to constitute that of work, knowing that different establishments lead to radically distinct algorithms (vectorial version or per blocks, tools of orthonormalisation…).*

3.2.2 Orthogonality

As one already announced, the directions of descents are K - orthogonal between them. Moreover, the choice of the optimal parameter of descent (cf (2.2-4), (2.3-3a) or stage (2)) impose, gradually, orthogonalities

$$\begin{aligned}&\langle d^i, r^m\rangle = 0 \quad \forall i < m\\ &\langle r^i, r^m\rangle = 0\end{aligned} \tag{3.2-6}$$

One thus notes a small approximation in the name of the GC, because the gradients are not combined (cf (3.2-6b)) and the directions combined do not comprise only gradients (cf (3.1-3)). But “let us not haggle” not, the indicated ingredients are nevertheless there!

At the conclusion of N iterations, two cases arise:

- That is to say the residue is null $r^N = 0$ thus convergence.
- Either it is orthogonal with N preceding directions of descent which constitute a base of the finished space of approximation $\Re^N$ (as they are linearly independent). From where necessarily $r^N = 0$ thus convergence.

It would thus seem that the GC is a direct method which converges in at the maximum N iterations, it is at least what one believed before testing it on practical cases! Because what remains true in theory, into arithmetic exact, is put at evil by arithmetic finished calculators. Gradually, in particular because of rounding errors, the directions of descent lose their beautiful properties of conjugation and minimization leaves necessary space.

Known as differently, one solves an approximate problem which is not completely any more the desired projection of the initial problem. The method (theoretically) direct revealed its true nature! It iterative and is thus subjected, in practice, with many risks (conditioning, starting point, tests of stop, precision of orthogonality…).

To cure it, one can force during the construction of the new direction of descent (cf algorithm 4, stage (7)), a phase of reorthogonalisation. This very widespread practice in modal analysis [Boi01] and appendix 2) and in decomposition of field [Boi08] can decline itself under various alternatives: total, partial, selective reorthogonalisation… *via* a whole panoply of procedures of orthogonalisation (GS, GSM, IGSM, Househölder, Givens).

These réorthogonalisations require, for example, the storage of N_{orth} first vectors $d^k \left(k = 1 \ldots N_{orth}\right)$ and of their product by the operator of work $z^k = Kd^k \left(k = 1 \ldots N_{orth}\right)$. Formally, it is thus a question of substituting at the two last stages of the algorithm following calculation

$$d^{i+1} = r^{i+1} - \sum_{k=0}^{\max\left(i, N_{orth}\right)} \frac{\langle r^{i+1}, \mathbf{Kd^k}\rangle}{\langle d^k, \mathbf{Kd^k}\rangle} d^k \quad (\text{nouvelle dd } K-\text{orthogonalisée}) \tag{3.2-7}$$

Through the various digital experiments which were undertaken (cf in particular work of J.P.Grégoire and [Boi03]), it seems that this reorthogonalisation is not always effective. Its overcost due mainly to the new products matrix-vector $K\,d^k$ (and with their storage) is not always compensated by the profit of iteration count total.

3.2.3 Convergence

Because of particular structure of the space of approximation (3.2-3) and property of minimization on this space of the approximate solution u^m (cf (3.2-5)), one obtains an estimate the speed of convergence of the GC

$$\begin{aligned}&\|e\left(u^i\right)\|_K^2 = \left(\omega^i\right)^2 \|e\left(u^0\right)\|_K^2\\ &\text{avec } \omega^i := \max_{1 \le i \le N}\left(1 - \lambda_i P_{m-1}\left(\lambda_i\right)\right)\end{aligned} \tag{3.2-8}$$

where one notes $(\lambda_i ; v_i)$ clean modes of the matrix K and P_{m-1} an unspecified polynomial of degree at the maximum $m-1$. Famous polynomials of Tchebycheff, *via* their good properties of increase within the space of polynomials, make it possible to improve the legibility of this attenuation factor ω^i. At the end of i iterations, in the worst case, the decrease is expressed then in the form

$$\|e(u^i)\|_K \leq 2\left(\frac{\sqrt{\eta(K)}-1}{\sqrt{\eta(K)}+1}\right)^i \|e(u^0)\|_K \quad (3.2\text{-}9)$$

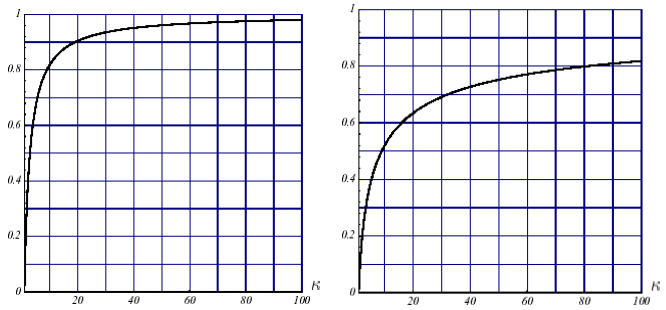
It ensures convergence superlinéaire (i.e. $\lim_{i\to\infty}\frac{J(u^{i+1})-J(u)}{J(u^i)-J(u)}=0$) process in an iteration count proportional to the square root of the conditioning of the operator. Thus, to obtain

$$\frac{\|e(u^i)\|_K}{\|e(u^0)\|_K} \leq \varepsilon \quad (\text{petit}) \quad (3.2\text{-}10)$$

one needs an iteration count about

$$i \approx \frac{\sqrt{\eta(K)}}{2}\ln\frac{2}{\varepsilon} \quad (3.2\text{-}11)$$

Obviously as we noticed many times, a badly conditioned problem will slow down the convergence of the process. But this deceleration will be less notable for the GC than for Steepest Descent (cf figures 3.2-1). And in any event, the total convergence of the first will be better.



(A) (b)

Figure 3.2-1. Compared convergences of Steepest Descent (A) and of the GC (b) (with the factor ½ near) according to conditioning κ .

Note:

- *In practice, benefitting from particular circumstances, _better starting point and/or advantageous spectral distribution _, the convergence of the GC can be much better than than lets hope (3.2-9). The methods of Krylov tending to flush out the extreme eigenvalues firstly, the "effective conditioning" of the operator of work is some improved.*
- *On the other hand, certain iterations of Steepest Descent can get a better decrease of the residue than the same iterations of the GC. Thus, the first iteration of the GC is identical to that of Steepest-Descent and thus with an equal rate of convergence.*

3.2.4 Complexity and occupation memory

As for Steepest Descent, the major part of the cost calculation of this algorithm lies in the stage (1), the product matrix-vector. Its complexity is about $\Theta(kcN)$ where c is the median number of nonworthless terms per line of K and k the iteration count necessary to convergence. To be much more effective than simple Cholesky (of complexity $\Theta\left(\frac{N^3}{3}\right)$) it is thus necessary:

- To take well into account the hollow character of the matrices resulting from the discretizations finite elements (storage MORSE, product optimized matrix-vector *ad hoc*): $c<<N$.
- To prepack the operator of work: $k<<N$.

One already pointed out that for an operator SPD his theoretical convergence occurs in, at most, N iterations and proportionally with the root of conditioning (cf (3.2-11)). In practice, for large badly conditioned systems (as it is often the case in mechanics of the structures), it can be very slow to appear (cf phenomenon of Lanczos of the following paragraph).

Taking into account as of conditionings operators generally noted in mechanics of the structures and recalled for the study of complexity of Steepest Descent, which one takes again the notations, calculation complexity of the GC is written $\Theta\left(cN^{\frac{1}{d}+1} \right)$ (resp. $\Theta\left(cN^{\frac{2}{d}+1} \right)$).

As regards the occupation memory, as for Steepest Descent, only the storage of the matrix of work is possibly necessary: $\Theta(cN)$. In practice, the data-processing installation of hollow storage imposes the management of additional vectors of entireties: for example for storage MORSE used in *Code_Aster*, vectors of the indices of end of line and the indices of columns of the elements of the profile. From where effective a memory complexity of $\Theta(cN)$ realities and $\Theta(cN+N)$ entireties.

Note:

- *These considerations on the obstruction memory do not take into account the problems of storage of a possible preconditionnor and workspace which its construction can temporarily mobilize.*

3.3 Complements

3.3.1 Equivalence with the method of Lanczos

In "triturating" the property of orthogonality of the residue with any vector of the space of Krylov (cf (3.2-2)), it is shown that them m iterations of the GC lead to the construction of one factorized of Lanczos of the same order (cf [Boi01] §5.2)

$$KQ^m = Q^m T^m - \underbrace{\frac{\sqrt{\beta^m}}{\alpha^{m-1}} q^m}_{R^m} e_m^T \qquad (3.3\text{-}1)$$

while noting:

- R^m the specific "residue" of this factorization,
- e_m $m^{\text{ième}}$ vector of the canonical base,
- $q^i := \frac{r^i}{\|r^i\|}$ the vectors residues standardized with the unit, called for the occasion vectors of Lanczos,
- $Q^m := \left[\frac{r^0}{\|r^0\|} \quad \dots \quad \frac{r^{m-1}}{\|r^{m-1}\|} \right]$ the orthogonal matrix which they constitute.

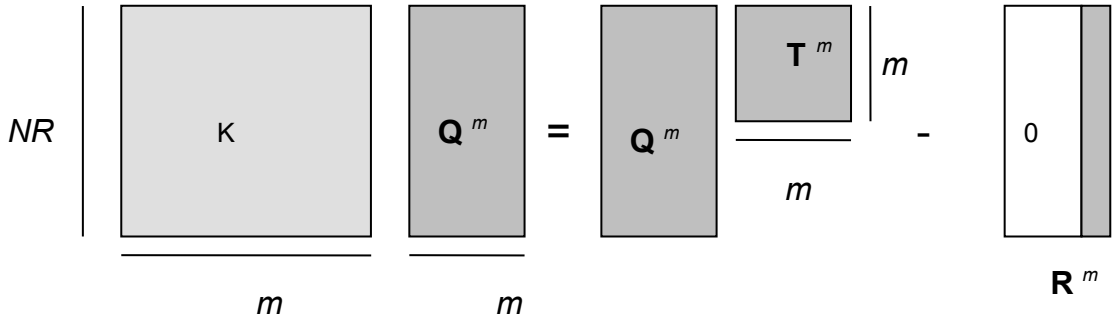


Figure 3.3-1. Factorization of Lanczos induced by the GC.

The matrix of Rayleigh which expresses the orthogonal projection of K on the space of Krylov $\kappa_m\left(K, r^0\right)$ takes the canonical shape then

$$T^m=\begin{bmatrix} \frac{1}{\alpha^0} & -\frac{\sqrt{\beta^1}}{\alpha^0} & 0 & 0 \\ -\frac{\sqrt{\beta^1}}{\alpha^0} & \frac{1}{\alpha^1}+\frac{\beta^1}{\alpha^0} & \cdots & 0 \\ 0 & \cdots & \cdots & -\frac{\sqrt{\beta^{m-1}}}{\alpha^{m-2}} \\ 0 & 0 & -\frac{\sqrt{\beta^{m-1}}}{\alpha^{m-2}} & \frac{1}{\alpha^{m-1}}+\frac{\beta^{m-1}}{\alpha^{m-2}} \end{bmatrix} \quad (3.3\text{-}2)$$

Almost without additional calculation, the GC thus provides the approximation of Rayleigh of the operator of work in a nice form, _symmetrical square matrix tridiagonale of flexible size _, of which it will be easy to deduce the spectrum (*via*, for example, robust **QR**, cf [Boi01] appendix 2)).

At the conclusion of a linear inversion of system led by a simple GC one can thus, with less expenses, to know, besides the sought solution, an approximation of the spectrum of the matrix and thus of his conditioning [Greg]. This functionality could be inserted in *Code_Aster* so as to better control the level of pre-packaging.

Note:

• *The operator of work being SPD, his matrix of Rayleigh inherits this property and one obtains his decomposition directly* $T^m=L^m D^m\left(L^m\right)^T$ *with*

$$L^m:=\begin{bmatrix} 1 & 0 & 0 & 0 \\ -\sqrt{\beta^1} & 1 & \cdots & 0 \\ 0 & \cdots & \cdots & 0 \\ 0 & 0 & -\sqrt{\beta^{m-1}} & 1 \end{bmatrix} \text{ et } D^m:=\begin{bmatrix} \frac{1}{\alpha^0} & 0 & 0 & 0 \\ 0 & \frac{1}{\alpha^1} & \cdots & 0 \\ 0 & \cdots & \cdots & 0 \\ 0 & 0 & 0 & \frac{1}{\alpha^{m-1}} \end{bmatrix} \quad (3.3\text{-}3)$$

Method of Lanczos not calculating these intermediate terms α^i *and* β^i*, but directly the tridiagonaux terms, she does not have access to this information. On the other hand, it can take into account positive symmetrical matrices not necessarily definite.*

• *Therefore, that it is a question of reversing a linear system or of determining part of its spectrum, these methods of Krylov function on the same principle: to determine gradually the vectors of bases at the same time generating the space of projection as the result of this projection. In the GC as in Lanczos, one strives to make this projection most robust (orthogonality of the basic vectors), most reduced (size of projection = iteration count) and simplest possible (orthogonal projection). For the modal solvor, another constraint is juxtaposed: to approximate the best possible spectrum of the operator of work by that of the matrix of Rayleigh. In nonsymmetrical, one speaks about either orthogonal projection but oblique and the bringings together is carried out then between GMRES and Arnoldi.*

• *Into arithmetic exact, at the end of* N *iterations, all the spectrum of the operator was completely determined. In practice, the problems of round-offs and orthogonalisation prevent us. But, if one is sufficiently patient (* $m \gg N$*), one is nevertheless able to detect all the spectrum: it is what is called the phenomenon of Lanczos (cf [Boi01] § 5.3.1/5.3.3).*

• *Paradoxically the loss of orthogonality is especially ascribable more with the convergence of a clean mode than to the rounding errors! This analysis due to CC.Paige (1980) attests that as soon as a clean*

mode is captured it disturbs the orthogonal fitting of the vectors of Lanczos (in the case which worries us, the residues and the directions of descent). Into modal, the digital processing of this parasitic phenomenon was the object of many palliative developments. As regards the GC, it must attach the effectiveness of the methods of reorthogonalisation of the directions of descent evoked previously.

3.3.2 Encased Solveurs

As one already pointed out, the linear solveurs are often hidden with deepest of other algorithms, and in particular, of the nonlinear solveurs of Newton type. It is for *Code_Aster* cases of application most frequently used: operators STAT_NON_LINE, DYNA_NON_LINE and THER_NON_LINE.
In such a configuration, a linear solvor such as the CG can draw his pin from the game (compared to a direct solvor). Its iterative character proves to be useful to make evolve the test of stop of the algorithm (cf stage (5) algorithm 4) according to the relative error of the nonlinear solvor who encapsulates it: it is the problems encased solveurs (one also speaks about relieving) [CM01] [GY97] [BFG] of which the CERFACS is “spearhead” in France.

By slackening at the convenient period the necessary precision on the cancellation of the residue (for example strategy in the case modal method of Krylov type + GC) or, on the contrary by hardening it (resp. modal method of standard power + Gauss-Seidel + GC), one can thus gain in calculation complexity on the GC without modifying the convergence of the total process. It is of course necessary to develop simple and inexpensive criteria so that the profit is substantial. This functionality could to be inserted in *Code_Aster* (internal and automatic piloting of the value RESI_RELA).

Note:

- Certain authors also put the question about sequence of these algorithms: “Newton GC” or “GC – Newton”? So from a data-processing and conceptual point of view, the question is quickly clear-cut: one prefers the first solution, more readable, more flexible and which does not disturb existing it, from a digital and algorithmic point of view, the division is more moderate. That can depend on the “technical tripailles” deployed by the non-linear solvor (tangent matrix, unidimensional minimization…) and of the deployed GC (preconditionnor). Nevertheless, it seems that the natural order, “Newton GC”, that is to say most effective and most evolutionary.

3.3.3 Parallelism

The GC like much of iterative solveurs lends itself well to parallelism. It is enough scalable[23]. The principal elementary operations which constitutes it (produced matrix-vector, operation standard[24] “daxpy and produced scalar) breaks up effectively between various processors, only the parallelization of the preconditionnor (often based on a direct solvor) can prove to be difficult (problems often called: ‘parallel preconditioning’). Moreover bookstores of linear algebras (PETSc, HYPRE, TRILINOS…) always to a parallel version their préconditionneurs do not propose. Concerning the préconditionneurs, there are thus two criteria which tend to be contradicted: parallel effectiveness and that algorithmic.

The codes thus tend to privilege not very effective préconditionneurs from an algorithmic point of view[25] (type Jacobi) but not very greedy in memory and very parallélisables. The parallel profit (called speed-up) is immediately very interesting, but the sequential point of comparison can not be very glorious[26] !

23 Or French extensibility. A method is known as scalable from a strong point of view (it is the criterion by default, ‘strong scalability’), if it makes it possible to all the more quickly solve a problem one proportionally adds of the processors. Less commonly, one speaks about weak scalability (‘weak scalability’), when it is a question of solving, in a comparable time, a larger problem while increasing by as much the number of processors (cf [Boi08] §2.4).

24 To take again a terminology specific to library BLAS. An operation of the type DAXPY fact of the linear combinations of vectors.

25 Going until tolerating convergence in thousands of iterations.

26 In any rigour, to express a parallel profit, one should take as reference best sequential calculation. This last which calls upon a more effective and possibly sequential preconditionnor. The exercise raises however a difficulty: this point of operation is often hard to display. The size of the studies, taking into account the

Another difficulty of parallelism also comes from the distribution of the data. It should be taken care that the various rules which govern the setting in data of calculation (attribution of the zones of limiting conditions, assignment of materials, choice of the finite elements…) by the fact that each processor knowing them is affected only partially. It is what is called the “consistency” of the distributed data [Boi08b].

In the same way, it is necessary to envisage in the code the communications *ad hoc* during each total handling of data (criteria of stop, postprocessings…).

Note:

- *Thus on the CRAY machine, J.P.Grégoire [Greg] adapted and paralleled, in shared memory and distributed memory, an algorithm of type prepacked combined gradient. Four types of operations are carried out jointly: elementary vectorial operations, scalar products, the products matrix-vector and the construction of the preconditionnor (diagonal). Problems of ‘parallel preconditioning’ which were solved only for the diagonal preconditionnor, unfortunately did not allow the bearing of these models in the official version of Code_Aster.*
- *The direct solveurs and the other iterative solveurs (stationary methods Gauss-Seidel, SSOR.) are famous less scalables that the methods of Krylov (GC, GMRES…).*

We besides now will tackle the thorny question of the pre-packaging for the GC. It will then become the Prepacked Combined Gradient: GCPC.

characteristics hardware of the machines, require from the start a calculation distributed on tens of processors!

4 The Prepacked Combined Gradient (GCPC)

4.1 General description

4.1.1 Principle

As one could note it (and hammer it!) in the preceding paragraphs, the speed of convergence of the methods of descent, and in particular that of the combined gradient, depends on the conditioning of the matrix $\eta(K)$. More it is close to its value floor, 1, better is convergence.

The principle of pre-packaging is then “simple”, it consists in replacing the linear system of the problem (P_1) by a system are equivalent of the type (pre-packaging on the left):

$$\left(\tilde{P}_1\right) \quad \underbrace{\mathbf{M^{-1}K}}_{\tilde{\mathbf{K}}}\,\mathbf{u}=\underbrace{\mathbf{M^{-1}f}}_{\tilde{\mathbf{f}}} \tag{4.1-1}$$

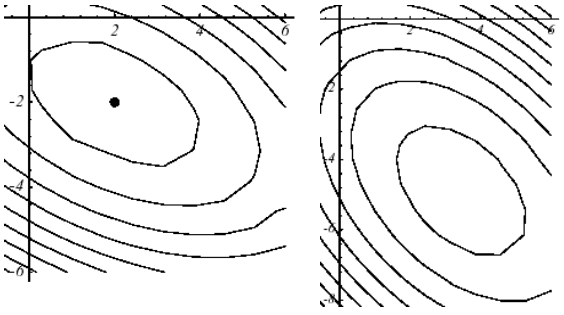
such as, ideally:

- Conditioning is obviously improved by it (this theoretical property, just like the following one, only are very seldom shown. They are often guaranteed only by digital experiments): $\eta(\tilde{K}) \ll \eta(K)$.
- Just like spectral distribution: more packed eigenvalues.
- M^{-1} that is to say inexpensive to evaluate (as for the initial operator, one has often right need to know the action of the preconditionnor on a vector): $\mathbf{Mv}=u$ easy to reverse.
- Even easy to establish and, possibly, effective to parallel.
- That it is as hollow as the initial matrix because it is a question of limiting the additional obstruction memory.
- That it preserves at the matrix of work $\tilde{K}$ same properties as that initial: here, character SPD.

In theory, the best choice would be $M^{-1}=K^{-1}$ because then $\eta\left(\tilde{K}=I_N\right)=1$, but if it is necessary to completely reverse the operator by a direct method to build this preconditionnor, it is only of little practical interest! Though, whom will be seen thereafter this idea is not as eccentric as that.

Known as differently, the objective of a preconditionnor is to thus pack the spectrum of the operator of work, as one already mentioned, his “effective conditioning” will be improved of pair with the convergence of the GCPC.

Graphically, that results in the more spherical shape of the graph of the quadratic form. Even on a system with $N=2$ dimensions and with a “rough” preconditionnor (cf figure 4.1-1), the effects are notable.



(A) (b)

Figure 4.1-1. Effect of the diagonal pre-packaging on the paraboloid of the n°1 example: (A) without, $\eta(K)=3.5$, (b) with, $\eta(\tilde{K})=2.8$.

In the absolute, one can prepack a linear system by the left (‘left preconditioning’), by the line (resp. ‘right’) or by making a mixture of both (resp. ‘Split’). It is this last version which will be retained for our operator SPD,

because one cannot directly apply the GC to solve $\left(\tilde{P}_1\right)$: even if M^{-1} and K are SPD, it is not inevitably the case of their product.

The trick then consists in using a matrix of pre-packaging SPD, M, for which one will thus be able to define another matrix (M being symmetrical real, it is diagonalisable in the form $M=\mathrm{UDU}^T$ with $D:=\mathrm{diag}\left(\lambda_i\right)\ \lambda_i>0$ and U orthogonal matrix). Required matrix SPD comes then from the associated decomposition $M^{1/2}=\mathrm{Udiag}\left(\sqrt{\lambda_i}\right)U^T$ with $M^{1/2}$ defined such as $(M^{1/2})^2=M$. From where the new problem of work, this time SPD

$$\left(\hat{P}_1\right)\ \underbrace{\mathrm{M}^{-1/2}\mathrm{KM}^{-1/2}}_{\hat{\mathrm{K}}}\underbrace{\mathrm{M}^{1/2}}_{\hat{\mathrm{u}}}\mathrm{u}=\underbrace{\mathrm{M}^{-1/2}}_{\hat{\mathrm{f}}}\mathrm{f} \qquad (4.1\text{-}2)$$

on which one will be able to apply the standard algorithm of the GC to constitute what is called a Gradient Combines Prepacked (GCPC).

4.1.2 Algorithm

In short, in substituent in the algorithm 4, expressions of the preceding problem $\left(\hat{P}_1\right)$ and while working a little with the simplification of the whole to handle only expressions in K, u and f, it occurs following unfolding.

Initialisation u^0 donné, $r^0=\mathbf{f-Ku^0}$, $d^0=M^{-1}r^0$, $g^0=d^0$
Boucle en i
(1) $z^i=\mathbf{Kd^i}$
(2) $\alpha^i=\dfrac{\langle r^i,g^i\rangle}{\langle d^i,z^i\rangle}$ (paramètre optimal de descente)
(3) $u^{i+1}=u^i+\alpha^i d^i$ (nouvel itéré)
(4) $r^{i+1}=r^i-\alpha^i z^i$ (nouveau résidu)
(5) Test d'arrêt via $\langle r^{i+1},r^{i+1}\rangle$ (par exemple)
(6) $g^{i+1}=M^{-1}r^{i+1}$ ($r$ésidu préconditionné)
(7) $\beta^{i+1}=\dfrac{\langle r^{i+1},g^{i+1}\rangle}{\langle r^i,g^i\rangle}$ (paramètre de conjugaison optimal)
(8) $d^{i+1}=g^{i+1}+\beta^{i+1}\ d^i$ (nouvelle dd)

Algorithm 5: Prepacked combined gradient (GCPC).

But makes some, the symmetrical character of the initial prepacked problem $\left(\widetilde{P}_1\right)$ is quite relative. It is indissociable subjacent scalar product. So instead of taking the usual Euclidean scalar product, one uses a definite matric scalar product compared to K, M or M^{-1}, it is possible to symmetrize the prepacked problem which was not it initially. How for the methods of Krylov into modal, it is the couple (operator of work, scalar product) which it is necessary to modulate to adapt to the problem!

Thus, $M^{-1}K$ being symmetrical compared to M- scalar product, one can substitute this new operator of work and this new scalar product in the algorithm of the GC not prepacked (algorithm 4)

$$K\Leftarrow M^{-1}K$$
$$\langle,\rangle\Leftarrow\langle,\rangle_M$$

And (surprised Ô!) by working the expressions a little, one finds the algorithm of the preceding GCPC exactly (algorithm 5). One proceeds in the same way with a pre-packaging on the right, $K M^{-1}$, via one M^{-1} - scalar product. Therefore, pre-packaging on the right, on the left or "splitté with mode SPD", all lead rigorously to the same algorithm. This observation will be useful thereafter to us when one realizes that the matrices provided by *Code_Aster* (and as préconditionneurs as one their will associate) are not always in conformity with the ideal scenario $\left(\hat{P}_1\right)$.

Note:

- *This alternative of the GCPC, which by far is spread, is sometimes called in the literature: combined gradient prepacked not transformed ('untransformed preconditioned conjugate gradient'). In opposition to the transformed version which handles the clean entities of the new formulation.*
- *The general methods of descent and, particularly the nonlinear GC, are also prepacked (cf [§2.3]). This time with a preconditionnor approximating the reverse of Hessien to the point considered, so as to make spherical the graph of the functional calculus in the vicinity of this point.*

4.1.3 "Overflight" of the principal préconditionneurs

The solution often adopted for its simplicity of implementation, his report "effectiveness digital/overcost calculation" for problems too badly not conditioned, consists in prepacking by the diagonal of the initial operator

$$M_J := \mathbf{diag}\left(K_{ii}\right) \tag{4.1-3}$$

It is what is called the diagonal pre-packaging or of Jacobi (JCG for 'Jacobi Conjugate Gradient') per reference to the stationary method of the same name.

Note:

- *It is a solution often adopted in CFD (for EDF R & D: Code_Saturne, TELEMAC). In these fields, a great attention paid to the diagram of resolution nonlinear and construction of the grid produce a problem conditioned often better. It is what made the fame of the JCG, transformed for the occasion into true "animal of race" dedicated to the large parallel calculators.*
- *It is a suggested solution, in mechanics of the structures, by a good amount of commercial codes: ANSYS, Zébulon, Nastran. It was present in Code_Aster but its unreliability led to its resorption.*
- *This principle of preconditionnor "of poor" extends to the methods of descent by taking this time the diagonal of Hessien.*

Another very widespread preconditionnor is the SSOR (for Symetric Successive Over Relieving). Like the precedent, he is deduced from a classical iterative method: method of Gauss-Seidel released. By breaking up the initial operator in the usual form $K := D + L + L^T$ where D is its diagonal and L its strictly lower triangular part, it is written (according to the parameter relieving ω)

$$M_{SSOR}(\omega) := \frac{1}{\omega(\omega-2)}\left(D+\omega L\right)D^{-1}\left(D+\omega L^T\right) \quad 0<\omega<2 \tag{4.1-4}$$

He has the advantage of not requiring report storage and of overcost calculation, since he is directly elaborate from K, while being very simple to reverse (an descent-increase). On model problems (famous "the Laplacian on the square unit") of the theoretical results were exhumed in finite elements

$$\eta(K) := \Theta\left(\frac{1}{h^2}\right) \Rightarrow \eta\left(M^{-1}K\right) := \Theta\left(\frac{1}{h}\right) \tag{4.1-5}$$

In addition, he has the ability, for an operator SPD, to fix quotas for his spectrum in the band $]0,1]$. However, it proved in practice industrial, less effective than the préconditionneurs of incomplete the Cholesky type than we will approach in the following paragraph. And it very poses the delicate problems of the choice of the parameter of optimal relaxation, by nature "problem-dependent".

Note:

- *This preconditionnor was proposed by D.J.Evans (1967) and was studied by O.Axelsson (1974). He declines himself in grinds versions: nonsymmetrical, by blocks, with parameter of optimal relaxation…*
- *While posing $\omega=1$ the Symmetrical typical case of Gauss-Seidel is found*

$$M_{SGS}(\omega) := -(D+L)D^{-1}(D+L^T) \quad (4.1\text{-}6)$$

A string of other préconditionneurs has thus been born for about thirty years in the literature: explicit (polynomial [BH89], SPAI, AINV…), implicit (Schwarz, IC…), multilevel (decomposition of field, multigrilles…) …. Some are specific of an application, others plus generals. The “effects of modes” also made their work! For more information one will be able to consult the monumental sum made by G.Meurant [Meu99] or the books of Y.Saad [Saa03] and H.A.Van der Vorst [Van01].

4.2 Incomplete factorization of Cholesky

4.2.1 Principle

One has just seen that the préconditionneurs is often inspired by linear solveurs except for whole: Jacobi for that diagonal, Gauss-Seidel for SSOR. That based on an Incomplete factorization of Cholesky (IC) does not escape the rule! But it is based this time, not on another iterative method, but on a direct method of Cholesky type. From where denomination of ICCG (‘Incomplete Cholesky Conjugate Gradient’) given to the coupling of this preconditionnor with the GCPC.

The initial operator being SPD, he admits a decomposition of Cholesky of the type $K = \boldsymbol{C}\boldsymbol{C}^{\boldsymbol{T}}$ where $\boldsymbol{C}$ is a lower triangular matrix. One calls incomplete factorization of Cholesky, the search for a matrix F triangular lower as digs as possible and such as FF^T that is to say near to K in a direction to be defined. For example, while posing $B = K - FF^T$ one will ask that the relative error (expressed in a matric standard with the choice)

$$\Delta := \frac{\|B\|}{\|K\|} \quad (4.2\text{-}1)$$

that is to say smallest possible. With the reading of this “rather evasive” definition one foresees the profusion of possible scenarios. Each one went there from its own incomplete factorization! The work of G.Meurant [Meu99] shows great diversity of it: $IC(n)$, $MIC(n)$, released, reordered, by blocks….

However, to be simplified the task, one often forces *a priori* the structure digs F, i.e. its graph

$$\Theta(F) := \{(i,j),\ 1 \le j \le i-1,\ 1 \le i \le N,\ F_{ij} \neq 0\} \quad (4.2\text{-}2)$$

It is obviously a question of finding a compromise: the wider this graph will be and the smaller the error (4.2-1) will be but more calculation and storage of what is not (in the case which interests us) that a preconditionnor will be expensive. Generally, the préconditionneurs are recursive and in their basic level, they impose on F the same structure as that of C : $\Theta(F) = \Theta(C)$.

Note:

- *Initially, these incomplete factorizations were developed to repeatedly solve a linear system of type* (P_1)

$$\mathrm{FF}^T u^{i+1} = f - \mathrm{Bu}^i \quad (4.2\text{-}3)$$

  The “nerve of the war” being then the spectral ray $\rho\left[\left(\mathrm{FF}^T\right)^{-1} B\right]$ *that a wise choice of* $\Theta(F)$ *can notably contribute to make fall.*
- *This principle of incomplete factorization spreads without sorrow with the standard case where the operator is written* $K = LU$ *with this time* $B = K - LU$ *. One speaks then about Incomplete factorization of type LU (ILU for ‘Incomplete LU’).*

4.2.2 Strategy retained in Code_Aster

It is about a preconditionnor of the type ILU (because we will see in the paragraph according to whether the matrices of work of *Code_Aster* often lose their definite-positivity) inspired of work of H. Van der Vorst [Van01]. The matrices remaining however symmetrical, one can write $K = LDL^T$ and $B = K - LDL^T$.

Note:

- *The matrix not being regular simply symmetrical SPD but one, a priori, is not ensured of the existence of one factorized LDL^T without resorting to permutations of lines and columns ($PK = LDL^T$ with P matrix of permutation). Scenario which was not envisaged in the native linear solveurs of Code_Aster ('LDLT', 'MULT_FRONT', 'GCPC') and would be difficult to effectively implement taking into account MORSE storage the matrices. Fortunately, it will be seen that a happy combination of circumstance makes it possible to resolve the situation (cf §5.1)!*

In any rigour one should speak about incomplete factorization of type ILDLT but in the literature and documentations of the codes, one amalgamates already ILU and IC, even their alternatives, it is not thus the sorrow to enrich the list by the acronyms !

This incomplete factorization, in the right line of the preceding theoretical recalls, is based on two reports which we now will back up:

- concept of filling by levels,
- low magnitude of the terms resulting from this filling.

4.2.3 Filling by levels

The construction of factorized is carried out, line by line, via the usual formula

$$L_{ij} = \frac{1}{D_j}\left(K_{ij} - \sum_{k=1}^{j-1} L_{ik} D_k L_{jk} \right) \tag{4.2-4}$$

From where the phenomenon of progressive filling of the profile ('English rope'): initially the matrix L with the same filling as stamps it K, but during the process, the null term of K_{ij} a term can correspond not no one of L_{ij}. It is enough that there exists a column k ($< j$) comprising a term not no one for the lines i and j (cf figure 4.2-1).

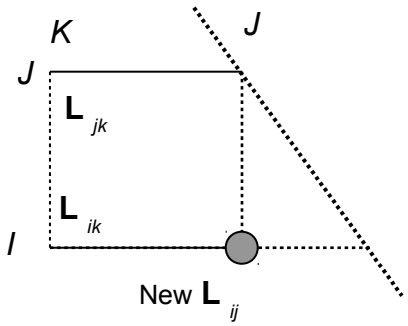


Figure 4.2-1. Phenomenon of filling during factorization.

Besides these nonworthless terms being able them same to correspond to former fillings, from where a concept of level of recursivity which can be interpreted like as many "levels" of filling. One will speak thus about factorized incomplete of level 0 (stored in $L(0)$) if it reproduces with identical the structure (but of course not values which is different) strict lower diagonal part of K (i.e the same graph). Factorized level 1 (resp. $L(1)$) will be able it to not include the filling induced by terms worthless of K (noted terms r^1 in the figure 4.2-2), that of level 2 (resp. $L(2)$) will be able to mingle the new nonworthless terms with it (r^1) precedents to constitute possible new terms (noted r^2), and so on recursively…

This is illustrated on the academic case of a matrix digs pentadiagonale resulting from the discretization differences finished of the Laplacian on a uniform grid 2D (cf appears 4.2-b). One represents only the structure of it: d , diagonal terms spectators, *, terms initially nonworthless, r^i , terms filled at the stage n°I.

The factorized incomplete one $L(3)$ conduit here with a complete filling, the relative error will be then worthless $\Delta=0$ and the GCPC will converge in an iteration. The interest of the exercise is of course purely didactic!

$$K=\begin{bmatrix} * & & & & & & & \\ * & * & & & \text{sym} & & & \\ 0 & * & * & & & & & \\ 0 & 0 & * & * & & & & \\ * & 0 & 0 & * & * & & & \\ 0 & * & 0 & 0 & * & * & & \\ 0 & 0 & * & 0 & 0 & * & * & \end{bmatrix} \quad L(0)=\begin{bmatrix} d & & & & & & & \\ * & d & & & 0 & & & \\ 0 & * & d & & & & & \\ 0 & 0 & * & d & & & & \\ * & 0 & 0 & * & d & & & \\ 0 & * & 0 & 0 & * & d & & \\ 0 & 0 & * & 0 & 0 & * & d & \end{bmatrix}$$

$$L(1)=\begin{bmatrix} d & & & & & & & \\ * & d & & & 0 & & & \\ 0 & * & d & & & & & \\ 0 & 0 & * & d & & & & \\ * & r^1 & 0 & * & d & & & \\ 0 & * & r^1 & 0 & * & d & & \\ 0 & 0 & * & r^1 & 0 & * & d & \end{bmatrix} \quad L(2)=\begin{bmatrix} d & & & & & & & \\ * & d & & & 0 & & & \\ 0 & * & d & & & & & \\ 0 & 0 & * & d & & & & \\ * & r^1 & 0 & * & d & & & \\ 0 & * & r^1 & r^2 & * & d & & \\ 0 & 0 & * & r^1 & r^2 & * & d & \end{bmatrix}$$

Figure 4.2-2. Structure of different factorized incomplete ILU(*p*) *on the academic case of the Laplacian.*

4.2.4 Low magnitude of the terms resulting from the filling

In addition, one empirically notices a very interesting property of these new terms resulting from the filling: their absolute value decrease while moving away from the zones already filled by the nonworthless terms of K. Maybe, in the preceding case, of the principal under-diagonal and that external. To be convinced some it is enough to visualize the following function (cf figure 4.2-3)

$$y(k):=\log_{10}\left(\sqrt{\frac{\sum_{i=1}^{N-k} L_{k+i,i}^2}{N-k}}\right) \tag{4.2-5}$$

who represents the orders of magnitude of the terms of k ième under-diagonal. For example $y(0)$ corresponds to the principal diagonal.

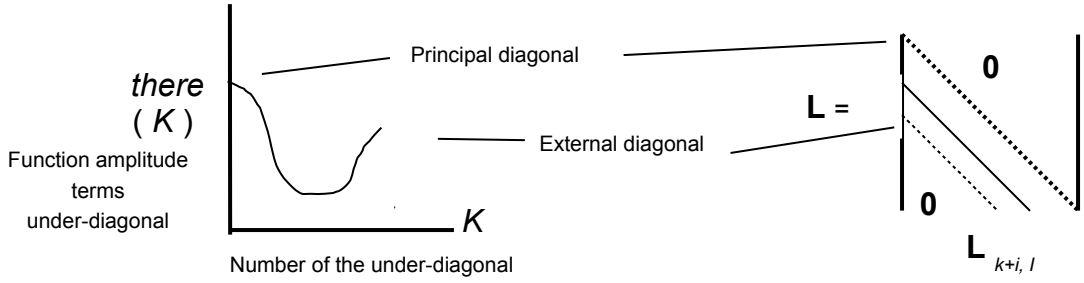


Figure 4.2-3. Relative importance of the diagonals of L

If one recapitulates, one thus has:
- level of a recursive and skeletal filling,
- a less importance of the terms corresponding to high levels of filling.

From where a certain “full power” left with incomplete factorizations $ILU(p)$ neglecting these median terms.

This approximation “cheaply” of K^{-1} will be used then as preconditionnor in algorithm 5 of the GCPC ($M^{-1}=K^{-1}$). To preserve a certain interest at the thing (if not to solve the problem as much directly!), one limits oneself to the first levels of filling: $p=0,1,2$ even 3 . All depends amongst linear systems which one will have to solve with the same matrix.

Note:
- *There exists little of theoretical results on this kind of preconditionnor. What does not prevent it from appearing often effective [Greg].*
- *It is a solution suggested by a good amount of great codes in mechanics of structures (ANSYS, Zébulon, Nastran…) and by all the bookstores of linear algebras (PETSc, TRILINOS, HYPRE…). On the other hand, this phase of pre-packaging is not always paralleled. Only the solvor of Krylov (GC, GMRES…) is. One then finds the already mentioned problems of ‘parallel preconditioning’.*

4.2.5 Complexity and occupation memory

As regards the cost additional calculation with one factorized incomplete, it is very difficult to estimate and, in practice, depends largely on the way in which it was coded (to our knowledge, there do not exist theoretical results on the thing). With a low level of filling, one hopes only that it is much lower than $\Theta\left(\frac{N^3}{3}\right)$ of a complete factorization.

Because a compromise is to be found between occupation memory and complexity. To constitute these factorized incomplete, it is often necessary to allocate temporary workspaces. Thus, in the establishment of the pre-packaging $ILU(p)$ of *Code_Aster*, it was selected to facilitate the algorithmic one (sorting, search of indices and coefficients in the profile, management of the recursivity and the filling…) by creating matrices temporarily representing storage full with the initial matrix and its filling.

With this space of temporary work, additional storage due to the final preconditionnor is added of course. On the level 0, it is theoretically at least of the same order as that of the matrix. From where minimal effective a memory complexity of the GCPC of $\Theta(2\,cN)$ realities and $\Theta(2\,cN+2\mathrm{N})$ entireties. When one goes up in completeness, only the practice can bring a pretence of answer.

In short, one shows empirically with *Code_Aster*, that it is necessary to envisage an obstruction memory total of $\Theta(\alpha\,cN)$ realities and $\Theta(\alpha\,cN+2\mathrm{N})$ entireties with
- $\alpha=2,5$ in $ILU(0)$ (level by default in *Code_Aster*),
- $\alpha=4,5$ in $ILU(1)$,
- $\alpha=8,5$ in $ILU(2)$.

For more precise details on the data-processing establishment of the GCPC in *Code_Aster* and its use on CAS-tests quasi-industrialists, one will be able to consult the notes [Greg] or [Boi03].

5 The “native” GCPC of *Code_Aster*

Two types of combined gradient are usable:

1. The combined gradient "historical" [Greg] of the code (keyword SOLVEUR/METHODE=' GCPC'), completely integrated into its sources and member with his business package report.
2. That of PETSC [FART] (keyword SOLVEUR/METHODE=' PETSC'+ALGORITHME=' CG') called as an external bookstore.

This chapter approaches the difficulties of establishment in *Code_Aster* first. The following chapter deals more specifically with PETSC.

5.1 Particular difficulties

5.1.1 Taking into account of the limiting conditions

In *Code_Aster*, there are two manners of taking into account the boundary conditions and this stage is carried out during the effective construction of the matrix of rigidity:

- By double dualisation [Pel01] (operators AFFE_CHAR_ACOU/MECA/THER) by using ddls specific, known as of Lagrange, which include the groups of unknown factors concerned and make it possible to check all types of linear limiting conditions (Dirichlet generalized)

  • $\mathrm{T\,u}=0$ (5.1-1)

  with T real matrix of size $p\times n$. This technique packs the matrix of rigidity in a new matrix, known as "dualized", which becomes the new matrix of work

  • $$\tilde{\mathrm{K}}=\begin{pmatrix} \mathrm{K} & \beta\,\mathrm{T}^{\mathrm{T}} & \beta\,\mathrm{T}^{\mathrm{T}} \\ \beta\,\mathrm{T} & -\alpha\,\mathrm{Id} & \alpha\,\mathrm{Id} \\ \beta\,\mathrm{T} & \alpha\,\mathrm{Id} & -\alpha\,\mathrm{Id} \end{pmatrix} \quad (5.1\text{-}2)$$

  where α and β are two strictly positive realities.

- By simple elimination of the unknown factors (operators AFFE_CHAR_CINE) in substituent and carrying out the setting zero have of p lines and columns concerned of the matrix of rigidity. This is valid only for blockings of ddls, one cannot thus take into linear account of relation. The matrix of rigidity is written then

  $$\tilde{\mathrm{K}}=\begin{pmatrix} \bar{\mathrm{K}} & 0 \\ 0 & \mathrm{Id} \end{pmatrix} \quad (5.1\text{-}3)$$

  while noting $\bar{\mathrm{K}}$ its unchanged part.

Each of the two approaches has its advantages and its disadvantages: generics, modularity but increase in the size of the problem, degradation of its conditioning and loss of its definite positivity for the first. Contrary to the second which decreases by it the size but which is circumscribed with certain types of limiting conditions and is, by means of computer, more delicate to implement.

Note:

Other approaches were possible: simple dualisation, taken into account of the limiting conditions in the variational formulation, gradient combined project…

5.1.2 Consequence on the GCPC

With AFFE_CHAR_CINE , the operator of work $\tilde{\mathrm{K}}$ remaining SPD, all the theory recalled in the preceding paragraphs applies. On the other hand, with AFFE_CHAR_ACOU/MECA/THER (the most frequent case in practice), it is not any more the case, because it becomes simply symmetrical and loses its positive definite character. The consequences are then of three orders:

- The total convergence of the GCPC (cf (3.2-9)) is not guaranteed any more,

- When it occurs, it is slowed down (cf (3.2-11)) by a degraded conditioning, $\eta(K) \ll \eta(\tilde{K})$
- The pre-packaging cannot be carried out any more *via* one $IC(p)$, but rather by one $ILU(p)$ (cf §4.2). It is necessary still that factorization LDL^T that is to say always possible without having to permute line or column!

Fortunately, an adequate provision of the multipliers of Lagrange compared to the groups of ddls which they relate to (they must include these degrees of freedom cf [Pel01] §4), makes it possible to obtain without blow to férir this incomplete factorization (phew!).

Note:

- *In a report EDF B.Nitrosso [Ni92] shows that the diagonal pre-packaging is not possible (cf §4.1) , because it leads to the cancellation of the scalar product to the denominator of the stage (7) of algorithm 5: calculation of the optimal parameter of conjugation. Therefore, contrary to Code_Saturne, Code_Aster cannot propose this not very reliable option.*
- *It does not remain about it less, that with dualized limiting conditions, the conditioning of the operator of work is degraded and thus, the convergence of the slowed down GCPC. Other commercial codes, with the instart of ANSYS [PLM], already made this report.*

5.1.3 Obstruction memory

Taking into account the elements of the §4.2, one empirically noted effective a memory complexity of the GCPC with the preconditionnor of Incomplete the Cholesky type ('LDLT_INC'), in α time size of K with:

- $\alpha = 2,5$ in $ILU(0)$ (level by default in *Code_Aster*),
- $\alpha = 4,5$ in $ILU(1)$,
- $\alpha = 8,5$ in $ILU(2)$.

With the preconditionnor using a factorization single precision resulting from MUMPS ('LDLT_SP'), consumption memory is more important (at least $\alpha > 10$). But into nonlinear, this factorization can be preserved during several resolutions of systems (keyword REAC_PRECOND). The overcost calculation due to its construction is thus finally less.

Note:

- *These figures are to be compared in keeping with the factorized complete one which is of the order typically of $\alpha = 30$ with 100 (that depends in particular on the renumerotor used in preprocessing). The disproportion of these figures illustrates well the principal competitive advantage of the GCPC compared to the direct solveurs: the occupation necessary memory.*
- *However, contrary to the direct solveurs of Code_Aster , the GCPC could not profit from a pagination memory. All the matrix must be contained in RAM because it is used a large number of times via the product matrix-vector. Whereas the direct methods handle objects much larger but those are discharged automatically and partially on disc (software package JEVEUX and/or Out-Of-Core faculties of MUMPS).*

5.1.4 Parallelism

Contrary to other solveurs of the code (MULT_FRONT, MUMPS, PETSC) and for problems of 'parallel preconditioning', the native GCPC is usable only into sequential (cf. § 3.3).
For example, it does not profit from the distribution of the tasks and the data that the parallelism MPI set up around direct solvor MUMPS gets it. Thus, a overcost report of $\alpha = 30$ with MUMPS In-Core sequential can very quickly go down on the level from a GCPC prepacked by *ILU (0)* as soon as a dozen processors are used and/or that his Out-Of-Core faculties are activated.

5.2 Perimeter of use

All operators generating of the real symmetrical systems except those requiring a detection of singularity obligatorily (modal calculation, stability and buckling…). Into nonlinear, if the problem is real nonsymmetrical, one can use this solvor provided that one forces symmetrization (cf §5.3).
List of the operators *Code_Aster* being able to use GCPC:

- CALC_FORC_AJOU,
- CALC_MATR_AJOU,
- CALC_PRECONT → Possible symmetrized approximation (cf. §5.3),
- DYNA_LINE_TRAN,
- DYNA_NON_LINE → Possible symmetrized approximation,
- MACRO_MATR_AJOU,
- ASSEMBLY,
- MECA_STATIQUE,
- NUME_DDL/FACTORISER/RESOUDRE,
- MODE_STATIQUE,
- STAT_NON_LINE → Possible symmetrized approximation,
- THER_LINEAIRE,
- THER_NON_LINE → Possible symmetrized approximation,
- THER_NON_LINE_MO → Possible symmetrized approximation.

5.3 Symmetrical character of the operator of work

If the matrix is not symmetrical two cases arise. Either the linear solvor is inserted in a non-linear process (operators mentioned cf lists §5.2), or this one is linear (other operators).

In the first case, the initial problem is transformed $\mathrm{Ku}=f$ in a new symmetrized problem $\frac{1}{2}\left(K+K^T\right)u=f$.
One supposes by there that the nonlinear solvor including (algorithm of Newton) will compensate for the approximation of the initial operator by $\tilde{K}:=\frac{1}{2}\left(K+K^T\right)$. It is not besides the only approximation of this nonlinear process… the choice of the tangent matrix in is, for example, another.

This symmetrized approximation does not harm the robustness and the coherence of the unit and avoids a more expensive nonsymmetrical resolution. This operation is carried out by activating the keyword SYME= 'OUI'(by default 'NOT') keyword factor SOLVEUR and it is licit for all the nonlinear solveurs of the code.

When the problem is purely linear, one can expect no compensation of any including process and this approximation becomes impossible. Resorts to the native GCPC is illicit and this nonsymmetrical system should be solved *via* other linear solveurs. The nature of the operator of work is detected automatically, no one is not need to notify it *via* the keyword SYME.

5.4 Parameter setting and posting

To activate this functionality, the keyword should be initialized METHOD with 'GCPC' in the keyword factor SOLVEUR [U4.50.01]. In addition, Dthey standard préconditionneurs are available (keyword PRE_COND):

- PRE_COND=' LDLT_INC': Incomplete a Cholesky classic (cf §4.2) whose level of filling is controlled by the keyword NIVE_REMPLISSAGE=K (K=0 does not mean "without pre-packaging" but pre-packaging of the type ILU(0)).
- PRE_COND=' LDLT_SP': A factorization single precision carried out by direct solvor external MUMPS [R6.02.03]. This solution is, *a priori* , more expensive than the first in terms of memory/time but this one are mutualisable, into nonlinear, between various resolutions of linear systems (keyword REAC_ITER). On the other hand, the robustness and the quality of this preconditionnor can accelerate the convergence of the GCPC largely.

Note:

- *To minimize the size of the profile of the matrix of work and its preconditionnor (with Incomplete Cholesky), an algorithm of renumerotation is available by default: 'RCMK' for 'Reverse Cuthill Mac-Kee' (cf [LT98] §5.4.2). It is nevertheless désactivable.*
- *The "arm of principal lever" to modify the compromise "time calculation/occupation memory" of the GCPC, remains the preconditionnor and his various digital parameters.*

Contrary to the direct methods, it was necessary to fix a maximum iteration count discriminating converged iterative calculations of not converged. This threshold (skeletal) is arbitrarily fixed at half of the ddls of the problem of work $\hat{\mathbf{K}}$. So at the end of *I*=NMAX_ITER stages, the relative residue $\delta^i := \frac{||r^i||}{||f||}$ is not lower than RESI_RELA, calculation stops in ERREUR_FATALE.

Keyword factor	Keyword	Value by default	References
SOLVEUR	METHODE=' GCPC'	'MULT_FRONT'	[§1]
	PRE_COND=' LDLT_INC'/ 'LDLT_SP'	'LDLT_INC'	[§4.2] [§5.4]
	NIVE_REMPLISSAGE=0,1,2...	0	[§4.2]
	REAC_PRECOND	5	[§5.4]
	RENUM=' RCMK' or 'WITHOUT'	'RCMK'	[§5.4]
	NMAX_ITER= i_{max}, acceptable iteration count maximum. If $i_{max}=0$, fixed automatically at $N/2$	0	[Algorithm 5]
	Test of stop in relative residue. With the iteration *I* $\frac{||r^i||}{||f||} < \text{RESI\_RELA}$.	10^{-6}	[Algorithm 5 and §3.3]
	SYME=' OUI' or 'NOT' Symmetrized approximation of the operator of work by $\tilde{K} := \frac{1}{2}\left(K + K^T\right)$ Sell by auction only into nonlinear	'NOT'	[§5.2/3]

Table 5.4-1. Summary of the parameter setting of the native GCPC.

To be complete on the parameter setting and posting in the file message (.mess), let us mention:

- The value INFORMATION =2 trace evolution of the standards of the residues absolute $||r^i||$ and relative $\frac{||r^i||}{||f||}$, as well as the correpondant number of iteration, *I*, as soon as it decrease from at least 10% (nonskeletal).
- With convergence, i.e. as soon as $\frac{||r^i||}{||f||} < \text{RESI\_RELA}$, one recalls in more the value of the standard of the initial absolute residue $||r^0||$.

5.5 The Councils of use

All in all, the best compromise robustness/obstruction memory/cost CPU seems to return [Anf03] [Boi03] with the two direct solveurs: the multifrontale and MUMPS. Especially when one deals with problems of the type

“multiple second members” (e.g. operator DYNA/STAT/THER_NON_LINE without reactualization of the tangent matrix).
However, for problems conditioned rather well (thermal, simple geometry with a grid and characteristic relatively homogeneous materials…) or very greedy in memory (several million ddls), the GCPC can prove to be an interesting alternative. Especially with the preconditionnor LDLT_SP, who is more expensive but also more reliable..

Because of its natural modularity and simplicity of its components (produced matrix-vector and scalar product), the GCPC remains much simpler to maintain and make evolve that the other direct solveurs. It is the solvor “passes everywhere” easy to establish (at least in its initial version) and very teaching. It is often connected in processes plus generals (encased solveurs, solveurs of interface of the decomposition of fields…) or adapted, on a case-by-case basis, for structures of particular matrices.

6 Iterative solveurs of Krylov *via* PETSc

6.1 The PETSc bookstore

The bookstore PETSc [FART] (for 'Extensive Portable Toolkit for Scientific Computation') of the laboratory Argonne (DOE and NSF) proposes, in addition to a large variety of solveurs (direct, iterative, modal, ODE…), utilities of handling of distributed data and profiling of parallel calculations. It is a product of the public domain, interfaced with various languages (F77/90, C/C++, Python), paralleled in MPI and which reached a certain maturity[27].
Certain codes, such MEF++ [Tar04] did not hesitate to lean completely in PETSc to fully benefit from its aspects toolkits and its faculties of parallelism. This aspect performance[28] is the priority of the product which asserts long time of the records on the matter (resolution of linear systems of 500 million unknown factors in 2004).

6.2 Establishment in *Code_Aster*

The building site software of the establishment of PETSc in Code_Aster is detailed in the note of T.De Soza [Des07] and traced in the tool of Rex *Aster*. We synthesize the broad outlines here of them:

1) A PETSc execution *via Code_Aster* is held in three stages: pre-allowance of the memory capacity and filling of the matrix, parameter setting of the solvor of selected Krylov and its preconditionnor, supply of the second member and effective resolution.
2) The conversion of the format of matrix *Aster* (format CSC 'Compressed Sparse Column') with the format suitable for PETSc (CSR 'Compressed Sparse Row'). Into symmetrical, one does not store in *Aster* that the higher triangular part whereas PETSc requires all the matrix. In addition, PETSc needs to be effective particular indicators: into sequential, the median number of nonworthless terms per line, in parallel, that of the diagonal and extra-diagonal blocks of the block of lines associated with the processor. This phase of preallocation of the PETSc objects is tiresome but fortunately inexpensive compared to the other stages[29].
3) In parallel mode, the distribution of the data is entirely controlled by PETSc. This last allots a block of lines contiguous to each processor. The user *Aster* cannot, for the moment, to intervene truly on this choice (except for changing the number of processors!). It is same philosophy as for OpenMP parallelism of the multifrontale *Aster*. With MUMPS, the choice is broader: the user can distribute his data directly or *via* automatic tools (possibly controllable).
4) Indefinite character of the majority of the matrices Aster (Lagrange, finite elements mixed…) prevent the use of the renuméroteurs and préconditionneurs suitable for PETSc. To be able to function effectively, one "bluffs" the tool by providing him renumbered matrices[30] by the RCMK[31] of*Aster* (which manages Lagranges specifically).

6.3 Perimeter of use

Concerning the operators *Aster*, it is the same one as for the native GCPC of the code (cf. §5.2).

As regards the features DE the bookstore PETSc, we limited ourselves for the moment to some iterative linear solveurs of Krylov type:

27 The project began in 1991 and among its characteristics let us quote: about twenty developers (core-TEAM: 9), a version a year, tens of thousands of users.
28 One of the founding fathers of PETSc, William Gropp [WG], also works with the standardisation of language MPI (MPICH). It is very active on the data-processing and algorithmic aspects of parallel calculation.
29 Tools for preparation of the data (cf MURGE [WALL]) or macro-bookstores (Numerical Plato, Mystery…) could in the long term release the codes hosts of these tiresome contingencies which are the filling of the structures of data specific to each external product.
30 For the préconditionneurs with incomplete factorization (ILU (*K*) cf. §4.2).
31 RCMK for algorithm of renumerotation of the type MacKee Reverse-Cuthill.

- CG ('CG'): standard combined gradient (the same one as the native GCPC D'*Aster; M.R.Hestenes and E.Stiefel 1952*),
- CR ('CR'): combined residue (*M.R.Hestenes and E.Stiefel 1952*),
- GMRES ('GMRES'): Minimal Generalised RESidual (*Saad and Schultz 1986*),
- FGMRES ('FGMRES') : Flexible device Minimal Generalised RESidual
- GCR ('GCR'): Generalised Conjugate Residual (*S.C. Eisenstat, H.C. Elman, and H.C. Schultz 1983*).

and with the préconditionneurs:

- ILU (K) ('LDLT_INC'): incomplete factorization by level (the same one as the native GCPC D'*Aster*; cf. §4.2),
- JACOBI ('JACOBI'): standard diagonal preconditionnor (cf. §4.1),
- SOR ('SOR'): Successive Over Relieving (cf. §4.2).
- Boomeramg ('BOOMER'): multigrille algebraic provided by the library *Hypre*.
- Multilevel ('Ml'): multigrille algebraic provided by the library *Ml*.
- GAMG ('GAMG'): multigrille algebraic of the PETSc library

In addition as for 'GCPC' another pre-conditioner (intern with *Code_Aster*) is available: it is about 'LDLT_SP'which is based on a factorization single precision renewed during several iterative resolutions.
The preconditionnor 'BLOC_LAGR' is a preconditionnor per blocks of the Lagrangian type increased which applies to the systems with multipliers of Lagrange. It is an internal preconditionnor with code_aster developed on PETSc basis. A system with multipliers of Lagrange has the structure of point-saddles following:

$$\begin{pmatrix} K & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix} \quad (1)$$

The preconditionnor of the Lagrangian type increased (see [Mer15]) is form:

$$M = \begin{pmatrix} K + \gamma B^T B & 2B^T \\ 0 & -\frac{1}{\gamma} I \end{pmatrix} \quad (2)$$

Lastly, it is possible to combine iterative solvor GMRES with preconditionnor LDLT_SP (known as preconditionnor of first level) and a preconditionnor with memory limited "Limited Memory To prepack", known as preconditionnor of second level. This combination makes it possible to accelerate the resolution of a non-linear problem. Indeed, the solution of the non-linear problem is obtained by solving a succession of linearized problems. The preconditionnor of second-level is built starting from spectral information (even of Ritz) exits of the preceding linear resolutions (see [Mer15]).

Note:

- *The methods CG and CR are to be held for modelings of Aster leading to matrices SPD (the most widespread case notwithstanding the problems of doubles Lagranges cf. §5.1/ 6.2). In nonsymmetrical, it is necessary to call on other methods (GMRES and GCR).*
- *All the préconditionneurs of PETSc propose a string of parameters. For the moment, only the factor of filling of the ILU is accessible to the user.*
- *In parallel mode, only JACOBI offers a preconditionnor strictly equivalent to the sequential mode. The others, with first chief ILU and SOR, prepack by using the local diagonal block with the processor.*

6.4 Symmetrical character of the operator of work

One can pass the same remarks as for the native GCPC (§ 5.3). Except, that with PETSc, one can use a solvor of Krylov compatible with the nonsymmetrical systems (GMRES, GCR). The parameter 'SYME' thus much of its interest loses.

6.5 Parameter setting and posting

To activate this functionality, the keyword should be initialized METHOD with 'PETSC' in the keyword factor SOLVEUR [U4.50.01]. All in all, one can make the same comments as with § 5.4.

Keyword factor	Keyword	Value by default	References
SOLVEUR	METHODE=' PETSC'	'MULT_FRONT'	[§1]
	ALGORITHME=' CG'/'CR'/ 'GMRES'/'GCR'/'FGMRES'/'GM RES_LMP'	'FGMRES'	[§6.3]
	PRE_COND=' LDLT_INC' /'JACOBI'/'SOR'/'BOOMER'/' ML'/'WITHOUT'	'LDLT_SP'	[§6.3]
	NIVE_REMPLISSAGE=0,1,2…	0 (if LDLT_INC)	[§4.2]
	REMPLISSAGE= α estimated size of the preconditionnor	1.0 (if LDLT_INC)	[§5.1]
	RENUM=' RCMK' or 'WITHOUT'	'RCMK'	
	NMAX_ITER= i_{max} , acceptable iteration count maximum. If $i_{max} \leq 0$, fixed automatically by PETSc (parameterized by PETSC_DEFAULT with maxits=10^5)	-1	[Algorithm 5] for GC; equivalent for the other solveurs of Krylov.
	Test of convergence: $\|r^i\| < \max(\text{RESI\_RELA}.\|f\|, 10^{-50})$ Test of divergence: $\|r^i\| > 10^5.\|f\|$	10^{-6}	[Algorithm 5] for GC; equivalent for the other solveurs of Krylov.
	SYME=' OUI' or 'NOT' Symmetrized approximation of the operator of work. Sell by auction only into nonlinear. Interest has only for 'CG' and 'CR'.	'NOT'	[§6.4]

Table 6.5-1. Summary of the parameter setting of the call to PETSc via Code_Aster.

6.6 The Councils of use

One can lavish the same advices of use as for the native GCPC (cf. § 5.5). Nonosbstant the fact that, contrary to the native GCPC, certain solveurs of PETSc are adapted to the nonsymmetrical systems.
In general, the use of the combined gradient "simple" (native GCPC or CG of PETSc) is not very robust on modelings Aster. More sophisticated algorithms are preferred to him: GMRES… On large calculations ($>5.10^5$ degrees of freedom) very consuming of resolution of linear systems (operators STAT_NON_LINE, MECA_STATIQUE…), one may find it very beneficial to activate the parallelism induced by PETSc.

7 Treatment of the failures

7.1 Failures met

Contrary to a direct solvor, an iterative solvor will in general tend to fail at the time of the phase of resolution of the linear system: for example if the convergence criteria are not checked at the end of the maximum number of iterations fixed beforehand. That can be caused by a pre-conditioner not robust enough or a quasi singular matrix.
Failures are also possible at the time of the phase of construction of the pre-conditioner but are rarer: for example if MUMPS lack of memory during factorization single precision to build the pre-conditioner LDLT_SP. In this last case the solution is to start again calculation with more memory or by increasing the value of PCENT_PIVOT.

7.2 Recovery in the event of failure

Like the direct solveurs, the iterative solveurs of *Code_Aster* are able to announce the failure or the success of the linear resolution. This functionality is exploited in the non-linear solvor [R5.03.01] in order to allow:
– the cutting of the step of time in the event of failure at the time of the phase of resolution whatever the pre-conditioner
– reactualization of the pre-conditioner when LDLT_SP is used

When LDLT_SP is used, one carries out the reactualization initially, then one carries out the step of current time again. If a new failure occurs, the step of time is then cut out.

It is it should be noted that no recovery is envisaged in the event of failure at the time of the phase of construction of the pre-conditioner.

7.3 Implementation

The operator DEFI_LIST_INST allows to activate the treatment of the failures. In the presence of an action of cutting (ACTION=' DECOUPE') or of reactualization of the pre-conditioner (ACTION=' REAC_PRECOND'), one activates in STAT_NON_LINE or DYNA_NON_LINE the capture of the errors in the iterative solveurs.

7.4 Interception of the exception

In the event of failure of the strategy of cutting or reactualization of the pre-conditioner, an exception is raised. It can be intercepted in the command file by using its name: ResolutionError.

8 Bibliography

8.1 Books/articles/proceedings/theses…

[BFG] A.Bouras, V.Fraysse & L.Giraud. *With relieving strategy for inner-outer linear solvers in DD methods* . Report CERFACS TR/PA/00/17 (2000).
[BGLS] J.F.Bonnans, J.C.Gilbert, C.Lemarechal & C.Sagastizabal. *Digital optimization: theoretical and practical aspects* . ED. Springer collection mathematics & applications, 27 (1997).
[Ci00] B.A. Cipra. *The best of the 20th century: editors name top10 algorithms.* SIAM News, 33-4 (2000).
[Che05] K.Chen. *Matrix preconditioning technical and applications* . ED. Cambridge University Close (2005).
[CM01] F.Chaitin-Chatelin & T.Meskausas. *Inner-outer iterations for solvers mode in structural mechanics: application to Code_Aster* . Report CERFACS (2001).
[Cu94] J.C. Culioli. *Introduction to optimization* . ED. Ellipse (1994).
[Dav06] T.A.Davis. *Direct methods for sparse linear systems* . ED. SIAM (2006).
[Duf06] I.S.Duff et al. *Direct methods for sparse matrices* . ED. Clarendon Close (2006).
[Go89] G.H.Golub & D.P.O' learly. *Somme history of the conjugate gradient and Lanczos algorithms : 1948-1976* . SIAM review, 31-1 (1989), pp50-102.
[Gol96] G.Golub & C. Van Loan. *Matrix computations* . ED. Johns Hopkins University Close (1996).
[GV97] G.H.Golub & H.A.Van der Vorst. *Closer to the solution: Iterative linear solvers. The state of the art in numerical analysis* . ED. Clarendon close (1997), pp93-92.
[GY97] G.H.Golub & Q.Ye. *Inaccurate preconditioned conjugate gradient method with inner-outer iteration* . Report Stanford University, SCCM-97-04 (1997).
[Jo84] P.Joly. *Methods of combined gradient* . N°84016 report of the lab. of digital analysis of PARIS VI (1984).
[LT98] P.Lascaux & R.Théodor. *Matric digital analysis applied to the art of the engineer* . ED. Masson (1998).
[Liu89] J.W.H.Liu. *Broad computer solution of sparse positive definite systems* . Prentice Hall (1981).
[Mer15] Sylvain Mercier *Fast nonlinear Solveurs in mechanics of the solids*, thesis of the University of Toulouse
[Meu99] G.Meurant. *Broad computer solution of linear systems* . ED. Elsevier (1999).
[Min08] M.Minoux. *Mathematical programming, theory and algorithms* . ED. Lavoisier (2008).
[Saa03] Y.Saad. *Iterative methods for sparse matrices* . ED. PWS (2003).
[Sh94] J.R.Shewchuk. *Year introduction to the conjugate gradient method without the agonizing bread.* Report interns University of Carnegie Mellon (1994).
[SV00] Y.Saad & H.A. Van Der Vorst. *Iterative solution of linear system in the 20th-century* . J. comp. Appl. Maths., 123 (2000), pp35-65.
[Van01] H. Van Der Vorst. *Iterative broad Krylov methods for linear systems* . ED. Cambridge University Close (2001).

8.2 Account-returned reports/EDF

[Anf03] N.Anfaoui. *A study of the performances of Code_Aster: proposal for an optimization* . Internship of mathematics applied of PARIS VI (2003).
[BH89] T. Buffard & J.M.Herard. *A method of polynomial pre-packaging for systems SPD* . Note EDF R & D HE-41/89.16 (1989).
[Boi01] O.Boiteau. *Algorithm of resolution for the generalized problem* . Reference material *Code_Aster* R5.01.01 (2001).
[Boi03] O.Boiteau. *Analysis of the establishment of the gradient combined in Code_Aster and tests of other alternatives via library CXML* . Report EDF R & D CRY 3/23/014 (2003).
[Boi08] O.Boiteau. *Decomposition of fields and parallelism* . Reference material *Code_Aster* . R6.01.03 (2008).
[Boi08b] O.Boiteau. *Improvement of the robustness of code TELEMAC in parallel mode* . Report EDF R & D I23/08/008 (2008).
[Boi09] O.Boiteau. *General information on the linear solveurs direct and use of product MUMPS* . Reference material *Code_Aster* . R6.02.03 (2009).
[Des07] T.DeSoza. *Evaluation and development of parallelism in Code_Aster* . Internship of Master degree ENPC (2007) and notes EDF R & D HT-62/08/01464 (2008).
[Greg] J.P.Gregoire. *Algorithm of the GCPC for the resolution of a symmetrical linear system: general presentation and instructions* . Note EDF R & D HI/4333-07 (1982).
Effective vectorization of the GC in the case of hollow symmetrical matrices . Note EDF R & D HI-72/6710 (1990).
Parallelization of the GC for hollow matrices on CRAY C98 . Note EDF R & D HI - 76/95/019 (1996).
Acceleration of the convergence of the combined gradient prepacked by using incomplete factorization by level. Note EDF R & D HI-76/00/005 (2000).
Parallelization in memory distributed of the GC and the code user. Note EDF R & D HI - 76/01/007 (2001).
GCPC and fast calculation of the eigenvalues. Note EDF R & D HI-76/01/001 (2001).
[Ni92] B.Nitrosso. *Direct methods for hollow matrices SPD. Basic techniques with the state of the art* . Note EDF R & D HE-41/92/27 (1992).
[Pel01] J.Pellet. *Dualisation of the limiting conditions* . Reference material *Code_Aster* R3.03.01 (2001).
[Tar04] N.Tardieu. *Parallelization of MEF++, principles and performances* . Report EDF R & D (2004).
[Tar09] N.Tardieu. *Solvor multigrille for the mechanics of the structures* . Note EDF R & D HT-62/08/01615 (2009).

8.3 Resources Internet

[WALL] *Official website of product MURGE* : http://murge.gforge.inria.fr/.
[FART] *Official website DE the bookstore PETSc* : http://www.mcs.anl.gov/petsc/petsc-as/.

[PLM] G.Poole, Y.C.Liu & J.Mandel: Advancing analysis capabilities in ANSYS though solver technology. (2001) available on site ANSYS.
[WG] Professional site of William Gropp: http://www.cs.uiuc.edu/homes/wgropp .

9 Description of the versions of the document

Version Aster	Author (S) Organization (S)	Description of the modifications
3.0	J.P.GREGOIRE EDF/R & D /SINETICS	Initial text
6.4	O.BOITEAU, J.P.GREGOIRE EDF/R & D /SINETICS J.PELLET EDF/R & D /AMA	
9.4	O.BOITEAU EDF/R & D /SINETICS	Corrections and addition of PETSC
v10.4	O.BOITEAU EDF/R & D /SINETICS	Formal corrections due to the bearings .doc/.odt; Corrections on parallelism; Addition of LDLT_SP.
V13.4	N.Béreux EDF/R & D /ERMES	Addition of the preconditionnor BLOC_LAGR and of the algorithm GMRES_LMP