Solveurs modaux et résolution du problème généralisé (GEP)

Résumé

Que cela soit pour étudier les vibrations d'une structure ou pour rechercher ses modes de flambement, le mécanicien doit souvent résoudre un problème modal : soit généralisé (GEP), soit quadratique (QEP) [R5.01.02]. Pour ce faire, ***Code_Aster*** **propose de plusieurs méthodes *via* les opérateurs** **MODE_ITER_INV** **et** **MODE_ITER_SIMULT**: puissances inverses et coefficient de Rayleigh, Lanczos, IRA, Bathe & Wilson et QZ. Elles ont chacune leur périmètre d'utilisation, leur avantages, leurs inconvénients et leur historique de développement.

Pour traiter efficacement de gros problèmes modaux (en taille de maillage et/ou en nombre de modes recherchés), on conseille l'usage de la macro-commande: **MACRO_MODE_MECA**. Elle décompose le calcul modal d'un GEP standard (symétrique et réel), en une succession de sous-calculs indépendants, moins coûteux, plus robustes et plus précis. Rien qu'en séquentiel, les **gains peuvent être notables**: facteurs 2 à 5 en temps, 2 ou 3 en pic RAM et 10 à 10^4 sur l'erreur moyenne des modes.
De plus, son **parallélisme multi-niveaux peut procurer des gains supplémentaires** de l'ordre de 20 en temps et 2 en pic RAM, en réservant une soixantaine de processeurs.

Dans la première partie du document, nous résumons la problématique générale de résolution d'un problème modal, les différentes classes de méthodes et leurs déclinaisons dans les librairies du domaine public. Toutes choses qu'il faut avoir à l'esprit avant d'aborder, dans la seconde partie, l'architecture générale d'un calcul modal dans *Code_Aster*. Puis nous détaillons les aspects numériques, informatiques et fonctionnels de chacune des approches disponibles dans le code.
Un chapitre spécifique détaille la mise en oeuvre du parallélisme et du calcul intensif dans le cadre des calculs modaux de type GEP standard.

Table des Matières

1 Introduction

Une majorité d'études concernant le **comportement dynamique de structures** est réalisée en effectuant une **analyse transitoire sur base modale**. Pour calculer ces modes de vibrations, de nombreux algorithmes ont été développés depuis une soixantaine d'années. Afin de faire face à l'augmentation continuelle de la taille des problèmes et à la dégradation des conditionnements des opérateurs discrétisés, seuls les plus efficaces et les plus robustes en pratique ont été incorporés dans les deux opérateurs modaux de *Code_Aster* : MODE_ITER_SIMULT et MODE_ITER_INV.

Les périmètres d'utilisation optimaux de ces opérateurs peuvent être dissociés. Lorsqu'il s'agit de **déterminer quelques valeurs propres** (typiquement une demi-douzaine) ou **d'affiner quelques estimations**, l'opérateur **MODE_ITER_INV** est tout à fait indiqué. Il regroupe des algorithmes heuristiques et ceux de type puissances (cf. §4).

Par contre, pour **capturer une partie significative du spectre,** on a recours à **MODE_ITER_SIMULT**. Ce dernier fédère les méthodes dites de «sous-espace» (Lanczos cf. §5/§6, IRAM §7, Bathe & Wilson §8) qui projettent l'opérateur de travail afin d'obtenir un problème modal approximé de taille plus réduite (traité alors par une méthode globale de type QR ou Jacobi).

Cet opérateur permet aussi de calculer de manière robuste le spectre entier du problème. Pour ce faire, on utilise une méthode globale de référence (méthode QZ cf. §9) qui calcule exhaustivement tous les modes. Compte-tenu de son coût, elle est cependant à réserver à certains usages : problème de petite taille (inférieure à 10^4 degrés de liberté) ou benchmark d'algorithmes.

Les deux opérateurs peuvent d'ailleurs se compléter car les méthodes mises en œuvre dans MODE_ITER_INV sont très performantes pour optimiser des modes propres déjà presque convergés. En une ou deux itérations, elles peuvent ainsi améliorer les vecteurs propres calculés *via* MODE_ITER_SIMULT. La projection sur base modale n'en sera que meilleure !

Dans la première partie du document nous résumons la problématique générale de résolution d'un problème modal, les différentes classes de méthodes et leurs déclinaisons dans les librairies du domaine public. Toutes choses qu'il faut avoir à l'esprit avant d'aborder, dans la seconde partie, l'architecture générale d'un calcul modal dans *Code_Aster*. Puis nous détaillons les aspects numériques, informatiques et fonctionnels de chacune des approches disponibles dans le code. Pour chaque méthode, on donne ses principales propriétés et ses limitations en reliant ces considérations, parfois complexes, à un paramétrage précis des opérateur de *Code_Aster*.

Opérateur/ Périmètre d'application	Algorithme	Mot-clé	Avantages	Inconvénients Remarques
MODE_ITER_INV				
1ère phase (heuristique)				Uniquement symétrique réel (GEP et QEP).
Calcul de quelques modes	Bissection (sans objet en QEP)	'SEPARE'		
Calcul de quelques modes	Bissection + Sécante (méthode de Müller-Traub en QEP)	**'AJUSTE'**	Meilleure précision	Coût calcul
Amélioration de quelques estimations	Initialisation par l'utilisateur	'PROCHE'	Reprise de valeurs propres estimées par un autre processus. Coût calcul de cette phase quasi-nul	Pas de capture de multiplicité

Opérateur/ Périmètre d'application	Algorithme	Mot-clé	Avantages	Inconvénients Remarques
2ième phase (méthode des puissances proprement dite)				Uniquement symétrique réel (GEP et QEP).
Méthode de base	Puissances inverses	**'DIRECT'**	Très bonne construction de vecteurs propres	Peu robuste
Option d'accélération	Quotient de Rayleigh (sans objet en QEP)	'RAYLEIGH'	Améliore la convergence	Coût calcul.
MODE_ITER_SIMULT				
Calcul d'une partie du spectre	Bathe & Wilson	'JACOBI'		Peu robuste. Uniquement symétrique réel (GEP).
	Lanczos (Newman-Pipano en GEP et Jennings en QEP)	'TRI_DIAG'	Détection spécifique des modes rigides.	Uniquement symétrique réel (GEP et QEP).
	IRAM (Sorensen)	**'SORENSEN'**	Robustesse accrue. Meilleures complexités calcul et mémoire. Contrôle de la qualité des modes.	Méthode par défaut. Portée en: - non symétrique réel, - avec $\mathbf{A}$ complexe symétrique.
Calcul de tout le spectre	QZ	**'QZ'**	Robustesse. Méthode de référence.	Très coûteuse en CPU et en mémoire. À réserver au petit cas ($<10^4$ degrés de liberté). Portée en: - non symétrique réel, - avec $\mathbf{A}$ complexe symétrique.

Tableau 1-1. Récapitulatif des solveurs modaux de Code_Aster (GEP et QEP).

Nota:

- L'implantation effective et la maintenance des solveurs modaux dans Code_Aster est le fruit d'un travail d'équipe : D.Séligmann, B.Quinnez, G.Devesa, O.Boiteau, O.Nicolas, E.Boyère, I.Nistor…
- On a essayé de constamment lier les différents items abordés et de limiter au strict minimum le recours à de longues démonstrations mathématiques. De toute façon, les nombreuses références qui émaillent le texte doivent permettent de rechercher l'information précise.
- L'objet de ce document n'est pas de détailler tous les aspects abordés, des ouvrages complets ayant déjà rempli cette mission. On citera notamment F.Chatelin[Cha88], G.H.Golub[GL89], P.Lascaux[LT86], B.N.Parlett[Par80], Y.Saad[Saa80], D.S.Watkins[Wat07] et la synthèse commise par J.L.Vaudescal[Vau00].

2 Généralités sur les solveurs modaux en mécanique des structures

2.1 Problèmes modaux

Que cela soit pour étudier les **vibrations d'une structure, éventuellement amortie et/ou tournante**, ou rechercher ses **modes de flambement**, le mécanicien doit souvent résoudre un problème modal. Pour ce faire, *Code_Aster* propose différents opérateurs[1] qui traitent deux types de problèmes modaux: les **généralisés** (GEP pour 'Generalized Eigenvalue Problem') et les **quadratiques** (QEP pour 'Quadratic Eigenvalue Problem'):

$$\begin{array}{c}\text{Trouver } (\lambda, \mathbf{u}) \text{ tel que} \\ (\mathbf{A}-\lambda\mathbf{B})\,\mathbf{u}=\mathbf{0} \quad (GEP) \\ (\mathbf{A}+\lambda\mathbf{B}+\lambda^2\mathbf{C})\,\mathbf{u}=\mathbf{0} \quad (QEP)\end{array} \tag{2.1-1}$$

Pour résoudre ces deux classes de problèmes, on les transforme souvent en un **problème modal standard** (SEP pour 'Standard Eigenvalue Problem'). Ce type d'approche a le mérite d'être générique et de s'appuyer alors sur des solveurs modaux classiques:

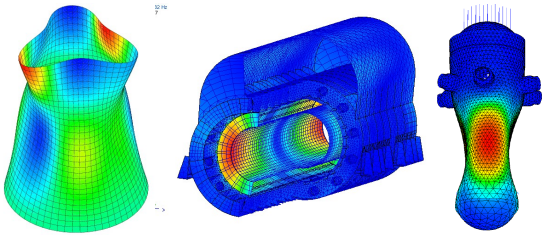
$$\begin{array}{c}\text{Trouver } (\tilde{\lambda}, \tilde{\mathbf{u}}) \text{ tel que} \\ (\mathbf{A}_\sigma-\tilde{\lambda}\,\mathbf{Id})\,\tilde{\mathbf{u}}=\mathbf{0} \quad (SEP)\end{array} \tag{2.1-2}$$

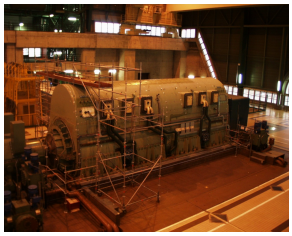
Une **transformation spectrale**, par exemple, du type 'shift and invert', permet de transformer un GEP en un SEP

$$\underbrace{(\mathbf{A}-\sigma\mathbf{B})^{-1}\mathbf{B}}_{\mathbf{A}_\sigma}\ \underbrace{\mathbf{u}}_{\tilde{\mathbf{u}}}-\underbrace{\frac{1}{\lambda-\sigma}}_{\tilde{\lambda}}\ \underbrace{\mathbf{u}}_{\tilde{\mathbf{u}}}=\mathbf{0} \tag{2.1-3}$$

Le complexe σ est un décalage spectrale (appelé souvent '**shift**') qui oriente la zone de recherche dans le spectre. Cette valeur peut être paramétrée dans *Code_Aster* , par exemple, par le mot-clé `CENTRE/FREQ` . Dans la même veine, on peut transformer un QEP en un GEP *via* une **technique de linéarisation** du type

$$\left[\underbrace{\begin{bmatrix}-\mathbf{A} & \mathbf{0}\\ \mathbf{0} & \mathbf{N}\end{bmatrix}}_{\tilde{\mathbf{A}}}-\underbrace{\lambda}_{\tilde{\lambda}}\underbrace{\begin{bmatrix}\mathbf{B} & \mathbf{C}\\ \mathbf{N} & \mathbf{0}\end{bmatrix}}_{\tilde{\mathbf{B}}}\right]\underbrace{\begin{bmatrix}\mathbf{u}\\ \lambda\mathbf{u}\end{bmatrix}}_{\tilde{\mathbf{u}}}=\mathbf{0} \quad \text{avec } \mathbf{N}=\alpha\,\mathbf{Id} \quad \left(\text{par ex. } \alpha:=\frac{(\|\mathbf{A}\|+\|\mathbf{B}\|+\|\mathbf{C}\|)}{3}\right) \tag{2.1-4}$$





1 `MODE_ITER_SIMULT`, `MODE_ITER_INV` et `CALC_MODAL/MACRO_MODE_MECA` (encapsulation de `MODE_ITER_SIMULT`).

Figure 2.1-1. Déformées modales de structures sollicitées dynamiquement (aéroréfrigérant, cuve de réacteur nucléaire) et contraintes modales sur un alternateur.

En mécanique des structures, suivant la problématique, les matrices $\mathbf{A}$, $\mathbf{B}$ et $\mathbf{C}$ précitées sont des combinaisons linéaires des différentes matrices mécaniques: masse, rigidité, rigidité géométrique, amortissement visqueux ou induit par la structure, effets gyroscopiques. Elles sont souvent réelles (sauf en présence d'amortissement hystérétique), mais pas toujours symétriques (par exemple, du fait d'effets gyroscopiques) et rarement définies positives (à cause des Lagranges notamment). Cet écart à la norme des problèmes modaux usuels (SEP symétriques et définis positifs) et cette variabilité des problématiques compliquent bien sûr leurs traitements algorithmiques.

Une **difficulté numérique majeure entre les QEP** et les deux autres classes de problèmes (GEP et SEP) concerne le tri des solutions calculées. Pour un problème de taille n, un QEP admet $2n$ modes propres (à valeur propre finie ou infinie) qui peuvent être dépareillées (réelle, imaginaire pure, complexe quelconque) ou en couple (, $(\lambda, \bar{\lambda})$ $(\lambda, -\bar{\lambda})$ …). Il faut alors définir des heuristiques robustes pour filtrer les valeurs propres souhaitées par l'utilisateur[2].

Une autre **complication inhérente au QEP est d'ordre algorithmique**. Il n'existe pas de décomposition de Schur (resp. Schur généralisé) comme pour les SEP (resp. GEP) sur laquelle va pouvoir s'appuyer l'algorithme de résolution. Par exemple, pour le SEP (2.1-2), cette décomposition nous assure l'existence d'une matrice unitaire (donc bien conditionnée et facilement inversible) $\mathbf{U}$, permettant la réécriture de la matrice de travail $\mathbf{A}_\sigma$ sous une forme plus facile à manipuler[3]: la matrice triangulaire supérieure $\mathbf{T}$.

$$\mathbf{U}\,\mathbf{A}_\sigma\,\mathbf{U}^* = \mathbf{T} \qquad (2.1\text{-}5)$$

2.2 Méthodes de résolution

Les solveurs modaux peuvent se regrouper en (au moins) **quatre familles**. Elles permettent de résoudre des SEP, parfois des GEP et rarement directement des QEP. Pour traiter ces deux derniers types de problème, on a déjà mentionné qu'il fallait souvent pré-traiter le problème mécanique initial (transformation spectrale, technique de linéarisation) pour construire une problème de travail «SEP-compatible» avec les solveurs modaux.

- **Les algorithmes de type QR** (cf. §9 et annexe 1) qui ont été présentés par H.Rutishauser (1958) et formalisés concurremment par J.C.Francis et V.N.Kublanovskaya (1961). QR est un algorithme fondamental souvent impliqué dans les autres méthodes. On le retrouve dans `MODE_ITER_SIMULT` (directement avec la méthode QZ cf. §9 et indirectement dans Lanczos et IRAM cf. §6,7).
  Périmètre d'utilisation: **Calcul de tout le spectre**.

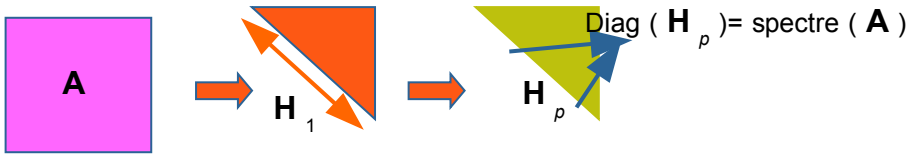


Figure 2.2-1. Schéma fonctionnel des méthodes de type QR: apparition des valeurs propres recherchées sur la diagonal.

Avantages: Bonne convergence, robustesse, calcul direct de la forme de Schur, adapté aussi au GEP.

2 Dans *Code_Aster*, en QEP on ne retient qu'un des deux modes couplés $(\lambda, \bar{\lambda})$. En l'occurrence celui à partie imaginaire positive. Suivant les cas de figure, la présence de valeurs propres d'autres natures (réelle, complexe non conjuguée) est signalée en `ALARME` ou à titre informatif. Attention, pour les GEP à modes complexes, la stratégie de sélection est différente: on garde tous les modes.

3 Les valeurs propres du problème se retrouvent sur la diagonale principale de $\mathbf{T}$ et on peut déduire facilement des vecteurs de Schur (les vecteurs colonnes de $\mathbf{Q}$), leurs vecteurs propres associés.

Inconvénients: Complexités mémoire et calcul prohibitives (à réserver aux problèmes de petite taille inférieure à 10^3 degrés de liberté), sensibilités aux différences d'amplitude des termes des matrices.
Variantes: Avec shift implicite ou explicite, simple ou double ...

•Les **méthodes de sous-espace** qui consistent à projeter l'opérateur de travail sur un espace H tel que le spectre de l'opérateur projeté soit une bonne approximation de la partie du spectre initial que l'on recherche. Ces algorithmes sont le noyau dur de l'opérateur MODE_ITER_SIMULT (cf. §5/6/7/8).
Périmètre d'utilisation: **Calcul d'une partie du spectre**.

Avantages: Réduction de la taille du problème et des complexités mémoire et calcul, nécessite seulement le calcul d'un produit matrice-vecteur et non pas la connaissance de toute la matrice.
Inconvénients: Utilise de nombreux pré- et post-traitements, convergence peut devenir problématique, capture plus ou moins facilement les multiplicités et les clusters suivant les variantes.
Variantes: Itérations de sous-espace, Bathe et Wilson(1971), Lanczos(1950), Arnoldi 1951), Davidson (1975), Sorensen(1992), Jacobi-Davidson(1996)...

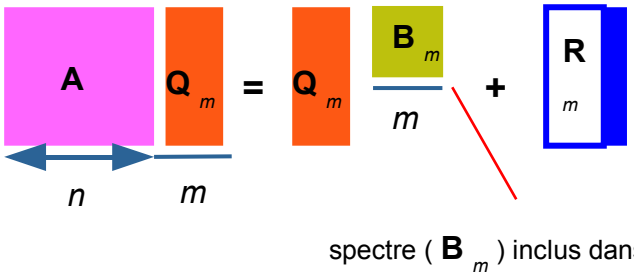


Figure 2.2-2. Schéma fonctionnel des méthodes de type sous-espace. Calcul de p modes en projetant la matrice de travail de taille n sur un espace de taille m ($p<m<n$).

•**Les algorithmes de type puissances** qui ont été historiquement développés les premiers pour résoudre des problèmes modaux génériques. Ce sont des algorithmes de base dont les autres sont une amélioration. Ils sont impliqués dans l'opérateur MODE_ITER_INV (cf. §4).
Périmètre d'utilisation: **Calcul des valeurs extrêmes du spectre**.
Avantages: Simplicité, très bonne estimation du vecteur propre en quelques itérations.
Inconvénients: Convergence peut devenir problématique, mauvaise capture des multiplicités, des clusters...
Variantes: Puissances inverses, (bi) itération du quotient de Rayleigh...

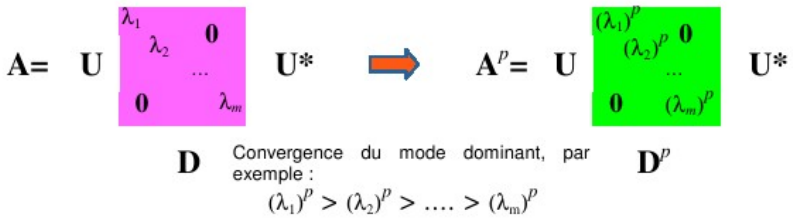


Figure 2.2-3. Schéma fonctionnel des méthodes de type puissance à partir de la décomposition $\mathbf{A}=\mathbf{U}\,\mathbf{D}\,\mathbf{U}^*$ de l'opérateur diagonalisable $\mathbf{A}$ ($\mathbf{U}$ matrice unitaire et $\mathbf{D}$ matrice diagonale des valeurs propres).

•Les **autres approches** sont plus ou moins empiriques et spécialisées. Elles sont souvent reliées à d'autres problématiques: recherche de racines de polynômes, de fonctions quelconques... On peut citer ainsi la méthode de bissection utilisée en pré-traitements dans MODE_ITER_INV (cf. §4), mais aussi celle de Müller-Traub (en QEP), de Jacobi, de Laguerre etc. Elles permettent parfois de traiter directement des GEP et des QEP.

Remarque:

•De nombreux parallèles peuvent être conduits entre ces familles (la méthode QR n'est ainsi qu'une méthode d'itérations de sous-espaces appliquée à l'espace tout entier), mais elles conduisent aussi à des processus analogues à ceux développés pour d'autres problématiques. Ainsi en optimisation: la méthode du quotient de Rayleigh est à la méthode des puissances inverses, ce que la méthode de Newton est pour une méthode de descente classique. Pour la résolution de systèmes linéaires: la méthode du gradient conjugué est une méthode de sous-espace pour les systèmes symétriques définis positifs. Pour la recherche de racines de polynômes : la méthode des puissances est une méthode de Bernoulli appliquée à la matrice «compagnon» du polynôme associé.

2.3 Les librairies d'algèbre linéaire

Pour effectuer efficacement la résolution d'un problème modal, **la question du recourt à une librairie ou à un produit externe est désormais incontournable**. Pourquoi ? Parce que cette stratégie permet:
•Des développements moins techniques, moins invasifs et beaucoup plus rapides dans le code hôte.
•D'acquérir, à moindre frais, un large périmètre d'utilisation tout en externalisant bon nombre des contingences associées (typologie du problème, représentation des données, architecture de la machine cible…).
•De bénéficier du retour d'expérience d'une communauté d'utilisateurs variée et des compétences (très) pointues d'équipes internationales.

Ces librairies conjuguent en effet souvent efficacité, fiabilité, performance et portabilité:
•Efficacité car elles exploitent la localité spatiale et temporelle des données et jouent sur la hiérarchie mémoire (exemple des différentes catégories de BLAS).
•Fiabilité car elles proposent parfois des outils pour estimer l'erreur commise sur la solution (estimation du conditionnement et des 'backward/forward errors') voire pour l'améliorer (par ex. équilibrage matriciel).

Depuis **l'émergence dans les années 1970/1980 des premières librairies publiques**[4] **et constructeurs**[5] et de leurs communautés d'utilisateurs, l'offre s'est démultipliée. La tendance étant bien sûr de proposer des solutions performantes (vectoriel, parallélisme à mémoire centralisé puis distribué, parallélisme multi-niveau *via* des threads) ainsi que des «toolkits» de manipulation d'algorithmes d'algèbre linéaire et des structures de données associées. Citons de manière non exhaustive: ScaLAPACK(Dongarra & Demmel 1997), SparseKIT(Saad 1988), PETSc(Argonne 1991), HyPre(LL 2000), TRILINOS(Sandia 2000)…

Figure 2.3-1. Quelques «logos» de bibliothèques d'algèbre linéaire incluant des solveurs modaux.

Concernant plus spécifiquement les **solveurs modaux** , une trentaine de packages sont disponibles. On distingue les produits «autonomes» de ceux incorporés à une librairie, les publics des commerciaux, ceux traitant des problèmes denses et d'autres des creux. Certains ne fonctionnent qu'en mode séquentiel, d'autres supportent un parallélisme à mémoire partagée et/ou distribuée. Enfin, certains produits sont généralistes (symétrique, non symétrique, réel/complexe, SEP/GEP…) d'autres adaptés à un besoin/scénario bien précis.

On peut trouver une liste assez exhaustive de tous ces produits dans une synthèse commise par l'équipe de SLEPc[HRTV07] ('Scalable Library for Eigenvalue Problem Computations'). Toutefois, elle ne reprend que les solveurs modaux creux du domaine public et oublie de mentionner JADAMILU, LZPACK et PARPACK.

4 EISPACK(1974), LINPACK(1976), BLAS(1978) puis LAPACK(1992)…
5 NAG(1971), IMSL/ESSL(IBM 1971), ASL/MathKeisan(NEC), SciLib(Cray), MKL(Intel/Bull), HSL(Harwell)…

Name	Description	Version	Date	Language	Par
ACPZ	Davidson	1.0	1995	F77	-
ARNCHEB	Arnoldi-Chebyshev	-	1997	F77	-
ARPACK	Arnoldi/Lanczos (implicit restart)	2.1	1995	F77	M
ARPACK++	Arnoldi/Lanczos (implicit restart)	1.1	1998	C++	-
LANCZOS	Lanczos [N]	-	1992	F77	-
LANZ	Lanczos [P]	1.0	1991	F77	-
LASO	Lanczos [S]	2	1983	F77	-
LOPSI	Subspace Iteration	1	1981	F77	-
NAPACK	Power, Lanczos [N]	-	1987	F77	-
PLANSO	Lanczos [P]	1.0	1997	F77	M
QMRPACK	Nonsymmetric Lanczos (lookahead)	-	1996	F77	-
SRRIT	Subspace Iteration	1	1997	F77	-
SVDPACK	SVD via Lanczos [P], Ritzit & Trace Minimization	-	1992	C/F77	-
Underwood	Block Lanczos [F]	-	1975	F77	-

Tableau 2.3-1. Extrait du survey de SLEPc[HRTV07] sur les produits libres implémentant un solveur modal creux. Solveurs déjà anciens et plus maintenus ; 'M' pour MPI, 'O' OpenMP et '-' pour séquentiel;

Type de réorthogonalisation (cf. §6): [F] pour complète, [S] sélective, [P] partielle ou aucune [N].

Name	Description	Version	Date	Language	Par
ANASAZI	Block Krylov-Schur, block Davidson, LOBPCG	7.0.6	2007	C++	M
BLKLAN	Block Lanczos [P]	-	2003	C/Matlab	-
BLOPEX	LOBPCG	-	2004	C/Matlab	M
BLZPACK	Block Lanczos [P+S]	04/00	2000	F77	M
EIGIFP	Inverse-free Krylov subspace method	2.1.1	2004	Matlab	-
IETL	Power, RQI, Lanczos [N]	2.2	2006	C++	-
INSYLAN	Indefinite Symmetric Lanczos	1.0	2000	Matlab	-
IRBLEIGS	Block Lanczos (implicit restart)	1.0	2002	Matlab	-
JDBSYM	Jacobi-Davidson (symmetric)	0.14	1999	C	-
JDCG	Jacobi-Davidson (symmetric)	-	2000	Matlab	-
JDQR/JDQZ	Jacobi-Davidson	-	1998	F77/Matlab	-
MPB	Conjugate Gradient, Davidson	1.4.2	2003	C	M
NA18	Block Davidson	-	1999	F77	-
PDACG	Deflation-accelerated Conjugate Gradient	-	2000	F77	M
PRIMME	Block Davidson, JDQMR, JDQR, LOBPCG	1.1	2006	C/F77	M
PROPACK	SVD via Lanczos [P]	2.1/1.1	2005	F77/Matlab	O
PySPARSE	Jacobi-Davidson (symmetric)	1.0.1	2007	Python	-
SLEPc	Krylov-Schur, Arnoldi, Lanczos, RQI, Subspace	2.3.2	2006	C/F77	M
SPAM	Davidson with SPAM	-	2001	F90	-
TRLAN	Lanczos (dynamic thick-restart)	-	2006	F90	M

Tableau 2.3-2. Extrait du survey de SLEPc[HRTV07] sur les produits libres implémentant un solveur modal creux. Solveurs réçents ou encore maintenus.

2.4 Quelques résultats et benchmarks

Pour attester du bien fondé de son approche, chaque produit procure sur son site web des résultats de runs séquentiels (voire parallèle). Ils sont souvent basés sur des matrices de tests issues de collections publiques (MatrixMarket[MaMa], Université de Floride, Harwell…). Compte-tenu, notamment, de la difficulté de l'exercice et de son fort investissement en temps et en moyens (humain et machine), on trouve assez peu de comparatifs sur les solveurs modaux.

Parmi ces **benchmark, trois ont retenu notre attention**:

- *SEP à matrices réelles symétriques définies positives*[BP02] traités par ARPACK[Arp], JD(Jacobi-Davidson[SV96]) et DACG('Deflation Accelerated Conjugate Gradient'[GSF92]). Etude en séquentiel sur une dizaine de matrices creuses issues de différentes applications (éléments finis, différences finies, éléments finis mixtes…) et de tailles variables: 5.10^3 à 3.10^5 ddls.

Résultats: **pour rechercher un nombre de modes propres inférieur à 40, JD et DACG sont plus efficaces qu'ARPACK. Au delà, ARPACK est plus compétitif**. Sur un problème difficile comportant un cluster de 10 valeurs propres (valeurs propres très proches), seul DACG a fonctionné de manière satisfaisante.

•*GEP à matrices réelles symétriques*[AHLT05] issus de problèmes de mécanique vibratoire de grande taille (supérieure à 10^6 degrés de liberté). Les problèmes sont discrétisés par des EF isoparamétriques et de structure (poutre, coque) en élasticité linéaire. Quatre types de solveurs modaux sont testés: LOBPCG('Locally Optimal block Preconditioned Conjugate Gradient'[Kny91]), DACG, JD et ARPACK. D'un point de vue numérique, l'étude est très fouillée en insistant sur les aspects préconditionneurs, réorthogonalisations et critère de redémarrage. Les calculs ont été menés en séquentiel en utilisant en interne du solveur modal, un solveur linéaire du type GCPC[Boi09b] préconditionné par l'AMG de Trilinos.
Résultats: en temps CPU et en fiabilité, **ARPACK est souvent dépassé par ces nouveaux concurrents**.

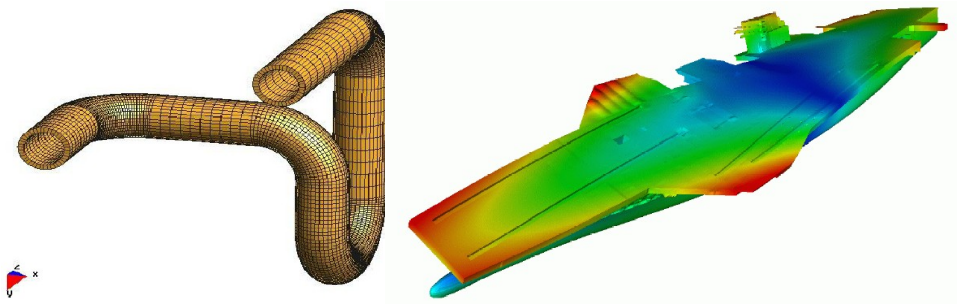


Figure 2.4-1. Exemples de problèmes de mécanique vibratoire utilisés dans les benchmarks[AHLT03]: tube coudé élastique et porte-avions !

•*SEP à matrices réelles non symétriques*[LS96] traités par ARPACK, un Arnoldi accéléré *via* Tchebyschev (code ARNCHEB) et des méthodes de sous-espace (packages LOPSI et SRRIT). Vieille étude en séquentiel sur une vingtaine de matrices de petite taille (10^4 degrés de liberté) issues de différents domaines de la physique.
Résultats: **ARPACK est souvent le meilleur** sur les aspects consommation mémoire, CPU et fiabilité.

Point positif, les méthodes modales disponibles dans *Code_Aster* sont référencées dans les benchmarks (notamment ARPACK). Cependant le code accuse un certain retard par rapport à la recherche actuelle et ne propose pas (pas encore !) les toutes dernières approches:
•Parallélisation des solveurs modaux et distribution des données associée.
•Algorithmes préconditionnés et par blocs (en particulier LOBPCG, JD et DACG).

3 Contexte

3.1 Problématique

Nous considérons le problème modal généralisé (GEP)

Trouver $(\lambda, \mathbf{U})$ tels que $\mathbf{A}\,\mathbf{u} = \lambda\,\mathbf{B}\,\mathbf{u}$, $\mathbf{u} \neq 0$ (3.1-1)

où $\mathbf{A}$ et $\mathbf{B}$ sont des matrices à coefficients réels ou complexes, symétriques ou non (en structure et/ou en valeurs). Ce type de problème correspond, en mécanique, notamment à:

- **L'étude des vibrations libres d'une structure** non amortie et non tournante. Pour cette structure, on recherche les plus petites valeurs propres ou bien celles qui sont dans un intervalle donné pour savoir si une force excitatrice peut créer une résonance. Dans ce cas standard, la matrice $\mathbf{A}$ est la matrice de rigidité, notée $\mathbf{K}$, réelle et symétrique (éventuellement augmentée de la matrice de rigidité géométrique notée $\mathbf{K_g}$, si la structure est précontrainte) et $\mathbf{B}$ est la matrice de masse ou d'inertie notée $\mathbf{M}$ (réelle symétrique). Les valeurs propres obtenues sont les carrés des pulsations associées aux fréquences cherchées.
  Le système à résoudre peut s'écrire: $(\mathbf{K}+\mathbf{K_g})\mathbf{u} = \omega^2 \mathbf{M}\,\mathbf{u}$ où est $\omega = 2\pi f$ la pulsation, f la fréquence propre et $\mathbf{u}$ le vecteur de déplacement propre associé. Les modes propres $(\omega, \mathbf{u})$ sont réels.
  En présence d'amortissement hystérétique, $\mathbf{K}$ devient complexe symétrique. Alors les modes propres sont potentiellement complexes et dépareillés.
  En revanche, si $\mathbf{K}$ et/ou $\mathbf{M}$ restent réelles mais éventuellement non symétriques[6], les modes propres sont soit réels, soit en couple $(\lambda, \bar{\lambda})$.
  Ce type de problématique est activé par le mot-clé `TYPE_RESU='DYNAMIQUE'`.

Calcul vibratoire avec amortissement hystérétique.

Calcul vibratoire non amortie, non tournant,

A / B	Réelle symétrique	Réelle non symétrique	Complexe
Réelle symétrique	**Cas le plus courant** `MULT/INV` sans restriction sur les méthodes; Modes réels.	`SIMULT` avec `'SORENSEN'/'QZ';` Modes réels ou complexes $(\lambda, \bar{\lambda})$.	Cas non traité
Réelle non symétrique	`SIMULT` avec `'SORENSEN'/'QZ';` Modes réels ou complexes $(\lambda, \bar{\lambda})$.	`SIMULT` avec `'SORENSEN'/'QZ'` Modes réels ou complexes $(\lambda, \bar{\lambda})$.	Cas non traité
Complexe symétrique	`SIMULT` avec `'SORENSEN'/'QZ'` Modes réels, complexes quelconques ou $(\lambda, \bar{\lambda})$.	Cas non traité	Cas non traité
Autres complexes (hermitien, non symétrique…)	Cas non traité	Cas non traité	Cas non traité

Tableau 1. Périmètre d'utilisation des opérateurs Aster (`MODE_ITER_SIMULT/INV`) et de leurs méthodes d'analyse (mot-clé `METHODE`) en fonction de propriétés des matrices du GEP.

- **La recherche de mode de flambement linéaire**. Dans le cadre de la théorie linéarisée, en supposant *a priori* que les phénomènes de stabilité sont convenablement décrits par le système d'équations obtenu en supposant la dépendance linéaire du déplacement par rapport au niveau de charge critique, la recherche du mode de flambement $\mathbf{u}$ associé à ce niveau de charge critique λ, se ramène à un

6 On retrouve cette propriété en complexe hermitien.

problème généralisé aux valeurs propres de la forme: $(\mathbf{K}+\lambda\mathbf{K_g})\mathbf{u}=0$ avec $\mathbf{K}$ matrice de rigidité et $\mathbf{K_g}$ matrice de rigidité géométrique. Pour se fondre dans le «moule» d'un calcul de GEP standard, le code calcule, dans un premier temps, les modes propres $(-\lambda, u)$ réels. Puis il les convertit au format d'un calcul de flambement: $(\lambda, \mathbf{u})$.
Ce type de problématique est activé par le mot-clé TYPE_RESU='MODE_FLAMB'. Il est à réserver aux GEPs symétriques réels (sinon le code le détecte et produit une erreur fatale).

Remarques:
•Ces types de GEP sont traités dans Code_Aster par deux opérateurs: MODE_ITER_INV et MODE_ITER_SIMULT. Chacun ayant son périmètre d'application, ses fonctionnalités et ses limitations. Une encapsulation de MODE_ITER_SIMULT, MACRO_MODE_MECA permet d'automatiser et de réduire les coûts (CPU et mémoire) d'une recherche d'une partie importante du spectre (uniquement en GEP à modes réels).
•L'utilisateur peut spécifier la classe d'appartenance de son calcul en initialisant le mot-clé TYPE_RESU à 'DYNAMIQUE' (valeur par défaut) ou à 'MODE_FLAMB'. L'affichage des résultats sera alors formaté en tenant compte de cette spécificité. Dans le premier cas on parlera de fréquence (FREQ) alors que dans le second, on parlera de charge critique (CHAR_CRIT).
•En présence d'amortissements et d'effets gyroscopiques, **l'étude de la stabilité dynamique** d'une structure conduit à la résolution d'un problème modal d'ordre plus élevé, dit quadratique (QEP): $(\mathbf{K}+i\,\omega\mathbf{C}-\omega^2\mathbf{M})\mathbf{u}=0$. Il est résolu par les deux opérateurs modaux et fait l'objet d'une note spécifique[Boi09].

Maintenant que les liens entre la mécanique des structures et la résolution de problèmes modaux généralisés ont été rappelés, nous allons nous intéresser aux traitements des conditions limites dans le code et à leurs incidences sur les matrices de masse et de rigidité.

3.2 Prise en compte des conditions limites

Il y a deux façons, lors de la construction des matrices de rigidité et de masse, de prendre en compte les conditions aux limites (cette description en terme de problème dynamique s'extrapole facilement au flambement):

•La **double dualisation,** en utilisant des degrés de liberté de Lagrange[Pel01], permet de vérifier
$$\mathbf{C}\,\mathbf{u}=0$$ (CLL pour Condition Limite Linéaire),
avec $\mathbf{C}$ matrice réelle de taille $p\times n$ ($\mathbf{K}$ et $\mathbf{M}$ sont d'ordre n). Elle entraîne la manipulation de matrices plus grosses (dites «dualisées») car incorporant ces nouvelles inconnues. Les matrices de rigidité et de masse dualisées ont alors la forme
$$\tilde{\mathbf{K}}=\begin{pmatrix}\mathbf{K} & \beta\mathbf{C}^{\mathbf{T}} & \beta\mathbf{C}^{\mathbf{T}}\\ \beta\mathbf{C} & -\alpha\mathbf{Id} & \alpha\mathbf{Id}\\ \beta\mathbf{C} & \alpha\mathbf{Id} & -\alpha\mathbf{Id}\end{pmatrix}\qquad \tilde{\mathbf{M}}=\begin{pmatrix}\mathbf{M} & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{pmatrix}$$
avec α et β réels strictement positifs (qui servent à équilibrer les termes de la matrice). La dimension du problème a été augmentée de $2p$, car aux n degrés de liberté dits "physiques", on a rajouté des Lagranges. Il y a deux Lagranges par relation linéaire affectée aux p conditions limites.

•**La mise a zéro de p lignes et colonnes des matrices de rigidité et de masse**. Ceci n'est valable que pour des blocages de degrés de liberté (Dirichlet simple, pas de relation de proportionnalité entre ddls). On ne peut pas prendre en compte de relation linéaire et on parlera de blocage cinématique (CLB pour Condition Limite de Blocage). Les matrices de rigidité et de masse deviennent:
$$\tilde{\mathbf{K}}=\begin{pmatrix}\bar{\mathbf{K}} & 0\\ 0 & \mathbf{Id}\end{pmatrix}\qquad \tilde{\mathbf{M}}=\begin{pmatrix}\bar{\mathbf{M}} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{pmatrix}$$
La dimension du problème reste inchangée mais il faut cependant retirer les participations des ddls bloqués aux composantes des matrices initiales ($\bar{\mathbf{K}}$ est obtenue à partir de $\mathbf{K}$ en éliminant les lignes et les colonnes des ddls qui sont bloqués; idem pour $\bar{\mathbf{M}}$).

Lorsqu'on impose des conditions limites, le nombre de valeurs propres (avec toutes leurs multiplicités) réellement impliquées dans la physique du phénomène est donc inférieur à la taille n du problème transformé:

- $n_{ddl-actifs}=n-\frac{3\,p'}{2}$ avec $p'=2\,p$ (double dualisation),
- $n_{ddl-actifs}=n-p$ (blocage cinématique).

L'encadré ci-dessous montre l'affichage dédié à ces paramètres dans le fichier message.

```
-----------------------------------------
LE NOMBRE DE DDL
TOTAL EST:                      220      →  n
DE LAGRANGE EST:                 58      →  p'=2p
LE NOMBRE DE DDL ACTIFS EST :    133     →  n_ddl_actifs
-----------------------------------------
```

Exemple 1. Affichage dans le fichier message de la taille du problème modal.

D'autre part, dans les algorithmes de calcul modal, on doit **s'assurer de l'appartenance des solutions à l'espace admissible**. On s'y ramène *via* des traitements auxiliaires. Ainsi lorsqu'on utilise des blocages cinématiques (CLB), il faut, dans les différents algorithmes et à chaque itération, utiliser un «vecteur de positionnement» $\mathbf{u}_{bloq}$, défini par

- si i ème degré de liberté n'est pas bloqué $\mathbf{u}_{bloq}(i)=1$,
- sinon $\mathbf{u}_{bloq}(i)=0$,

et en pré-multiplier chaque vecteur manipulé

$$\mathbf{u}^1(i)=\mathbf{u}^0(i)\cdot\mathbf{u}_{bloq}(i) \qquad (i=1...\mathrm{n})\Rightarrow\mathbf{u}^1$$

Cette astuce introduit la contrainte des blocages dans tout l'algorithmique et oriente implicitement la recherche de solution dans l'espace admissible.

De même, si on utilise la méthode de double dualisation, on a besoin d'un vecteur de positionnement des degrés de liberté de lagrange $\mathbf{u}_{lagr}$ défini comme $\mathbf{u}_{bloq}$. Il est seulement utilisé lors du choix du vecteur initial aléatoire. Pour que ce vecteur $\mathbf{u}^0$ vérifie les conditions limites (CLL) on opère de la façon suivante:

$$\left|\begin{array}{l}\mathbf{u}^1(i)=\mathbf{u}^0(i)\cdot\mathbf{u}_{lagr}(i)\ \ (i=1...\mathrm{n})\Rightarrow\mathbf{u}^1\\ \tilde{\mathbf{K}}\,\mathbf{u}^2=\mathbf{u}^1\end{array}\right.$$

D'autre part, on inclut souvent la contrainte supplémentaire que ce **vecteur initial appartiennent à l'ensemble image de l'opérateur de travail**. Cela permet d'enrichir plus rapidement le calcul modal en ne se limitant pas au noyau. Ainsi, dans le cas de Lanczos et d'IRAM, on prendra comme vecteur initial, non pas le $\mathbf{u}^2$ précédent, mais $\mathbf{u}^3$ tel que

$$\mathbf{u}^3=(\mathbf{K}-\sigma\,\mathbf{M})^{-1}\,\mathbf{M}\,\mathbf{u}^2$$

Par la suite, pour simplifier les notations, nous ne ferons pas le *distinguo* entre les matrices initiales et leurs pendants dualisés (notés avec un tilde) que si nécessaire. Bien souvent, elles seront désignées par $\mathbf{A}$ et $\mathbf{B}$ afin de se rapprocher de la notation modale usuelle sans se rattacher à telle ou telle classe de problèmes.

3.3 Propriétés des matrices

Dans le cas (le plus courant) où les matrices considérées sont **symétriques et à coefficients réels**, on répertorie les cas de figure décrits dans le tableau ci-dessous. Les matrices peuvent être définies positives (noté >0), semi-définies positives (≥ 0), indéfinies (≤ 0 ou ≥ 0) voire singulières (S).

	Structure libre	Lagranges [7]	Flambement	Fluide-structure
$\mathbf{A}(\mathbf{K})$	≥ 0 et S	<0 ou >0	>0	≥ 0 et S

	Structure libre	Lagranges	Flambement	Fluide-structure
$\mathbf{B}$ (resp. $\mathbf{M}$ ou $\mathbf{K_g}$)	>0	≥ 0 et S	≤ 0 ou ≥ 0	>0

Tableau 3.3-1. Propriétés des matrices du GEP.

Les colonnes de ce tableau s'excluant mutuellement, en pratique, un problème de flambement utilisant des doubles Lagranges pour modéliser certaines de ses conditions limites, voit ses matrices dualisées ($\tilde{\mathbf{K}}$ et $\tilde{\mathbf{K}}_{\mathbf{g}}$) devenir potentiellement indéfinies.

Cet éventail de propriétés doit être pris en compte lors du choix du couple «(opérateur de travail, produit scalaire)». Ce cadre peut ainsi renforcer, avec efficacité et transparence, la robustesse et le périmètre de l'algorithme de calcul modal dans tous les cas de figure rencontrés par *Code_Aster*.

Dans le cas où les **matrices sont complexes symétriques ou réelles non symétriques**, on ne peut plus définir de produit scalaire matriciel. Seules alors sont disponibles les méthodes QZ (§9) et IRAM(§7). La première n'a pas besoin de ce type de mécanisme et, on «bluffe» la seconde en lui fournissant un «faux» produit-scalaire matriciel, en fait le produit scalaire euclidien usuel (cf. §7.5). Cette dernière astuce est licite avec IRAM, car en tant que variante d'Arnoldi, elle peut fonctionner avec un couple (opérateur de travail,produit-scalaire) non symétrique.

Les paragraphes suivants vont nous permettent de mesurer l'incidence de ces propriétés sur le spectre de problème généralisé.

3.4 Propriétés des modes propres

Rappelons tout d'abord que si la matrice du SEP $\mathbf{A}\,\mathbf{u}=\lambda\,\mathbf{u}$ est réelle symétrique, alors ses éléments propres sont réels; Les éléments propres d'une matrice sont ses valeurs et ses vecteurs propres. D'autre part, $\mathbf{A}$ étant normale, ses vecteurs propres sont orthogonaux.

Dans le cas du GEP $\mathbf{A}\,\mathbf{u}=\lambda\,\mathbf{B}\,\mathbf{u}$, cette condition n'est pas suffisante. Ainsi, considérons le problème généralisé suivant:

$$\begin{bmatrix}1 & 1\\ 1 & 0\end{bmatrix}\begin{pmatrix}u_1\\ u_2\end{pmatrix}=\lambda\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\begin{pmatrix}u_1\\ u_2\end{pmatrix}$$

ses modes propres sont

$$\lambda_{\pm}=\frac{1}{2}(1\pm i\sqrt{(3)}) \text{ et } \mathbf{u}_{\pm}=\frac{1}{\sqrt{1+\lambda_{\pm}^2}}\begin{pmatrix}-\lambda_{\pm}\\ 1\end{pmatrix}$$

Si on ajoute l'hypothèse "une des matrices $\mathbf{A}$ ou $\mathbf{B}$ est **définie positive**", alors le problème généralisé a ses **solutions réelles**. On a même la caractérisation (condition suffisante) plus précise suivante.

Théorème 1
Soient $\mathbf{A}$ et $\mathbf{B}$ deux matrices symétriques réelles. S'il existe $\alpha\in\mathbb{R}$ et $\beta\in\mathbb{R}$ tels que $\alpha\,\mathbf{A}+\beta\,\mathbf{B}$ soit définie positive, alors le problème généralisé a ses éléments propres réels.

Preuve:
Ce résultat s'obtient immédiatement en multipliant le problème généralisé par α et en effectuant un décalage spectral β . On obtient alors le problème $(\alpha\,\mathbf{A}+\beta\,\mathbf{B})\mathbf{u}=(\lambda\alpha+\beta)\,\mathbf{B}\,\mathbf{u}$. Comme est définie $\alpha\,\mathbf{A}+\beta\,\mathbf{B}$ positive, elle admet une décomposition de Cholesky unique sous la forme $\mathbf{C}\mathbf{C}^{\mathbf{T}}$ avec $\mathbf{C}$ matrice régulière.

7 Cette colonne concerne bien sûr les propriétés des matrices dualisées constituées à partir des matrices initiales.

Le problème s'écrit alors $\mathbf{C^{-1}BC^{-T}z}=\mu\,\mathbf{z}$ avec $\mathbf{z}=\mathbf{C^{T}u}$ et $\mu=\frac{1}{\alpha\,\lambda+\beta}$, ce qui permet de conclure, car la matrice $\mathbf{C^{-1}BC^{-T}}$ est symétrique.

Remarques
•Cette caractérisation n'est pas nécessaire, ainsi le problème généralisé associé aux matrices $\mathbf{A}=diag(1,-2,-1)$ et $\mathbf{B}=diag(-2,1,1)$ admet un spectre réel tout en ne répondant pas à la condition de définie positivité.
•Dans le cas de matrices complexes et hermitienne, ce théorème reste valide. Par contre, en complexe non hermitien (c'est le cas lors de la prise en compte de l'amortissement hystérétique dans Code_Aster), les modes propres peuvent être complexes: vecteurs propres à composantes complexes et valeurs propres λ réelles ou complexes quelconques.
•En réel non symétrique, les modes propres peuvent être complexes: vecteurs propres à composantes complexes et valeurs propres λ réelles ou complexes par couple $(\lambda,\bar{\lambda})$.

Proposition 2
Si les matrices $\mathbf{A}$ et $\mathbf{B}$ sont réelles et symétriques, les vecteurs propres du problème généralisé sont $\mathbf{A}$ et $\mathbf{B}$ - orthogonaux, ce qui signifie qu'ils vérifient les relations

$$\begin{cases}\mathbf{u_i^T B\,u_j}=\delta_{ij}\,a_j\\ \mathbf{u_i^T A\,u_j}=\lambda_j\,\delta_{ij}\,a_j\end{cases}$$

où a_j est un scalaire dépendant de la norme du j ième vecteur propre, δ_{ij} est le symbole de Kronecker et $\mathbf{u_j}$ est le vecteur propre associé à la valeur propre λ_j .

Preuve:
Immédiate pour des valeurs propres distinctes, en écrivant les $\mathbf{A}$ et $\mathbf{B}$ - produit scalaire entre deux couples (i,j) et (j,i) , puis en utilisant la symétrie des matrices (cf. [Imb91]).

Remarques:
•On montre que les $\mathbf{A}$ et $\mathbf{B}$ - orthogonalités des vecteurs propres sont une conséquence de l'hermiticité des matrices. Elles sont clairement une généralisation des propriétés du problème standard hermitien (voire normaux): dans le cas d'une matrice à coefficients complexe et hermitienne, le produit scalaire à considéré est un produit hermitien.
•L'orthogonalité par rapport aux matrices ne signifie surtout pas que les vecteurs propres sont orthogonaux pour la norme euclidienne classique. Celle-ci ne peut être que le fruit de symétries particulières (cf. TP n°1 [BQ00]).
•Cette propriété simplifie les calculs de recombinaisons modales (DYNA_TRAN_MODAL[Boy07]), lorsqu'on manipule des matrices de rigidité et de masse généralisées qui sont diagonales. Les quantités j $k_j=\lambda_j\,a_j$ et $m_j=a_j$ sont appelées, respectivement, rigidité modale et masse modale du j ième mode.
•Pour les matrices non hermitiennes, le théorème 1 n'est plus vérifié.
•
Sachant que les modes sont souvent réels, nous allons maintenant nous préoccuper de leur estimation.

3.5 Estimation du spectre réel

La doc. R5.01.04 est dédiée à cette problématique transverse à bon nombre d'opérateurs modaux: MODE_ITER_SIMULT/INV, INFO_MODE ainsi que leurs macros: MACRO_MODE_MECA et CALC_MODAL.

Rappelons juste que dans le cas le plus courant de modes réels (GEP réel symétrique), le problème du comptage de valeurs propres se résout à l'aide du fameux test de Sturm (cf. §2.2/3.2). La situation est beaucoup moins favorable lorsque le spectre réside dans le plan complexe (GEP complexe ou non symétrique et QEP). Dans ce cas, seul l'opérateur INFO_MODE dispose d'une méthode adaptée: la méthode APM (cf. §2.3/3.3). Mais du fait de ses énormes coûts calcul et de son caractère novateur, il est conseillé de la réserver, pour l'instant, aux problèmes simplifiés de petite taille (<10^4 degrés de liberté).

Maintenant que nous sommes en mesure de comptabiliser le spectre du GEP, il reste à le construire ! Les algorithmes génériques étant destinés aux SEP, il faut transformer notre problème initial.

3.6 Transformation spectrale

Ces techniques permettent de répondre à un triple objectif:
•identifier un SEP,
•Orienter la recherche du spectre,
•Séparer les valeurs propres.

Les algorithmes de calcul spectral convergeant d'autant mieux que le spectre (de travail) qu'ils traitent est séparé, ces techniques peuvent être considérées comme des préconditionnement du problème de départ. Elles permettent de rendre la séparation de certains modes beaucoup plus importante que celles d'autres modes, et d'améliorer ainsi leur convergence.

La plus répandue de ces transformations est la technique dite de **"shift and invert"** qui consiste à travailler avec l'opérateur $\mathbf{A}_\sigma$ tel que:

$$\mathbf{Au}=\lambda\,\mathbf{Bu}\Rightarrow\underbrace{\left(\mathbf{A}-\sigma\,\mathbf{B}\right)^{-1}}_{\mathbf{A}_\sigma}\mathbf{B}\,\mathbf{u}=\underbrace{\frac{1}{\lambda-\sigma}}_{\mu}\mathbf{u}$$

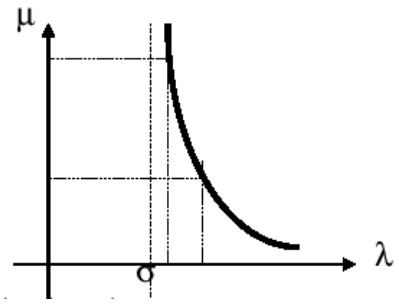


Figure 3.6-1. Effet du "shift and invert" sur la séparation des valeurs propres.

La figure 3.6-1 montre que cette séparation et cette orientation du spectre de travail en μ sont dues aux propriétés particulières de la fonction hyperbolique. D'autre part, on observe que seules les valeurs propres sont affectées par la transformation. En fin de processus modal il suffit donc de repasser dans le plan des λ par un changement de variable idoine.

Remarques:
•La variable σ est usuellement désignée par le terme de «shift» ou de décalage spectral.
•La matrice de travail $\mathbf{A}_\sigma$ doit bien sûr être inversible, cela peut d'ailleurs devenir une des motivations de ce décalage (cf. §6.5).

Pour mémoire, notons qu'avec un shift complexe plusieurs scénarios sont envisageables:
•Travailler complètement en arithmétique complexe,
•En arithmétique réelle, en isolant les contributions réelles et imaginaires de $\mathbf{A}_\sigma$, par exemple via les opérateurs de travail

$$A_\sigma^+=\mathrm{Re}\left(A_\sigma\right)\Rightarrow\mu^+=\frac{1}{2}\left(\frac{1}{\lambda-\sigma}+\frac{1}{\lambda-\bar{\sigma}}\right),$$
$$A_\sigma^-=\mathrm{Im}\left(A_\sigma\right)\Rightarrow\mu^-=\frac{1}{2\mathrm{i}}\left(\frac{1}{\lambda-\sigma}-\frac{1}{\lambda-\bar{\sigma}}\right).$$

Chacune de ses approches a ses avantages et ses inconvénients. Pour les QEP de *Code_Aster*[Boi09], c'est la première démarche qui a été retenue pour Sorensen (METHODE='SORENSEN'+APPROCHE='COMPLEXE'). La seconde est réservée aux autres approches: METHODE='SORENSEN' ou 'TRI_DIAG' + APPROCHE='REEL'/'IMAGINAIRE').

Remarques

•Ce choix de l'opérateur de travail est indissociable de celui du (pseudo)-produit scalaire. Il permet de s'orienter vers tel ou tel algorithme et peux ainsi influer sur la robustesse du calcul.
•D'autres classes de transformations spectrales existent. Par exemple celle de Cayley, avec un double shifts ($\sigma_{1,}\sigma_2$), permet de sélectionner les valeurs propres situées à droite d'un axe vertical

$$\underbrace{(\mathbf{A}-\sigma_1\mathbf{B})^{-1}(\mathbf{A}-\sigma_2\mathbf{B})}_{A_\sigma}\mathbf{u}=\underbrace{\frac{\lambda-\sigma_2}{\lambda-\sigma_1}}_{\mu}\mathbf{u}$$

Le paragraphe suivant va synthétiser ce qui précède dans l'organigramme global de résolution d'un problème modal généralisé de *Code_Aster*.

3.7 Implantation dans *Code_Aster*

Le déroulement d'un calcul modal dans *Code_Aster* peut se décomposer en quatre phases. Les paragraphes suivant les détaillent.

3.7.1 Détermination du shift

La première opération consiste à déterminer le shift ainsi que certains paramètres du problème. Cela s'effectue de manière plus ou moins transparente suivant l'option de calcul choisie par l'utilisateur:
•MODE_ITER_INV+OPTION='SEPARE' ou 'AJUSTE' => le shift est déterminé par la première phase de l'algorithme et le nombre de modes propres recherchés par bandes fréquentielles (fourni par le critère de Sturm) est borné par NMAX_FREQ.
•MODE_ITER_INV+OPTION='PROCHE' => le shift est fixé par l'utilisateur et le nombre de modes propres est égal au nombre de shifts.
•MODE_ITER_SIMULT+OPTION='PLUS_PETITE' => le shift est nul et le nombre de modes est paramétré par NMAX_FREQ.
•MODE_ITER_SIMULT+OPTION='BANDE' => le shift est égal au milieu de la bande fixée par l'utilisateur et le nombre de modes est déterminé par le critère de Sturm.
•MODE_ITER_SIMULT+OPTION='CENTRE' => le shift est fixé par l'utilisateur et le nombre de modes propres est paramétré par NMAX_FREQ.

Remarques
•Cette phase ne concerne pas l'algorithme QZ de MODE_ITER_SIMULT. En effet ce dernier calcule tout le spectre du GEP. C'est juste après le calcul modal que l'on tient compte des desiderata de l'utilisateur (OPTION='PLUS_PETITE'/'BANDE'/'CENTRE'/'TOUTE') pour sélectionner les modes recherchés.
•Pour une GEP à modes complexes ($\mathbf{K}$ complexe symétrique ou matrices non symétriques), seules les options 'PLUS_PETITE'/'CENTRE'/'TOUTE' avec MODE_ITER_SIMULT sont disponibles (avec Sorensen et QZ).

3.7.2 Factorisation de matrices dynamiques

Dans la seconde phase de pré-traitements, on factorise[Boi08] **des matrices «dynamique» du type** $\mathbf{Q}(\sigma):=\mathbf{A}-\sigma\mathbf{B}$ **.** C'est-à-dire qu'on décompose la matrice dynamique $\mathbf{Q}(\sigma)$ en un produit de matrices particulières (triangulaire, diagonale) plus faciles à manipuler pour résoudre des systèmes linéaires du type $\mathbf{Q}(\sigma)\mathbf{x}=\mathbf{y}$. En symétrique, la décomposition est de la forme $\mathbf{Q}(\sigma)=\mathbf{L}\mathbf{D}\mathbf{L}^{\mathbf{T}}$, en non symétrique, on a $\mathbf{Q}(\sigma)=\mathbf{L}\mathbf{U}$ **,** avec $\mathbf{U}$, $\mathbf{L}$ et $\mathbf{D}$, respectivement, triangulaire supérieure, inférieure et diagonale.
Une fois cette factorisation numérique (coûteuse) effectuée, la résolution d'autres systèmes linéaires comportant la même matrice mais avec un second membre différent (problème de type multiple seconds membres) est très rapide.

Ce cas de figure se retrouve:
•A chaque fois que l'on met en œuvre le **test de Sturm**. C'est-à-dire pour certains scénarios de la phase 1 de sélection du shift (MODE_ITER_INV+OPTION='SEPARE'/'AJUSTE',

MODE_ITER_SIMULT+OPTION='BANDE') et en prévision de la phase 4 de post-vérification lors du comptage des valeurs propres réelles (GEP à matrices réelles et symétriques uniquement).

•Lorsqu'on doit manipuler une matrice de travail comportant un inverse. Par exemple dans le cas de Lanczos ou d'IRAM (MODE_ITER_SIMULT+METHODE='TRI_DIAG'/'SORENSEN'), on s'intéresse à $\mathbf{A}_{\sigma}=\left(\mathbf{A}-\sigma\mathbf{B}\right)^{-1}\mathbf{B}$. Avec la méthode des puissances inverses (MODE_ITER_INV), il s'agit de $\mathbf{A}^{\sigma}=\left(\mathbf{A}-\sigma\mathbf{B}\right)^{-1}$. Les deux autres approches, Bathe & Wilson ou QZ (MODE_ITER_SIMULT+METHODE='JACOBI'/'QZ') ne sont pas concernées par cette factorisation préliminaire d'une matrice de travail.

Remarques
•Cette factorisation subit d'ailleurs les mêmes aléas que le critère de Sturm lorsque le shift est proche d'une valeur propre. On procède alors aux même décalages suivant l'algorithme 1 de R5.01.04.
•Lorsque cette phase de pré-traitements met en œuvre l'algorithme 5 de R5.01.04 et que NMAX_ITER_SHIFT est atteint le calcul produit une alarme s'il s'agit d'une factorisation pour le test de Sturm (risque d'instabilités numériques), une erreur fatale si on cherche à factoriser la matrice de travail (risque de résultats faux).
•Lorsqu'on utilise l'algorithme QZ sur des GEP non symétriques ou complexes, on n'a pas besoin de factoriser de matrice dynamique. Ce distinguo permet d'économiser un peu de complexité calcul, l'algorithme QZ étant déjà assez coûteux en-soi !
•En chaînant les opérateurs INFO_MODE[U4.52.01] et MODE_ITER_SIMULT, on peut mieux orienter/calibrer son calcul spectral voire limiter le surcoût qu'engendre le test de Sturm initial de l'option 'BANDE' (cf. mot-clé TABLE_FREQ/CHAR_CRIT[U4.52.03]).

3.7.3 Calcul modal

On effectue le calcul modal proprement dit: on résout le problème standard SEP, puis on revient au GEP initial. Lors de cette conversion, on filtre et transcrit les résultats du calcul précédent.
Concernant les valeurs propres, on retient toutes les solutions calculées: réelles, complexes seules ou en couple.
Concernant les vecteurs propres, on force leur $\mathbf{B}$ -orthonormalisation uniquement en GEP symétrique réels (sinon on ne peut pas définir de $\mathbf{B}$ -produit scalaire).

Dans MODE_ITER_INV deux variantes sont disponibles: la méthode des itérations inverses ('DIRECT') et son accélération par quotient de Rayleigh ('RAYLEIGH'). Elles affinent les valeurs propres préalablement détectées par les heuristiques ('SEPARE' ou 'AJUSTE') ou les estimations fournies par l'utilisateur ('PROCHE'). Pour ce qui est de MODE_ITER_SIMULT, il permet l'usage de quatre méthodes distinctes: la méthode de Bathe & Wilson ('JACOBI'), les méthodes de sous-espace de type Lanczos ('TRI_DIAG') et Sorensen ('SORENSEN') et la méthode globale QZ ('QZ').

Remarque
•*Chacune de ces méthodes possède des tests d'arrêts internes. Sans compter que les méthodes de projection emploient des méthodes modales auxiliaires: Jacobi (cf. Annexe 3) pour 'Bathe & Wilson' et* $\mathbf{QR}/\mathbf{QL}$ *(cf. Annexe 1) pour Lanczos et Arnoldi. Elles nécessitent aussi des tests d'arrêts. L'utilisateur a souvent accès à ces paramètres, bien qu'il soit chaudement recommandé, au moins dans un premier temps, de conserver leurs valeurs par défaut.*

3.7.4 Post-traitements de vérification

Cette dernière partie regroupe les **post-traitements qui vérifient le bon déroulement du calcul**. Ils sont de deux types:

•**Norme[8] du résidu du problème initial**

8 On rappelle que $\|\mathbf{u}\|_{\infty}=\max_i|\mathbf{u_i}|$, $\|A\|_{\infty}=\max_i\sum_j|A_{ij}|$, $\|u\|_2=\left(\sum_i|u_i|^2\right)^{\frac{1}{2}}$ et $\|\mathbf{A}\|_2=\max_i|\lambda_i(\mathbf{AA^*})|^{\frac{1}{2}}\left(=\max_i|\lambda_i(\mathbf{A})|\text{ si }\mathbf{A}\text{ hermitien}\right)$.

$$u \Leftarrow \frac{u}{\|u\|_\infty}$$
Si $|\lambda| >$ SEUIL_FREQ alors
$$\frac{\|\mathbf{Au} - \lambda \mathbf{Bu}\|_2}{\|\mathbf{Au}\|_2} \; ?< \text{SEUIL},$$
Sinon
$$\|\mathbf{Au} - \lambda\, \mathbf{Bu}\|_2 \; ?< \text{SEUIL},$$
Fin si.

Algorithme 1. Test de la norme du résidu.

Cette séquence est paramétrée par les mot-clés SEUIL et SEUIL_FREQ, appartenant respectivement aux mot-clés facteur VERI_MODE et CALC_FREQ. D'autre part, ce post-traitement est activé par l'initialisation à 'OUI' (valeur par défaut) de STOP_ERREUR dans le mot-clé facteur VERI_MODE. Lorsque cette règle est activée et non-respectée, le calcul s'arrête, sinon l'erreur est juste signalée par une alarme. On ne saurait bien sûr que trop recommander de ne pas désactiver ce paramètre passe-droit !

•Comptage des valeurs propres

Ce post-traitement est mis en place uniquement dans le cas de matrices réelles symétriques et il n'est activé que pour l'opérateur MODE_ITER_SIMULT (et ses macros CALC_MODAL et MACRO_MODE_MECA). Dans ce cadre, il permet de vérifier que le nombre de valeurs propres contenues dans une bande test $\left[\sigma_1, \sigma_2\right]$ est égal au nombre détecté par l'algorithme. Cette procédure de comptage s'active en deux temps: détermination de la bande test (cf. figure et algorithme ci-dessous) puis calcul de Sturm proprement dit. Ce calcul de Sturm fait l'objet d'une documentation spécifique (cf. [R5.01.04]), on ne détaille donc ici que l'étape préliminaire qui est spécifique aux post-traitements de MODE_ITER_SIMULT.

L'inclusion de la bande initiale[9] dans $\left[\sigma_i, \sigma_f\right]$ est conduite afin de détecter d'éventuels problèmes de clusters ou de multiplicités aux bornes initiales.

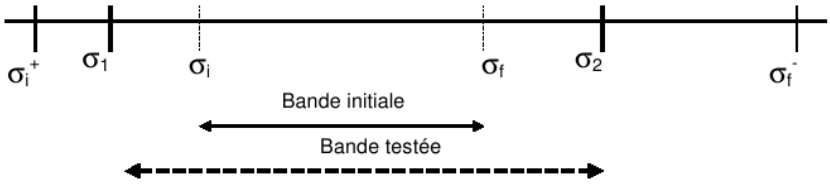


Figure 3.7-1. Comptage des valeurs propres par le test de Sturm en GEP réel symétrique.

En notant σ_i^+ et σ_f^-, respectivement, la plus grande et la plus petite valeur propre non demandée par l'utilisateur et englobant la bande initiale (cf. figure 3.7-1), on a l'algorithme de construction de la bande test suivant:

9 En option 'BANDE' il s'agit des valeurs renseignées par l'utilisateur, sinon ce sont les valeurs extrêmes du spectre calculé.

Si σ_i^+ existe $\left(\text{resp. } \sigma_f^-\right)$ et si $\frac{|\sigma_i^+ - \sigma_i|}{|\sigma_i|} < \text{PREC\_SHIFT}$ $\left(\text{resp. } \frac{|\sigma_f^- - \sigma_f|}{|\sigma_f|}\right)$

alors $\sigma_1 = \frac{\sigma_i^+ + \sigma_i}{2}$ $\left(\text{resp. } \sigma_2 = \frac{\sigma_f^- + \sigma_f}{2}\right)$,

Sinon

$\sigma_1 = \sigma_i\left(1 - \text{sign}\left(\sigma_i\right)\text{PREC\_SHIFT}\right)$ $\left(\text{resp. } \sigma_2 = \sigma_f\left(1 + \text{sign}\left(\sigma_f\right)\text{PREC\_SHIFT}\right)\right)$,

Fin si.

Algorithme 2. Construction de la bande test pour le test de Sturm.

Cette séquence de construction de la bande test, *via* les modes calculés et non demandés, n'est active que pour les méthodes de Lanczos et de Bathe & Wilson de MODE_ITER_SIMULT. Avec les méthodes de Sorensen et QZ, on ne sélectionne que les modes demandés et donc les bornes de l'intervalle de test sont déterminées par la seconde partie de l'algorithme n°2.
Cet algorithme est paramétrée par le mot-clé PREC_SHIFT du mot-clé facteur VERI_MODE. L'encadré ci-dessous affiche la trace des messages d'erreurs lorsque ces post-vérifications se déclenchent.

```
VERIFICATION A POSTERIORI DES MODES
<E> <MODE_ITER_SIMULT> <VERIFICATION DES MODES> POUR LE CONCEPT >MOD_4< LE MODE NUMERO 2 DE
CHARGE CRITIQUE 8.7243E+07 A UNE NORME D'ERREUR
DE 6.3919E-01

<E> <MODE_ITER_SIMULT> <VERIFICATION DES MODES> POUR LE CONCEPT >MOD_4< DANS L'INTERVALLE
(-1.0608E+08,1.8130E+08) IL Y A THEORIQUEMENT 3 CHARGE(S) CRITIQUE(S) ET L'ON EN A CALCULEE(S)
4.
```

Exemple 3. Impressions dans le fichier message des post-vérifications.

La procédure de comptage est désactivable *via* le mot-clé VERI_MODE/STURM. De même, lorsque au moins une des deux étapes de post-vérification (norme du résidu ou procédure de comptage) n'est pas respectée, la suite des événements est pilotée par STOP_ERREUR (si 'OUI' le calcul s'arrête, sinon l'erreur est juste signalée par une alarme). On ne saurait bien sûr que trop recommander de ne pas désactiver ces paramètres 'passe-droit' !

3.7.5 Affichage dans le fichier message

Dans le fichier message sont mentionnées les informations relatives aux modes retenus. Par exemple, dans le cas le plus courant d'un GEP symétrique réel (modes propres réels), on précise le solveur modal utilisé, la liste des fréquences f_i retenues (FREQUENCE) et leur norme d'erreur (NORME D'ERREUR, cf. algorithme n°1).

```
-------------------------------------------------
      CALCUL MODAL:  METHODE GLOBALE DE TYPE QR
                     ALGORITHME QZ_SIMPLE

   NUMERO     FREQUENCE (HZ)     NORME D'ERREUR
        1      1.67638E+02       2.47457E-11
        2      1.67638E+02       1.48888E-11
        3      1.05060E+03       2.00110E-12
        4      1.05060E+03       1.55900E-12
.....
```

Exemple 4a. Impressions dans le fichier message des valeurs propres retenues lors de GEP à modes réels. Extrait du cas-test forma11a.

Pour un GEP à modes complexes (non symétrique réel ou complexe symétrique[10]), contrairement aux QEP[Boi09], *Code_Aster* ne filtre pas les valeurs propres conjuguées des complexes quelconques. Il les

10 Le cumul des deux, non symétrique complexe ou complexe quelconque n'est pas traité actuellement.

conserve toutes et les affiche par ordre croissant de partie réelle. Ainsi, les colonnes FREQUENCE et AMORTISSEMENT regroupent, respectivement, $\mathrm{Re}(f_i)$ et $\frac{\mathrm{Im}(f_i)}{2\,\mathrm{Re}(f_i)}$.

```
-------------------------------------------------------------------------
      CALCUL MODAL:  METHODE D'ITERATION SIMULTANEE
                     METHODE DE SORENSEN

   NUMERO     FREQUENCE (HZ)     AMORTISSEMENT     NORME D'ERREUR
        1       5.93735E+02        5.00000E-03       2.73758E-15
        2       9.45512E+02        5.00000E-03       1.64928E-15
        3       3.51464E+03        5.00000E-03       2.14913E-15
.....
```

Exemple 4b. Impressions dans le fichier message des valeurs propres retenues lors de GEP à modes complexes. Extrait du cas-test zzzz208a.

Maintenant que le contexte des GEPs dans *Code_Aster* a été brossé, nous allons nous intéresser plus particulièrement aux méthodes de type puissance et à leur implantation dans l'opérateur MODE_ITER_INV.

4 Méthode des puissances inverses (MODE_ITER_INV)

4.1 Introduction

Cette méthode n'est accessible dans *Code_Aster* que pour le cas des matrices à coefficients réels. On se limitera donc à ce cas dans la suite de ce chapitre. Pour calculer plusieurs valeurs propres du problème généralisé, on **n'utilise pas la méthode générale des itérations inverses telle quelle**.

On peut, par exemple, la combiner à une **technique de déflation**[11] de manière à filtrer automatiquement l'information spectrale mise à jour pour ne plus la retrouver à l'itération suivante. Avec la déflation par restriction[12] de Wielandt on construit itérativement l'opérateur de travail (dans le cas symétrique), en notant $(\mu_k, \mathbf{u}_k)$ le mode à filtrer

$$\mathbf{A}_{k+1} = \mathbf{A}_k - \mu_k \mathbf{u}_k \mathbf{u}_k^t \ .$$

Cette stratégie ,qui n'appréhende pas les multiplicités, doit être complétée par un critère de Sturm. D'autre part, le fait de travailler en arithmétique finie et de ne pas construire effectivement l'opérateur $\mathbf{A}_k$ à chaque itération, contraint à mâtiner cette démarche de processus d'orthogonalisation performants.

Toutes ces complications ont conduit à choisir une autre voie, qui se décompose en deux parties:
- La *localisation des valeurs propres* (détermination d'une valeur approchée de chaque valeur propre contenue dans un intervalle donné par une **technique de bissection**, affinée ou non, par une **méthode de la sécante**).
- *L'amélioration de ces estimations et le calcul de leurs vecteurs propres* associés par une **méthode d'itérations inverses**.

La recherche d'une valeur approchée pour chaque valeur propre considérée est sélectionnée dans le mot-clé facteur CALC_FREQ *via* OPTION:
- Si OPTION='SEPARE', dans chaque intervalle de fréquences définies par le mot-clé FREQ, une valeur approchée de chaque valeur propre contenue dans cet intervalle est calculée en utilisant la méthode de dichotomie (cf. §4.2.1).
- Si OPTION='AJUSTE', on effectue tout d'abord les mêmes opérations que précédemment et ensuite, partant de ces approximations, on affine le résultat par la méthode de la sécante.
  Pour ces deux options, on calcule en même temps la position modale de chaque valeur propre ce qui permet de détecter les modes multiples. Soit on ne retient que les NMAX_FREQ fréquences les plus basses contenues dans l'intervalle maximal spécifié par l'utilisateur, soit on calcule toutes les valeurs de cet intervalle (si NMAX_FREQ=0).
- Si OPTION='PROCHE', les fréquences données par le mot-clé FREQ, sont considérées comme les valeurs approchées des valeurs propres cherchées.

Remarques:
- Bien sûr, comme on l'a déjà précisé, cet opérateur n'est à utiliser que pour déterminer ou affiner quelques valeurs propres. Pour une recherche plus étendue il faut utiliser l'opérateur MODE_ITER_SIMULT voire MACRO_MODE_MECA (uniquement en GEP symétrique réel).
- Avec l'option 'PROCHE' on ne peut pas calculer de modes multiples.
- C'est un algorithme coûteux car il fait beaucoup appel au test de Sturm et donc à ses factorisations associées.
- Les bornes des intervalles de recherche sont fournies par FREQ ou CHAR_CRIT suivant l'initialisation de TYPE_RESU.
-

Nous allons maintenant détailler les différents algorithmes (et leurs paramétrages) qui sont mis en place dans la première partie du processus.

4.2 Localisation et séparation des valeurs propres

11 Technique de filtrage consistant à transformer l'opérateur de travail de telle sorte qu'il possède les mêmes valeurs propres sauf certains modes prédéfinis qui se voient affecter une valeur nulle.

12 D'autres types de déflation existent, telle que la méthode de Ducan-Collar qui utilise, pour filtrer l'information spectrale, la première ligne de la matrice et le vecteur propre. Ces techniques vectorielles se généralisent bien sûr par blocs.

4.2.1 Méthode de bissection

Comme on l'a déjà vu précédemment, le corollaire 5bis de la loi d'Inertie de Sylvester (cf. §3.5) permet de déterminer le nombre de valeurs propres contenues dans un intervalle donné en effectuant deux décompositions $\mathbf{LDL}^{\mathbf{T}}$. Ce critère peut donc conduire à raffiner l'intervalle jusqu'à n'englober qu'une valeur propre. Ce pilotage étant mis en place, on passe d'une itération à l'autre en utilisant le **principe de la dichotomie**.

Sur un intervalle de départ $[\lambda_a, \lambda_b]$, on opère donc de la façon suivante, connaissant $\mathrm{pm}(\lambda_a)$ et $\mathrm{pm}(\lambda_b)$:

Calcul de $\lambda^* = \frac{1}{2}(\lambda_a + \lambda_b)$.
Calcul de $\mathrm{pm}(\lambda^*)$.
Si $\mathrm{pm}(\lambda^*) = \mathrm{pm}(\lambda_a)$ $(\text{resp. } \mathrm{pm}(\lambda_b))$ alors
  recommencer sur $[\lambda^*, \lambda_b]$ $(\text{resp. } [\lambda_a, \lambda^*])$,
Sinon
  recommencer sur $[\lambda_a, \lambda^*]$
  et sur $[\lambda^*, \lambda_b]$,
Fin si.

Algorithme 3. Méthode de Bissection.

On arrête le processus si on a découpé plus de NMAX_ITER_SEPARE fois l'intervalle de départ, ou si pour un intervalle donné, on a ($\left(\frac{|\lambda_a - \lambda_b|}{\lambda^*}\right) \leq \text{PREC\_SEPARE}$ dans ce cas on ne raffine plus la recherche dans cet intervalle). On obtient finalement une liste de fréquences. Dans chaque intervalle défini par les arguments de cette liste, se trouve une valeur propre ayant une certaine multiplicité. Comme approximation de cette valeur propre, on prend le milieu de l'intervalle.

Remarques:
- On aurait pu utiliser comme critère le changement de signe du polynôme caractéristique, mais outre le fait qu'il est très coûteux à évaluer, il ne permet pas, tel quel, de détecter les multiplicités,
- L'initialisation du processus peut s'effectuer de manière empirique suivant les besoins de l'utilisateur. Pour englober une partie du spectre on peut aussi utiliser les régionnements du plan complexe des théorèmes de Gerschgörin-Hadamard (sur $\mathbf{A}, \mathbf{A}^{\mathrm{T}}$ ...). Dans cette optique, la méthode de bissection peut s'avérer plus efficace qu'un QR en présence de cluster. Sa convergence, bien que linéaire, est en effet majorée par $1/2$ alors que celle de QR peut tendre vers 1[LT86].

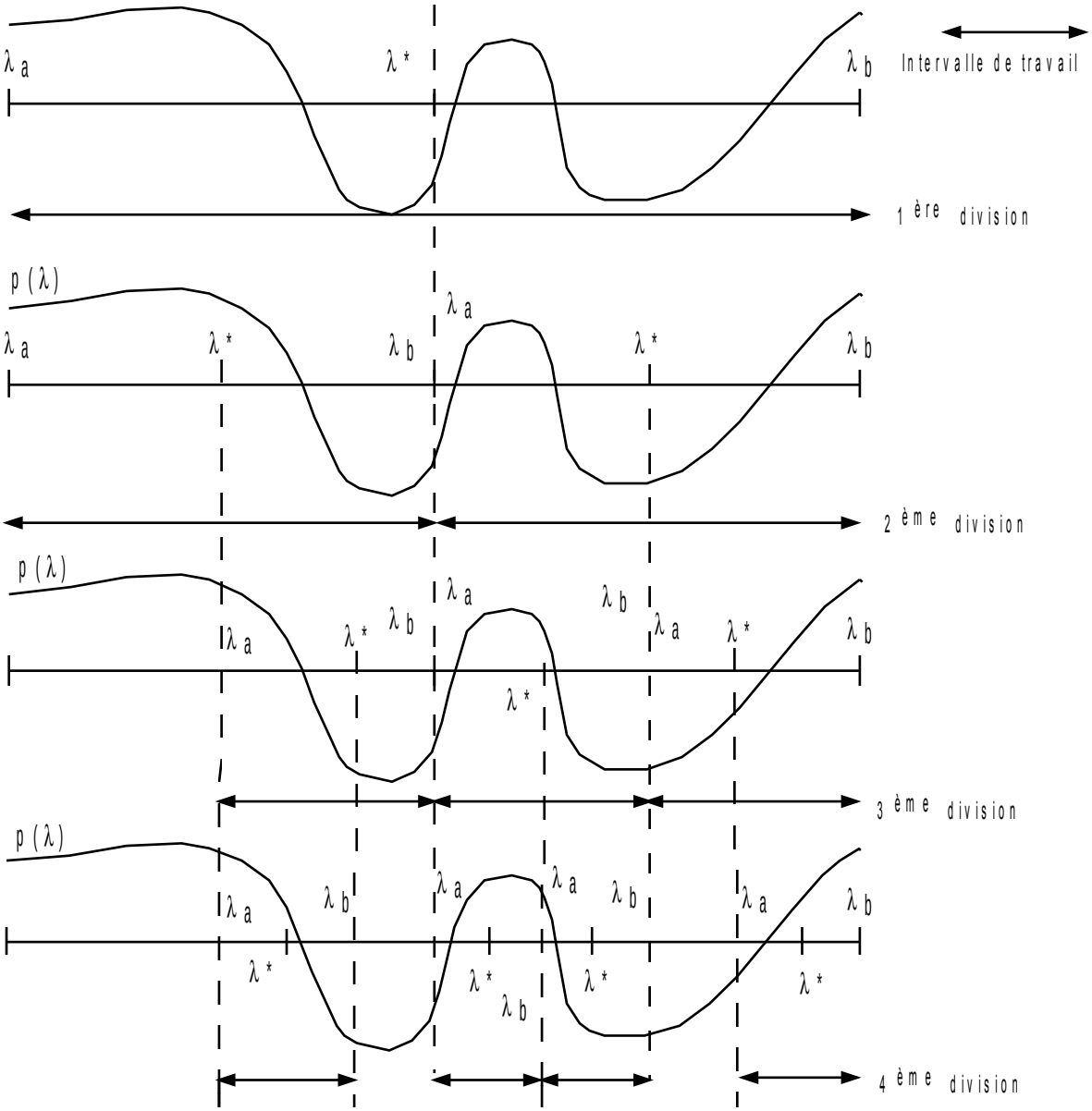


Figure 4.2-1. Méthode de bissection.

4.2.2 Méthode de la sécante

La méthode de la sécante est une **simplification de la méthode de Newton-Raphson**. A l'étape quelconque , k connaissant une valeur λ_k et en approximant la fonction non linéaire $p(\lambda)$ par sa tangente en ce point, on détermine la prochaine valeur λ_{k+1} comme étant l'intersection de cette droite avec l'axe des λ , et ainsi de suite, suivant le schéma itératif

$$\lambda_{k+1}=\lambda_k-p(\lambda_k)\frac{\lambda_k-\lambda_{k-1}}{p(\lambda_k)-p(\lambda_{k-1})}$$

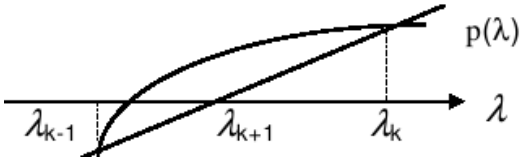


Figure 4.2-2. Méthode de la sécante.

La tangente étant approximée par une différence finie afin de ne pas avoir à calculer de dérivée de $p(\lambda)$, seule l'estimation du polynôme est requise. On considère qu'on a atteint la convergence quand $\frac{|\lambda_{k+1}-\lambda_k|}{\lambda_k} < \text{PREC\_AJUSTE}$ et, par ailleurs, on se limite à NMAX_ITER_AJUSTE itérations si ce critère n'est pas atteint.

Remarques:
•Cette méthode a une convergence presque quadratique lorsqu'elle est proche de la solution, dans le cas contraire, elle peut diverger. D'où l'intérêt de combiner la méthode de bissection avec cette approche. Cette stratégie élégante de combiner bissection et sécante a été initiée par Van Wijngaardeen et Dekker(1960).
•Si on ne réactualise par le calcul du dernier point dans la formule de la sécante ($\lambda_{k+1}=\lambda_0$), on retrouve la classique méthode de la «fausse position»[13]. Celle-ci converge moins vite (tout en requièrant une évaluation de fonction par itération), mais elle est robuste et stable (comme la méthode de bissection). On est sûr d'améliorer l'évaluation initiale.
•On montre que la méthode de la sécante réalise le meilleur compromis «vitesse de convergence/nombre d'évaluations de fonction par itération».
•La méthode de la sécante s'appuie sur une interpolation linéaire entre les deux derniers itérés ($\lambda_{k+1}, p(\lambda_{k+1})$) et ($\lambda_k, p(\lambda_k)$). En généralisant ce procédé à un ordre d'interpolation supérieure, c'est-à-dire les trois dern iers itérés, on retrouve la méthode de Müller-Traub utilisé pour les QEP avec MODE_ITER_INV [Boi09].
•
Nous allons maintenant détailler l'algorithme des puissances inverses (couplé à une accélération de Rayleigh) constituant la seconde partie du processus.

4.3 Méthode des puissances inverses

4.3.1 Principe

Pour déterminer la valeur propre du problème généralisé $\mathbf{A}\,\mathbf{u}=\lambda\,\mathbf{B}\,\mathbf{u}$ la plus proche en module de σ, on applique la méthode des puissances à l'opérateur $\left(\mathbf{A}-\sigma\,\mathbf{B}\right)^{-1}\mathbf{B}$. En fait, on ne construit que la **matrice factorisée**[14] $\mathbf{A}^{\sigma}=\left(\mathbf{A}-\sigma\,\mathbf{B}\right)^{-1}$ et cela revient à traiter le problème généralisé $(\mathbf{A}^{\sigma})^{-1}\mathbf{B}\,\mathbf{u}=\underbrace{\frac{1}{\lambda-\sigma}}_{\mu}\mathbf{u}$. La méthode des puissances convergeant prioritairement vers les valeurs propres de plus fort module (en μ), on capturera ainsi les λ les plus proches du shift.
Le principe est le suivant, connaissant une estimation σ de la valeur propre recherchée et partant d'un vecteur initial normalisé $\mathbf{y}_0$, on construit une suite de vecteurs propres approchés $(\mathbf{y}_k)_k$ par la formule récurrente

Pour $k = 1,\ldots$ faire
$\mathbf{A}^{\sigma}\ \tilde{\mathbf{y}}_k = \mathbf{B}\,\mathbf{y}_{k-1}$,
$\mu_k = \|\tilde{\mathbf{y}}_k\|$,
$\mathbf{y}_k = \frac{\tilde{\mathbf{y}}_k}{\mu_k}$,
Fin boucle.

13 Méthode appellée aussi «*regula falsi*» (en latin). Très vieille méthode de recherche de zéro de fonction, probablement inventée par les mathématiciens Indous du V^e siécle, et re-découverte en Europe par L.Fibonnacci (le monsieur des suites !) au XII^éme siècle.

14 Cette notation ne doit pas être confondue celle symbolisant le «shift and invert» $\mathbf{A}_{\sigma}=\left(\mathbf{A}-\sigma\,\mathbf{B}\right)^{-1}\mathbf{B}$.

Algorithme 4. Méthode des puissances inverses.

Avec λ_1, λ_2 ... les premières valeurs propres (de vecteur propre $\mathbf{u}_i$) les plus proches en module de σ , on montre que l'on a une convergence linéaire de $\mathbf{y}_k \to \mathbf{u}_1$ et une convergence quadratique de $\mu_k \to \frac{1}{|\lambda_1 - \sigma|}$ et $\frac{\tilde{\mathbf{y}}_k(i)}{\mathbf{y}_{k-1}(i)} \to \frac{1}{|\lambda_1 - \sigma|} \quad (i = 1....n)$ (le facteur de convergence de ces suites est de l'ordre de $\frac{|\lambda_1 - \sigma|}{\min\limits_{i \neq 1} |\lambda_i - \sigma|}$).

Remarques:
•Ce résultat n'est acquis (au facteur de convergence près) que lorsque la valeur propre dominante (ici celle la plus proche en module du shift) est unique. Dans le cas contraire, des résultats restent cependant possibles, même dans le cadre complexe, mais l'analyse rigoureuse de la convergence de cet algorithme est encore incomplète[GL89][Vau00].
•En théorie, ces résultats nécessitent que le vecteur initial ne soit pas orthogonal au sous-espace propre à gauche recherché. En pratique, les erreurs d'arrondis évitent ce problème (cf. [LT86] pp500-509).
•Même si l'estimation de la valeur propre est grossière, l'algorithme fournit rapidement une très bonne estimation du vecteur propre.
•L'inconvénient majeur de cette méthode est qu'il faut effectuer une factorisation pour chaque valeur propre à calculer.

En général on travaille en norme euclidienne ou en norme infinie, mais pour faciliter les calculs post-modaux on cherche ici à $\mathbf{B}$ -normaliser les vecteurs propres (lorsque $\mathbf{B}$ est indéfinie, on travaille avec la pseudo-norme associée). L'algorithme de base peut être réécrit en posant $\mathbf{z}_0 = \mathbf{B}\,\mathbf{y}_0$

Pour k = 1... faire

$$\mathbf{A}^{\sigma}\,\mathbf{y}_k = \mathbf{z}_{k-1},$$
$$\tilde{\mathbf{z}}_k = \mathbf{B}\,\mathbf{y}_k,$$
$$\rho(\mathbf{y}_k) = \frac{\mathbf{y}_k^T \,.\, \mathbf{z}_{k-1}}{\mathbf{y}_k^T \,.\, \tilde{\mathbf{z}}_k},$$
$$\varepsilon_k = sign\left(\mathbf{y}_k^T \,.\, \tilde{\mathbf{z}}_k\right),$$
$$\mathbf{z}_k = \varepsilon_k \frac{\tilde{\mathbf{z}}_k}{\left|\mathbf{y}_k^T \,.\, \tilde{\mathbf{z}}_k\right|^{\frac{1}{2}}},$$

Fin boucle.

Algorithme 5. Méthode des puissances inverses avec $\mathbf{B}$ -pseudo-norme.

Notons que $\rho(\mathbf{y_k}) \to \frac{1}{|\lambda_1 - \sigma|}$. Cette présentation évite les produits matrice-vecteur par la matrice $\mathbf{B}$ lors du calcul des produits scalaires et préfigure déjà le coefficient de Rayleigh du paragraphe suivant. On observe que le facteur de convergence est d'autant plus petit que le décalage spectral σ est proche de la valeur propre cherchée et donc que $\mathbf{A} - \sigma\,\mathbf{B}$ est proche de la **singularité**. Cela n'est en fait **pas préjudiciable** au processus car l'erreur faite en résolvant le système est "principalement" dans la direction engendrée par le vecteur propre qui est la direction cherchée. Cela signifie que lors de la résolution de $\mathbf{A}^{\sigma}\mathbf{y}_k = \mathbf{z}_{k+1}$, on ne trouve pas la solution exacte $\mathbf{y}_k$ mais que les erreurs d'arrondis conduisent à solution voisine de la forme $\tilde{\mathbf{y}}_k = \mathbf{y}_k + \mathbf{w}$. Celle-ci est proportionnelle à la solution exacte, mais comme la normalisation est arbitraire, tout se passe correctement[Par80].

Remarque:

•Ce mauvais conditionnement, loin d'avoir un effet défavorable, améliore même la convergence de l'algorithme.

Cet algorithme est donc utilisé pour améliorer le vecteur propre associé à la valeur approximée de la phase 1. Pour affiner cette estimation de la valeur propre, on introduit un quotient de Rayleigh.

4.3.2 Méthode d'itération du quotient de Rayleigh

Rappelons que le **quotient de Rayleigh** appliqué au problème généralisé se définit par le nombre $R(\mathbf{x})$, avec $\mathbf{x}$ un vecteur non nul de $\mathbb{R}^n$, tel que:

$$R(\mathbf{x})=\frac{\mathbf{x}^{\mathbf{T}}\mathbf{A}\,\mathbf{x}}{\mathbf{x}^{\mathbf{T}}\mathbf{B}\mathbf{x}}$$

Ce quotient possède la propriété remarquable de stationnarité au voisinage de tout vecteur propre et d'atteindre un **extremum** (local) qui est **la valeur propre** correspondante: pour chaque $\mathbf{x}$ fixé, $R(\mathbf{x})$ minimise $\lambda\rightarrow\|(\mathbf{A}-\lambda\,\mathbf{B})\,\mathbf{x}\|_2$.

Ce que nous pouvons traduire par "si $\mathbf{x}$ est une approximation d'un vecteur propre du système $\mathbf{A}\,\mathbf{x}=\lambda\,\mathbf{B}\,\mathbf{x}$, alors $R(\mathbf{x})$ est une approximation de la valeur propre associée au vecteur $\mathbf{x}$ ", et réciproquement, nous avons vu que si on disposait d'une bonne estimation d'une valeur propre, la méthode des itérations inverses permettait d'obtenir une bonne estimation du vecteur propre correspondant. D'où l'idée de combiner ces deux propriétés en considérant l'algorithme d'itération inverse avec décalage spectral pour lequel on réévalue, à chaque itération, la valeur propre via le quotient de Rayleigh. On obtient alors l'algorithme dit d'itérations du quotient de Rayleigh (en $\mathbf{B}$ -pseudo-norme)

Pour k = 1... faire

$$\left(\mathbf{A}-\mathrm{R}\left(\mathbf{y}_{k-1}\right)\,\mathbf{B}\right)\,\mathbf{y}_k=\mathbf{z}_{k-1},$$
$$\tilde{\mathbf{z}}_k=\mathbf{B}\,\mathbf{y}_k,$$
$$\mathrm{R}\left(\mathbf{y}_k\right)=\frac{\mathbf{y}_k^T\,.\,\mathbf{z}_{k-1}}{\mathbf{y}_k^T\,.\,\tilde{\mathbf{z}}_k}+\mathrm{R}\left(\mathbf{y}_{k-1}\right),$$
$$\varepsilon_k=sign\left(\mathbf{y}_k^T\,.\,\tilde{\mathbf{z}}_k\right),$$
$$\mathbf{z}_k=\varepsilon_k\,\frac{\tilde{\mathbf{z}}_k}{\left|\mathbf{y}_k^T\,.\,\tilde{\mathbf{z}}_k\right|^{\frac{1}{2}}},$$

Fin boucle.

Algorithme 6. Méthode d'itérations du quotient de Rayleigh avec $\mathbf{B}$ -pseudo-norme.

Pour le problème modal standard, on peut montrer[Par80] que la convergence de cet algorithme est cubique dans le cas où l'opérateur de travail est normal[15] (*a fortiori* dans le cas hermitien) et au mieux quadratique dans les autres cas. Si on utilisait cette méthode sans la modifier, il faudrait à chaque itération du processus d'amélioration de chaque valeur propre, effectuer une factorisation $\mathbf{L}\,\mathbf{D}\,\mathbf{L}^{\mathrm{T}}$, ce qui serait très coûteux. D'où l'idée de n'effectuer[16] ce décalage de Rayleigh, que si on est dans un voisinage (notion arbitraire à définir) de la solution.

Remarque:

15 Une matrice $\mathbf{A}$ est dite normale si $\mathbf{A}\,\mathbf{A}^*=\mathbf{A}^*\mathbf{A}$. C'est donc le cas des opérateurs hermitiens, antihermitiens ou unitaires.

16 Cette stratégie du couplage de méthodes aux caractéristiques complémentaires voire antagonistes est souvent usitée en analyse numérique. Par exemple, en optimisation, la méthode de Levenberg-Marquadt couple une steepest-descent et un Newton.

•Dans un cadre plus général, certains auteurs ont introduit un algorithme dit «de bi-itérations du quotient de Rayleigh». Basé sur la stationnarité du bi-quotient $R_b(\mathbf{x},\mathbf{y})=\frac{\mathbf{y}^T\mathbf{A}\,\mathbf{x}}{\mathbf{y}^T\mathbf{B}\,\mathbf{x}}$ au voisinage des vecteurs propres à droite et à gauche, il fournit (en non hermitien) les deux types de vecteurs propres. Son coût est cependant rédhibitoire car il requiert deux fois plus de factorisations (cf. B.N.Parlett 1969 [Par80]).

4.3.3 Implantation dans Code_Aster

Ce décalage spectral n'est activé que si le mot-clé OPTION du mot-clé facteur CALC_MODE est initialisé à 'RAYLEIGH'. Par défaut, on a 'DIRECT' et le décalage est alors classique (correction globale plutôt que progressive de la valeur propre). L'algorithme mis en place dans le code se découpe comme suit (en $\mathbf{B}$ -pseudo norme):

•Initialisation de la valeur propre à partir de l'estimation de la première phase: λ^* .
•Calcul d'un vecteur initial $\mathbf{y}_0$ aléatoire vérifiant les conditions limites.
• $\mathbf{B}$ -orthonormalisation de $\mathbf{y}_0$ par rapport aux modes précédemment calculés (si c'est un mode multiple d'après la première phase) par un Gram-Schmidt Modifié[17] (GSM).
•Calcul de $\delta_0=\mathrm{sign}(\mathbf{y}_0^T\mathbf{B}\,\mathbf{y}_0)$.
•Pour $k=1$, NMAX_ITER faire
Résoudre $(\mathbf{A}-\lambda^*\mathbf{B})\,\mathbf{y}_k=\delta_{k-1}\mathbf{B}\,\mathbf{y}_{k-1}$.
$\mathbf{B}$ -orthonormalisation (éventuelle) de $\mathbf{y}_k$
Calcul de la correction de la valeur propre $c=\frac{\mathbf{y}_k^T\mathbf{B}\,\mathbf{y}_{k-1}}{\mathbf{y}_k^T\mathbf{B}\,\mathbf{y}_k}$.
Si $|\,|\mathbf{y}_k^T\mathbf{B}\,\mathbf{y}_{k-1}|-1|\le PREC$ alors
$\lambda^*=\lambda^*+c$,
exit;
Sinon
Si OPTION='RAYLEIGH' et si $|c|\le 0.1\,\lambda^*$ alors
$\lambda^*=\lambda^*+c$;
Fin si.
Fin si.
Fin boucle.

Algorithme 7. Schéma fonctionnel de la méthode des puissances inverses dans MODE_ITER_INV.

La norme d'erreur maximale admissible PREC et le nombre maximal d'itérations autorisées NMAX_ITER sont des arguments du mot-clé facteur CALC_MODE.

Remarques:
•Le vecteur propre étant $\mathbf{B}$ -normé, on considère avoir atteint la convergence lorsque la valeur absolue du produit scalaire impropre est proche de l'unité.
•Pour éviter de prendre un vecteur initial $\mathbf{B}$ -orthogonal à la valeur propre cherchée on utilise une méthode de tirage aléatoire pour les composantes de ce vecteur.
•D'autre part pour pouvoir déterminer des modes multiples ou proches, on utilise une $\mathbf{B}$ -orthogonalisation aux modes déjà calculés.
•L'option "d'accélération" de l'algorithme par quotient de Rayleigh étant coûteuse, elle n'est utilisée à chaque itération que si on est au voisinage de la valeur propre recherchée.

17 Depuis le développement de cet opérateur, des méthodes d'orthogonalisation plus efficaces et plus robustes se sont répandues, tels que les Gram-Schmidt Modifiés Itératifs (IGSM) utilisés dans MODE_ITER_SIMULT (cf. Annexe 3 et [Par80]).

4.4 Périmètre d'utilisation

GEP à matrices réelles symétriques.

L'utilisateur peut spécifier la classe d'appartenance de son calcul (dynamique ou flambement) en initialisant le mot-clé TYPE_RESU. Suivant cette valeur, on renseigne le vecteur FREQ ou CHAR_CRIT.

4.5 Affichage dans le fichier message

Dans le fichier message, les résultats sont affichés sous forme de tableau

```
-----------------------------------------------------------------------
LE NOMBRE DE DDL
TOTAL EST:                     192
DE LAGRANGE EST:                84
LE NOMBRE DE DDL ACTIFS EST:        66
-----------------------------------------------------------------------
   VERIFICATION DU SPECTRE DE FREQUENCES (SUITE DE STURM)
LE NOMBRE DE FREQUENCES DANS LA BANDE ( 1.00000E-02, 6.00000E-02) EST 4
-----------------------------------------------------------------------
CALCUL MODAL:  METHODE D'ITERATION INVERSE
                          DICHOTOMIE   SECANTE       INVERSE
NUMERO FREQUENCE(HZ) AMORT NB_ITER/NB_ITER/PRECISION/NB_ITER/  PRECISION
4   1.97346E-02   0.00000E+00  4    6    2.97494E-07    4   1.22676E-07
5   2.40228E-02   0.00000E+00  4    5    4.21560E-05    3   4.49567E-09
6   4.40920E-02   0.00000E+00  3    5    2.19970E-05    3   2.62910E-09
7   5.23415E-02   0.00000E+00  3    5    2.34907E-07    5   1.32212E-07
-----------------------------------------------------------------------
VERIFICATION A POSTERIORI DES MODES
-----------------------------------------------------------------------
```

Exemple 4. Trace de MODE_ITER_INV (GEP) dans le fichier message.

Avec l'option 'PROCHE', les colonnes "Dichotomie" et "Sécante" n'apparaissent pas, tandis qu'avec l'option 'SEPARE', seule la colonne "Sécante" disparaît. La dernière colonne précision regroupe des données intermédiaires et ne représente pas, comme pour l'autre opérateur modal MODE_ITER_SIMULT, la norme d'erreur du résidu. C'est un artefact qui va être amené à disparaître.

4.6 Récapitulatif du paramétrage

Récapitulons maintenant le paramétrage de l'opérateur MODE_ITER_INV.

Opérande	Mot-clé	Valeur par défaut	Références
	TYPE_RESU ='DYNAMIQUE'	'DYNAMIQUE'	§3.1
	'MODE_FLAMB'		§3.1
CALC_FREQ	**FREQ**		§4.1
	CHAR_CRIT		§4.1
	OPTION ='SEPARE'	'AJUSTE'	§4.1
	'AJUSTE'		§4.1
	'PROCHE'		§4.1
	NMAX_FREQ	0	§4.1
	NMAX_ITER_SEPARE	30	§4.2
	PREC_SEPARE	1.E-04	§4.2
	NMAX_ITER_AJUSTE	15	§4.2
	PREC_AJUSTE	1.E-04	§4.2
	NMAX_ITER_SHIFT	3	[Boi12] §3.2

Opérande	Mot-clé	Valeur par défaut	Références
	PREC_SHIFT	0.05	[Boi12] §3.2
	SEUIL_FREQ	1.E-02	§3.7
CALC_MODE	**OPTION** ='DIRECT'	'DIRECT'	§4.3
	'RAYLEIGH'		§4.3
	PREC	1.E-05	§4.3
	NMAX_ITER	30	§4.3
VERI_MODE	STOP_ERREUR='OUI'	'OUI'	§3.7
	'NON'		
	SEUIL	1.E-02	§3.7

Tableau 4.6-1. Récapitulatif du paramétrage de MODE_ITER_INV (GEP).

Remarques:
•On retrouve toute la "tripaille" de paramètres liée aux pré-traitements du test de Sturm (NMAX_ITER_SHIFT, PREC_SHIFT) et aux post-traitements de vérification (SEUIL_FREQ, VERI_MODE).
•Lors des premiers passages, il est fortement conseillé de ne modifier que les paramètres principaux notés en gras. Les autres concernent plus les arcanes de l'algorithme et ils ont été initialisés empiriquement à des valeurs standards.

5 Méthode de sous-espace (MODE_ITER_SIMULT)

5.1 Introduction

Le cas des GEP à modes complexes (cf. §3.1) peut être traité dans *Code Aster* uniquement par les méthodes IRAM et QZ. Il est donc abordé au chapitre relatif à ces méthodes. Par soucis de lisibilité, on se limitera au cas des matrices à coefficients réels pour exposer les principes généraux des méthodes de sous-espace.

Si on souhaite **seulement calculer les** p **éléments propres d'un problème généralisé d'ordre** n où $n \ll p$ (par exemple, les p plus petites valeurs propres ou toutes les valeurs propres comprises dans un intervalle donné), on a vu qu'il était préférable d'avoir recourt à des méthodes de sous-espace. Elles sont basées sur l'analyse de Rayleigh-Ritz qui consiste à projeter efficacement le problème considéré sur un sous espace de dimension $m\,(p<m<n)$ et à rechercher certains éléments propres de ce problème projeté (beaucoup plus facile à traiter) à l'aide d'algorithmes robustes (QR ou QL pour Lanczos et IRAM, Jacobi pour la méthode de Bathe et Wilson).

Les critères d'efficacité de ladite projection sont:
•La petite taille de l'espace de projection (directement liée aux complexités calcul et mémoire).
•La facilité de sa construction.
•La robustesse de la projection orthogonale[18] .
•La mise sous une forme canonique de la matrice projetée.
•La bonne approximation de la partie du spectre initial recherché par celui de l'opérateur projeté.
•La minimisation des complexités calcul et mémoire et celle des effets d'arrondis (ceux-ci sont surtout liés aux problèmes d'orthogonalité soulevés par le 3ième point).

On présentera tout d'abord l'analyse de Rayleigh-Ritz avant de détailler trois méthodes issues de cette analyse : la méthode de Lanczos, celle dite de Sorensen (IRA) et celle de Bathe et Wilson.

5.2 Analyse de Rayleigh-Ritz

Considérons le problème modal standard d'ordre n , $\mathbf{A}\,\mathbf{u}=\lambda\,\mathbf{u}$, et le sous-espace H de $\mathbb{R}^n$ engendré par une base orthonormée $(\mathbf{q}_1,\mathbf{q}_2,\ldots,\mathbf{q}_m)$. Cette dernière constitue la matrice orthogonale $\mathbf{Q}_m$ permettant de définir l'opérateur de projection $\mathbf{P}_m=\mathbf{Q}_m\mathbf{Q}_m^T$.

La **méthode de Galerkin** utilisée par cette analyse consiste à résoudre le problème suivant

$$\left\{\begin{array}{c}\text{Trouver } (\tilde{\lambda},\tilde{\mathbf{u}})\in\mathbb{R}\times H \text{ tel que}\\ \mathbf{P}_m(\mathbf{A}\,\tilde{\mathbf{u}}-\tilde{\lambda}\,\tilde{\mathbf{u}})=\mathbf{0}\end{array}\right. \Leftrightarrow \left\{\begin{array}{c}\text{Avec } \tilde{\mathbf{u}}=\mathbf{Q}_m\mathbf{x}\text{, trouver } (\tilde{\lambda},\mathbf{x})\in\mathbb{R}\times\mathbb{R}^m \text{ tel que}\\ \underbrace{\mathbf{Q}_m^*\mathbf{A}\mathbf{Q}_m}_{\mathbf{B}_m}\mathbf{x}=\tilde{\lambda}\,\mathbf{x}\end{array}\right.$$

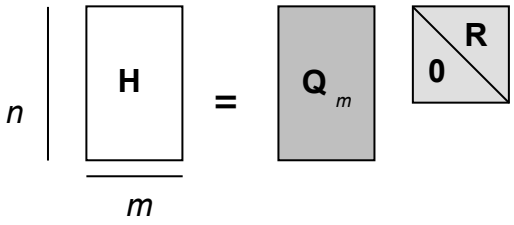
La recherche des éléments de Ritz ($\tilde{\lambda}$, $\tilde{\mathbf{u}}$) qui nous intéressent revient donc à trouver les modes propres de la matrice de Rayleigh $\mathbf{B}_m=\mathbf{Q}_m^T\mathbf{A}\,\mathbf{Q}_m$.

Les valeurs propres restent inchangées dans les deux formulations, par contre connaissant un vecteur propre $\mathbf{x}$ de $\mathbf{B}_m$ on doit remonter à l'espace tout entier via la transformation $\tilde{\mathbf{u}}=\mathbf{Q}_m\mathbf{x}$.

La démarche peut donc se résumer sous la forme:

18 Dans le cas non hermitien, des projections obliques ont été développées par F.Chatelin[Cha88].

•Choix d'un espace H et d'une base $(\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_m)$ représentée par $\mathbf{H}$.
•Orthonormalisation de la base



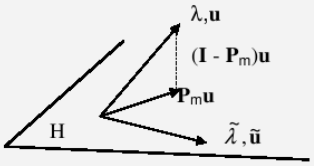
•Calcul de la matrice de Rayleigh $\mathbf{B}_m := \mathbf{Q}_m^* \mathbf{A} \mathbf{Q}_m$
•Calcul des éléments propres de $\mathbf{B}_m : (\tilde{\lambda}, \mathbf{x})$
•Calcul de ses éléments de Ritz: $(\tilde{\lambda}, \tilde{\mathbf{u}} = \mathbf{Q}_m \mathbf{x})$,
•Test d'erreur via le résidu: $\mathbf{r} = \mathbf{A}\tilde{\mathbf{u}} - \tilde{\lambda}\tilde{\mathbf{u}}$.

Algorithme 8. Procédure de Rayleigh-Ritz.

Remarques:
•Les éléments de Ritz sont en fait les modes propres de la matrice (d'ordre N) de l'approximation de Galerkin $\mathbf{C}_m = \mathbf{P}_m \mathbf{A} \mathbf{P}_m$.
•La complexité calcul de ce processus, de l'ordre au pire de $O(m^2(4n+m))$ (beaucoup moins avec un bon espace, cf. Lanczos et IRAM), est sans commune mesure avec celle d'un bon QR ($O(n^2)$) ou celle d'une itération du quotient de Rayleigh encapsulée dans un processus heuristique tel que celui développé dans MODE_ITER_INV ($\gg pn^3$). Mais il est bien clair que ces méthodes sont plus complémentaires que concurrentes.
•
Pour estimer l'erreur commise en utilisant les éléments propres de la matrice $\mathbf{B}_m$ on a le résultat suivant.

Théorème 1
Soit $(\lambda, \mathbf{u})$ un élément propre de la matrice diagonalisable (c'est le cas des matrices normales et des matrices non défectives (matrices dont toutes les valeurs propres ont même multiplicité arithmétique et géométrique) $\mathbf{A}$ et, $\mathbf{B}_m$ la matrice de Rayleigh associée, alors celle-ci admet une valeur propre $\tilde{\lambda}$ vérifiant

$$|\lambda - \tilde{\lambda}| \leq \beta \frac{\|(\mathbf{I} - \mathbf{P}_m)\mathbf{u}\|_2}{\|\mathbf{P}_m \mathbf{u}\|_2^2}$$



où β est un nombre dépendant de λ , $\mathbf{A}$ et $\mathbf{P}_m$.
Dans le cas hermitien le numérateur de cette majoration est au carré. Le vecteur propre associé, $\tilde{\mathbf{u}}$, vérifie quant à lui

$$\sin(\mathbf{u}, \tilde{\mathbf{u}}) \leq \beta \frac{\|(\mathbf{I} - \mathbf{P}_m) - \mathbf{u}\|_2}{\|\mathbf{P}_m \mathbf{u}\|_2} \quad .$$

Preuve
Cf. [LT86] pp531-537.

On montre que pour m assez grand, β peut être majoré par un nombre qui ne dépend que de $\mathbf{A}$ et de $\min_{\lambda_i \neq \lambda_j} |\lambda_i - \lambda_j|$. L'estimation du second membre $\|(\mathbf{I} - \mathbf{P}_m)\mathbf{u}\|_2$ dépend du choix du sous-espace.

Remarque:
•Le nombre $\min_{\lambda_i \neq \lambda_j} |\lambda_i - \lambda_j|$ (et ses variantes) est omniprésent dans les analyses d'erreurs, de convergence ou de conditionnements spectraux. On voit, une fois de plus, l'importance de la séparation de spectre de travail sur la bonne tenue numérique des algorithmes.

Bien sûr, pour tenir compte de l'information spectrale déjà obtenue, voire pour l'améliorer ou la filtrer, on itère cette procédure de Rayleigh-Ritz en modifiant le sous-espace de projection. On enchaînera alors la construction de ce nouvel espace (récurrent du précédent) et ce processus de projection. Deux types d'espaces (chacun rattaché à des méthodes distinctes) sont les plus souvent usités.

5.3 Choix de l'espace de projection

Deux choix d'espace de projection H sont les plus souvent mis en place:

•Le premier, celui de la **méthode de Bathe et Wilson** (cf. §8 METHODE='JACOBI') , consiste à choisir un sous espace H de dimension m puis à construire successivement:

$$\mathbf{H}_1 = \mathbf{A}\,\mathbf{H}$$
$$\mathbf{H}_2 = \mathbf{A}\,\mathbf{H}_1 = \mathbf{A}^2\,\mathbf{H}$$
$$\ldots$$
$$\mathbf{H}_i = \mathbf{A}\,\mathbf{H}_{i-1} = \mathbf{A}^i\,\mathbf{H}$$

Cette méthode, qui tient à la fois de la généralisation sous forme bloc de la méthode des puissances et de la troncature de l'algorithme QR, conduit à un appauvrissement de l'espace de travail: $dim\,H_i \leq dim\,H_{i-1}$. Pour ne pas retrouver toujours le même mode propre dominant il faut insérer dans le processus une réorthogonalisation (avec tous les problèmes de complexité calcul et d'arrondis que cela implique).

•Le second, celui de la **méthode de Lanczos** (cf. §6 METHODE='TRI_DIAG') et d'**IRAM** (cf. §7 METHODE='SORENSEN') consiste à partir d'un vecteur initial $\mathbf{y}$, à construire la suite d'espaces de Krylov (H_i)$_{i=1,m}$ où H_i est l'espace engendré par la famille de vecteurs $\left(\mathbf{y}, \mathbf{A}\mathbf{y}, \ldots, \mathbf{A}^{i-1}\mathbf{y}\right)$. Ce dernier est appelé l'espace de Krylov de $\boldsymbol{A}$ et d'ordre i initié par $\mathbf{y}$:

$$H_i = K_i(\mathbf{A}, \mathbf{y}) \equiv Vect\left(\mathbf{y}, \mathbf{A}\mathbf{y}, \ldots, \mathbf{A}^{i-1}\mathbf{y}\right) .$$

Ils vérifient la propriété suivante $dim\,H_i \leq dim\,H_{i+1}$. Contrairement au choix précédent, on a donc un certain enrichissement de l'espace de travail. D'autre part, on verra qu'ils permettent de projeter de manière optimale au sens des critères définis précédemment.

Historiquement, le premier type d'espace a été très utilisé en mécanique des structures. Mais pour diminuer la complexité calcul liée à la taille des sous-espaces et aux orthogonalisations, on leur préfère désormais les méthodes de type Lanczos/Arnoldi.

Remarque:
•On rencontre une variété impressionnante d'algorithmes utilisant un espace du premier type. Ils sont appelés "'itérations de sous-espace", "itérations orthogonales" ou encore "itérations simultanées". Au-delà des différents vocables, il faut surtout retenir que ces adaptations concernent les stratégies de redémarrages[19], les techniques d'accélération[20] et de factorisation mises en œuvre pour enrichir l'espace de travail. Il existe même des versions basées sur les puissances inverses permettant de calculer les p plus petits modes (cf. [LT86] pp538-45, [Vau00] pp49-54).

Pour ces méthodes de projection, en supposant que l'espace initial H ne soit pas trop pauvre suivant les p directions dominantes, le facteur de convergence du i ème mode au bout de k itérations s'écrit (lorsqu'ils sont rangés classiquement, c'est à dire par ordre décroissant de module):

19 Technique consistant à redémarrer un algorithme modal avec un vecteur initial comportant déjà de l'information spectrale. Typiquement, on choisit une combinaison linéaire des vecteurs propres ou des vecteurs de Schur déjà identifiés.

20 Technique consistant à remplacer la matrice de travail $\mathbf{A}$ par un polynôme matriciel $\mathbf{P}(\mathbf{A})$ qui a la particularité d'être de grande amplitude dans les régions spectrales d'intérêt (cf. §7.4).

$$O\left(\left|\frac{\lambda_{m+1}}{\lambda_i}\right|^k\right)$$

Il exprime clairement deux phénomènes:
•La convergence prioritaire des modes dominants.
•L'amélioration de ces convergences (et donc de leurs normes d'erreur pour un nombre d'itérations fixé) lorsque la taille du sous-espace augmente.

Pour transformer le problème modal généralisé en un problème standard on a recourt à des transformations spectrales. Elles permettent aussi d'orienter la recherche du spectre et d'améliorer la convergence (cf. §3.6).

5.4 Choix du décalage spectral

Pour calculer les plus petites valeurs propres voisines d'une fréquence donnée ou toutes les valeurs propres comprises dans une bande de fréquence donnée, on effectue un décalage spectral du problème généralisé. Soit σ la valeur de décalage, on transforme le problème initial $\mathbf{A}\,\mathbf{u}=\lambda\,\mathbf{B}\,\mathbf{u}$ en un problème standard décalé.

$$\mathbf{A}^{\sigma}\mathbf{B}^{-1}\mathbf{u}=\mu\,\mathbf{u} \quad \text{avec} \quad \mathbf{A}^{\sigma}=\mathbf{A}-\sigma\,\mathbf{B} \quad \text{et} \quad \mu=\lambda-\sigma$$

Cette transformation spectrale, dite de "**shift simple**", est utilisée dans la méthode de Bathe et Wilson. Comme le processus détecte les plus petites valeurs propres en μ , on capture progressivement celles qui sont les plus proches de σ (en module si le shift est réel et en partie réelle et imaginaire s'il est complexe).

On effectuant la transformation inverse ("**shift and invert**") il advient

$$\mathbf{A}_{\sigma}\mathbf{u}=\mu\,\mathbf{u} \quad \text{avec} \quad \mathbf{A}_{\sigma}=(\mathbf{A}-\sigma\,\mathbf{B})^{-1}\mathbf{B} \quad \text{et} \quad \mu=\frac{1}{\lambda-\sigma}$$

C'est le problème traité avec Lanczos et IRAM qui permet de calculer les plus grandes valeurs propres μ donc les valeurs propres les plus proches du shift σ (en module). Dans le cas de la méthode IRAM pour des matrices à coefficients complexes, le shift est obligatoirement complexe.

Dans la commande MODE_ITER_SIMULT, il y a quatre façons de choisir ce shift (pour le flambement, la transposition aux niveaux de charges critiques est immédiate):

- $\sigma=0$, on cherche les plus petites valeurs propres du problème de départ. Ceci correspond à OPTION='PLUS_PETITE' sous le mot-clé facteur CALC_FREQ.
- $\sigma=\sigma_0$ avec $\sigma_0=(2\pi f_0)^2$, on cherche les fréquences proches de la fréquence FREQ= f_0 (OPTION='CENTRE').
- $\sigma=\frac{\sigma_0+\sigma_1}{2}$ avec $\sigma_0=(2\pi f_0)^2$ et $\sigma_1=(2\pi f_1)^2$, on cherche toutes les fréquences comprises dans la bande $[f_0, f_1]$ (OPTION='BANDE' avec FREQ={ f_0, f_1 }).
- $\sigma=\sigma_0+j\sigma_1$ avec $\sigma_0=(2\pi f_0)^2$ et $\sigma_1=\eta\frac{(2\pi f_0)^2}{2}$ on cherche les fréquences proches de la fréquence FREQ= f_0 (OPTION='CENTRE') et de l'amortissement réduit AMOR_REDUIT= η (OPTION='CENTRE'). Cette approche n'est disponible, qu'en dynamique avec un GEP à modes complexes: matrices $\mathbf{A}$ et/ou $\mathbf{B}$ complexes symétriques ou réelles non symétriques (cf. tableau 1).

Le nombre de fréquences à calculer est donné en général par l'utilisateur à l'aide de NMAX_FREQ sous le mot-clé facteur CALC_FREQ. Mais pour l'option 'BANDE', il est déterminé automatiquement en utilisant le corollaire 3bis de R5.01.04.

Remarque:
•On rappelle que la factorisation de la matrice de travail, tout comme les tests de Sturm de l'option 'BANDE' sont tributaires d'instabilités numériques lorsque le shift est proche d'une valeur propre. Des traitements palliatifs, paramétrables par l'utilisateur, ont été implantés (cf. R5.01.04).

6 Méthode de Lanczos (METHODE='TRI_DIAG')

6.1 Introduction

Cet algorithme, mis au point par **Lanczos**[Lan50] **en 1950**, n'a guère été utilisé jusqu'au milieu des années 70. De prime abord très simple et efficace, il est le siège néanmoins **de grandes instabilités numériques** pouvant provoquer la capture de modes fantômes ! Celles-ci sont liées principalement à des **problèmes de maintien d'orthogonalité** entre les vecteurs engendrant l'espace de Krylov. La compréhension de ce type de comportement face à l'arithmétique finie des ordinateurs fut longue à exhumer. Depuis, de nombreuses solutions palliatives ont vu le jour et une abondante littérature couvre le sujet. Citons par exemple l'ouvrage de J.K. Cullum[CW85] qui lui est entièrement consacré, celui de B.N.Parlett[Par80] et la synthèse actualisée et exhaustive de J.L. Vaudescal[Vau00] (pp55-78).

Dans la suite de ce paragraphe, nous allons tout d'abord nous attarder sur le cadre théorique de fonctionnement de l'algorithme. Puis, avant de détailler la variante mise en place dans *Code_Aster*, nous nous attacherons au cadre réaliste de l'arithmétique finie.

6.2 Algorithme de Lanczos théorique

6.2.1 Principe

Son périmètre d'application recouvre les couples opérateur de travail-(pseudo)produit scalaire assurant l'**hermiticité** de $\mathbf{A}_\sigma$. Il permet de construire itérativement une matrice de Rayleigh $\mathbf{B}_m$ de **taille modulable, tridiagonale et hermitienne** (avec un véritable produit scalaire, sinon elle perd cette dernière propriété). Cette forme particulière facilite grandement le calcul des modes de Ritz: avec QR on perd alors un ordre de magnitude passant, lorsqu'on recherche p modes propres en projetant sur un sous-espace de taille m , de $O(pm^2)$ à $O(20\mathrm{pm})$. L'algorithme consiste à construire progressivement une famille de vecteurs de Lanczos $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_m$ en projetant orthogonalement, à l'itération k , le vecteur $\mathbf{A}_\sigma \mathbf{q}_k$ sur les deux vecteurs précédents $\mathbf{q}_k$ et $\mathbf{q}_{k-1}$. Le nouveau vecteur devient $\mathbf{q}_{k+1}$ et ainsi, de proche en proche, on assure structurellement l'orthogonalité de cette famille de vecteurs. Le processus itératif se résume ainsi.

$\mathbf{q_0}=0, \beta_0=0$.
Calcul de $\mathbf{q}_1/\|\mathbf{q}_1\|=1$.
Pour k=1,m faire
$\mathbf{z}=\mathbf{A}_\sigma \mathbf{q}_k-\beta_{k-1}\mathbf{q}_{k-1}$,
$\alpha_k=(\mathbf{z}, \mathbf{q}_k)$,
$\mathbf{v}=\mathbf{z}-\alpha_k \mathbf{q}_k$,
$\beta_k=\|\mathbf{v}\|$,
Si $\beta_k \neq 0$ alors
$\mathbf{q}_{k+1}=\dfrac{\mathbf{v}}{\beta_k}$,
Sinon
Déflation ;
Fin si.
Fin boucle.

Algorithme 9. Lanczos théorique.

En notant, $\mathbf{e}_m$ le $m^{\text{ième}}$ vecteur de la base canonique, le **vecteur résidu de la factorisation de Lanczos** s'écrit $\mathbf{R}_m = \beta_m \mathbf{q}_{m+1} \mathbf{e}_m^T$.

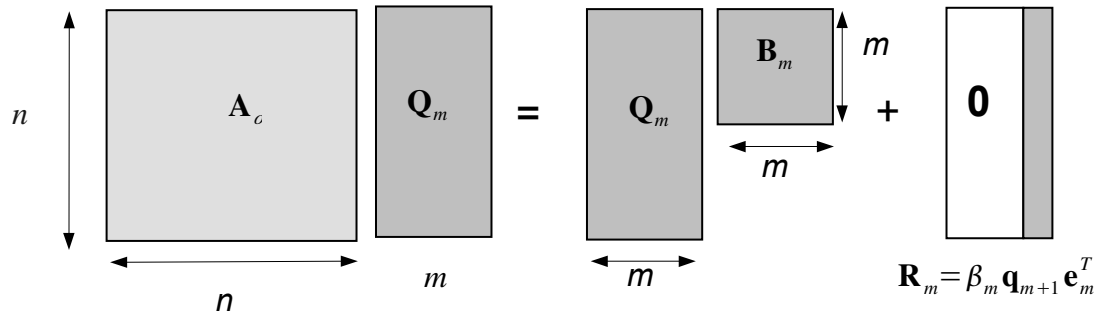


Figure 6.2-1. Factorisation de Lanczos.

La **matrice de Rayleigh** prend alors la forme (en complexe avec un produit scalaire hermitien)

$$\mathbf{B}_m = \begin{bmatrix} \alpha_1 & \overline{\beta}_1 & 0 & 0 \\ \beta_1 & \alpha_2 & \dots & 0 \\ 0 & \dots & \dots & \overline{\beta}_{m-1} \\ 0 & 0 & \beta_{m-1} & \alpha_m \end{bmatrix}$$

Par construction, cet algorithme nous assure des résultats suivants (en arithmétique exacte):

•Les $(\mathbf{q}_k)_{k=1,m}$ constituent une famille orthonormale.

•Ils engendrent l'espace de Krylov d'ordre m initié par $\mathbf{q}_1$, $\mathbf{K}_m(\mathbf{A}_\sigma, \mathbf{q}_1) = Vect(\mathbf{q}_1, \mathbf{A}_\sigma \mathbf{q}_1, \dots, \mathbf{A}_\sigma^{m-1} \mathbf{q}_1)$.

•Ils permettent de construire une matrice $\mathbf{B}_m$ tridiagonale, hermitienne et de taille modulable.

•Le spectre de $\mathbf{B}_m$ n'admet que les m modes simples dominants sur lesquels $\mathbf{q}_1$ a une composante.

Remarques:

• *L'algorithme ne permet donc pas, en théorie, la capture de modes multiples. Cela peut s'expliquer en remarquant que toute matrice de Hessenberg [21] irréductible (donc a fortiori une matrice tridiagonale) n'admet que des modes simples.*

•Le coût d'une itération est faible, de l'ordre de celui d'un produit matrice-vecteur $\mathbf{A}_\sigma \mathbf{q}_k$, soit pour les m premières itérations une complexité calcul de $O(nm(c+5))$ (avec c le nombre moyen de termes non nuls sur les lignes de la matrice de travail). D'autre part, la complexité mémoire requise est faible car on n'a pas besoin de construire $\mathbf{A}_\sigma$ en entier, on a juste besoin de connaître son action sur un vecteur. Cette caractéristique est très intéressante pour résoudre des problèmes de grandes tailles[Vau00].

•En général le domaine d'activité oriente le choix d'un vecteur de Lanczos initial, celui-ci doit par exemple appartenir à un espace de solutions admissibles (vérifiant des contraintes, des conditions limites...) ainsi qu'à l'ensemble image de l'opérateur. Ce dernier point est important car il permet d'enrichir plus rapidement la recherche modale en ne se limitant pas au noyau.

•L'algorithme de Lanczos n'est qu'un moyen d'aborder les sous-espaces de Krylov. Sa généralisation aux configurations non symétriques est appelée algorithme d'Arnoldi (cf.§7.2).

6.2.2 Estimations d'erreurs et de convergences

21 Une matrice $\mathbf{A}$ est dite de Hessenberg supérieure (resp. inférieure) si $\mathbf{A}_{i,j} = 0$ pour $i > j+1$ (resp. $j > i+1$). Elle est de plus irréductible si $\mathbf{A}_{i+1,i} \neq 0 \; \forall i$.

Du fait de la forme particulière de $\mathbf{B}_m$, l'extraction des éléments de Ritz recherchés est simplifiée, et en plus, le résultat suivant nous permet rapidement (*via* un produit de deux réels !) d'estimer leurs qualités.

Propriété 2
La norme euclidienne du résidu de l'élément de Ritz $\left(\tilde{\lambda}, \tilde{\mathbf{u}}=\mathbf{Q}_m \mathbf{x}\right)$ est égale à

$$\|\mathbf{r}\|_2=\|(\boldsymbol{\sigma}-\tilde{\lambda}\mathbf{I})\tilde{\mathbf{u}}\|_2=|\beta_m||\mathbf{e}_m^T\mathbf{x}|$$

Preuve:
Triviale en prenant la norme euclidienne de la factorisation de Lanczos $\mathbf{A}_\sigma \mathbf{B}_m \mathbf{x}=\mathbf{Q}_m \mathbf{B}_m \mathbf{x}+\beta_m \mathbf{q}_{m+1}\mathbf{e}_m^T\mathbf{x}$ où $\mathbf{q}_{m+1}$ est normé à l'unité.

Remarques:
•Un résultat plus général, indépendant de toute méthode de résolution, nous assure que pour chaque mode propre $(\tilde{\lambda}, \mathbf{x})$ de $\mathbf{B}_m$, il existe un mode propre $(\tilde{\lambda}, \mathbf{u})$ de $\mathbf{A}_\sigma$ tel que:

•
$$|\lambda-\tilde{\lambda}|\leq\frac{\|\mathbf{r}\|_2^2}{\gamma}$$
$$|\sin(\mathbf{u}, \tilde{\mathbf{u}})|\leq\frac{\|\mathbf{r}\|_2}{\gamma} \text{ avec } \gamma=\min_{\lambda_i\neq\tilde{\lambda}}|\lambda_i-\tilde{\mathbf{u}}^T\mathbf{A}_\sigma\tilde{\mathbf{u}}|$$

•Donc, plus que la minimisation du résidu, le critère d'arrêt des méthodes de type Lanczos/Arnoldi $|\beta_m|<\varepsilon$ permet d'approcher au mieux le spectre originel. Ces majorations ne sont accessibles que dans le cas hermitien.
•Dans le cas général (et en particulier non normal) elles sont plus difficiles voire impossibles à construire sans information complémentaire (niveau de non-normalité, de «défectivité»..). L'estimation du résidu n'est alors plus le bon critère pour estimer la qualité des modes approximés.

Intéressons nous maintenant à la qualité de convergence de l'algorithme. Particularisons le théorème 2 pour cet algorithme. En bornant la norme d'erreur de $\left\|\left(\mathbf{I}-\mathbf{P}_m\right)\mathbf{u}\right\|_2$ on obtient les majorations suivantes:

Théorème 3
Soit $(\lambda_i, \mathbf{u}_i)$ le *i*ème mode propre dominant de $\mathbf{A}_\sigma$ diagonalisable, il existe alors un mode de Ritz $(\tilde{\lambda}_i, \tilde{\mathbf{u}}_i)$ tel que:

$$|\lambda_i-\tilde{\lambda}_i|\leq\frac{\alpha^2}{\beta}\left(\prod_{j<i}\frac{\lambda_j-\lambda_n}{\lambda_j-\lambda_i}\right)^2 T_{m-i}^{-2}(\gamma_i)$$
$$|\sin(\mathbf{u}_i, \tilde{\mathbf{u}}_i)|\leq\alpha\left(\prod_{j<i}\frac{\lambda_j-\lambda_n}{\lambda_j-\lambda_i}\right) T_{m-i}^{-1}(\gamma_i)$$

avec $\gamma_i=1+2\frac{\lambda_i-\lambda_{i+1}}{\lambda_{1+i}-\lambda_n}$, $T_{m-i}(x)$ le polynôme de Tchebycheff de degré $m-i$, β la constante du théorème 1 et $\alpha=\frac{\beta}{\|\mathbf{P}_m\mathbf{u}_i\|_2}\tan(\mathbf{q}_1, \mathbf{u}_i)$.

Preuve:
Cf. [LT86] pp554-557.

Ces majorations indiquent que:
•Si le vecteur initial n'a aucune contribution le long des vecteurs propres, on ne peut capturer lesdits modes ($\alpha\to+\infty$).

•On a **prioritairement convergence des modes extrêmes** et d'autant mieux que le spectre est **séparé** (ces méthodes sont moins sensibles aux clusters que les méthodes de types puissances itérées), sans tassement de valeurs propres (les fameux "clusters").
•**L'erreur décroît lorsque** m **augmente** (comme l'inverse du polynôme $\mathbf{T}_{m-i}(\mathbf{x})$, donc comme l'inverse d'une exponentielle).
•L'estimation sur les valeurs propres est meilleure que celle de leurs vecteurs propres associés.

Remarque:
•La convergence de la méthode a été étudiée par P. Kaniel puis améliorée (et corrigée) par S.C.Paige ; On trouvera une synthèse de ces études dans le papier de Y. Saad[Saa80].

Regardons maintenant comment se comporte l'algorithme en arithmétique finie. Nous allons voir qu'il est le siège de phénomènes surprenants.

6.3 Algorithme de Lanczos pratique

6.3.1 Problème d'orthogonalité

Lors de la mise en œuvre effective de cet algorithme, on s'aperçoit que très vite **les vecteurs de Lanczos perdent leurs orthogonalités**. La matrice de Rayleigh n'est alors plus la matrice projetée de l'opérateur initial et le spectre capturé est de plus en plus entaché d'erreurs. Longtemps ce phénomène inéluctable a été attribué à tort aux effets d'arrondi de l'arithmétique finie. En 1980, C.C.Paige montra que la perte d'orthogonalité était surtout **imputable à la convergence d'un mode propre**. Ainsi, dès qu'on capture un mode dominant, on perturbe l'agencement des vecteurs de Lanczos précédents. Le résultat suivant exprime clairement ce paradoxe.

Propriété 4 (Analyse de Paige)
En reprenant les notations de l'algorithme 9 et en notant ε la précision machine, à l'itération $k+1$ l'orthogonalité du nouveau vecteur de Lanczos s'exprime sous la forme:

$$|\mathbf{q}_{k+1}^T \mathbf{q}_i| = \frac{|\mathbf{v}^T \mathbf{q}_i| + \varepsilon \|\mathbf{A}_\sigma\|_2}{|\beta_k|} (i \le k)$$

Preuve:
Cf. [Pai00].

Le problème survient en fait lors de la normalisation du nouveau vecteur de Lanczos $\mathbf{q}_{k+1}$. Lorsque ce mode converge, d'après la propriété 2, cela provient de la petitesse du coefficient β_k et donc malgré l'orthogonalité théorique ($|\mathbf{v}^T \mathbf{q}_i| = 0 (i \le k)$), l'orthogonalité effective est loin d'être acquise ($|\mathbf{q}_{k+1}^T \mathbf{q}_i| >> \varepsilon (i \le k)$).

Le traitement numérique de ces problèmes a fait l'objet de nombreuses recherches depuis trente ans et de nombreuses stratégies palliatives ont été développées. Le choix de telle ou telle méthode dépend du type d'information spectrale requis et des moyens informatiques disponibles, la synthèse de JL.Vaudescal propose d'ailleurs un très bon survol de ces variantes:
•**Algorithme de Lanczos sans réorthogonalisation** , développé par J.K.Cullum et R.A.Willoughby[CW85] qui consiste à expurger le spectre calculé de ses modes fantômes en étudiant les entrelacements des valeurs propres de la matrice de travail et d'une de ses sous-matrices. Cette variante admet un faible surcoût calcul et mémoire pour déterminer les valeurs propres, mais elle complexifie (parfois grandement) la recherche des vecteurs propres.

•**Algorithmes de Lanczos avec réorthogonalisation totale** (B.N.Parlett) ou **sélective** (J.Scott) qui consistent à chaque pas à réorthogonaliser le dernier vecteur de Lanczos obtenu par rapport à tous les vecteurs déjà calculés ou simplement par rapport aux vecteurs de Ritz convergés (cette variante permet ainsi de piloter dynamiquement la perte d'orthogonalité admissible). Ces variantes sont beaucoup plus coûteuses en complexité calcul (les réorthogonalisations automatiques) et mémoires (stockage des vecteurs précédents) mais elles s'avèrent plus efficaces et plus robustes.

Dans la variante de Newmann-Pipano[NP77] (METHODE='TRI_DIAG') du *Code_Aster*, c'est la **stratégie de réorthogonalisation complète**[22] qui a été choisie: le vecteur q_{k+1} n'est donc pas tout à fait calculé comme dans l'algorithme théorique.

Remarques:
•Ces problèmes de perte d'orthogonalité surviennent avec d'autant plus d'acuité que la taille du sous-espace augmente. C'est un argument supplémentaire pour la mise en place d'un processus itératif limitant cette taille.
•Ces stratégies se déclinent vectoriellement ou par blocs, elles sont implantées dans des algorithmes simples ou itératifs. Ces derniers pouvant bénéficier de restarts explicites ou implicites, pilotés ou automatiques. La variante IRAM bénéficie ainsi d'un restart implicite calculé automatiquement.
•Le point central de ces algorithmes est constitué par la méthode d'orthogonalisation mise en place. Tout le processus est dépendant de son succès et de sa robustesse. Aux transformations orthogonales de type Housholder ou Givens, très dispendieuses mais très robustes, on préfère désormais les algorithmes de type Gram-Schmidt Itératifs (IGSM) qui sont un meilleur compromis entre efficacité et complexité calcul (cf. Annexe 2). C'est d'ailleurs ce choix qui a été fait pour les variantes de Lanczos/Arnoldi mises en place dans le code.
•La variante de réorthogonalisation sélective par rapport aux modes convergés revient à effectuer une déflation implicite par restriction sur l'opérateur de travail pour ne pas avoir à les recalculer (cf. § 4.1).

6.3.2 Capture des multiplicités

On a vu qu'en théorie l'algorithme de Lanczos ne permettait pas, quelle que soit sa stratégie de réorthogonalisation, la capture théorique de modes multiples. **En pratique**, pour une fois, les **effets d'arrondi viennent à la rescousse** et "saupoudrent" de petites composantes le long de quasiment tous les vecteurs propres. On peut donc **capturer des multiplicités**, cependant elles peuvent être erronées et nécessitent un post-traitement de vérification complémentaire.

Remarque:
•En fait, seule une version par blocs (G.Golub & R.Underwood 1979) peut nous permettre une détection correcte des multiplicités, du moins tant que la taille des blocs est suffisante. Cette version a été étendue (M.Sadkane[Sad93] 1993) à l'algorithme d'Arnoldi mais il serait dommage de se priver de la variante de Sorensen (IRAM) qui est plus efficace et plus robuste.

6.3.3 Phénomène de Lanczos

Le succès de cet algorithme reposait au départ sur ce qu'on appelle le «**phénomène de Lanczos**». Cette conjecture prédit que pour une taille de sous-espace suffisamment grande ($m \gg n$) on est capable de détecter tout le spectre (à charge par la suite de distinguer le «bon grain de l'ivraie») de l'opérateur de travail. Compte tenu des faibles pré-requis mémoire d'un Lanczos «basique» (*grosso modo* une matrice tridiagonale et quelques vecteurs), ceci est particulièrement intéressant pour traiter des systèmes creux de très grandes tailles (les autres algorithmes de type puissances itérées ou QR nécessitent eux la connaissance de toute la matrice de travail).

Nous allons maintenant détailler (un peu) quelques éléments dont l'intérêt dépasse largement le cadre de cet algorithme.

6.4 Traitements complémentaires

6.4.1 Détection d'espaces invariants

Dans l'algorithme 8, la nullité du coefficient β_{l-1} empêche la normalisation du nouveau vecteur et requiert un traitement annexe dit de **déflation** . Le sous-espace de Krylov calculé est alors un sous- **espace invariant** et son spectre est donc exact. En d'autres mots, les $l-1$ premières valeurs propres exhumées par l'algorithme de soutien (QR ou QL) sont celles de $\mathbf{A}_\sigma$.

22 En GEP comme en QEP, on utilise cette stratégie de réorthogonalisation complète (pour Lanczos et IRAM).

Pour continuer, il faut alors choisir un nouveau vecteur aléatoire respectant les conditions limites, l'orthogonaliser avec tous les vecteurs de Lanczos déjà calculés et construire une deuxième famille de vecteurs de Lanczos qu'on orthogonalise par rapport à la première famille constituant l'espace:

$$\mathbf{H}_1=\mathbf{K}_1\left(\mathbf{A}_\sigma,\mathbf{q}_1\right)=Vect\left(\mathbf{q}_1,\mathbf{A}_\sigma\mathbf{q}_1,\ldots,\mathbf{A}_\sigma^{l-1}\mathbf{q}_1\right)$$

La matrice tridiagonale obtenue a la forme suivante:

$$\mathbf{B}_m=\begin{bmatrix}\alpha_1 & \bar{\beta}_1 & & & & & \\ \beta_1 & \alpha_2 & \ldots & & & 0 & \\ & \ldots & \ldots & \simeq 0 & & & \\ & & \simeq 0 & \alpha_l & \bar{\beta}_l & & \\ & & & \beta_l & \alpha_{l+1} & \ldots & \\ & 0 & & & \ldots & \ldots & \bar{\beta}_{m-1} \\ & & & & & \beta_{m-1} & \alpha_m\end{bmatrix}$$

La factorisation de Lanczos intermédiaire (à l'ordre l) s'écrit:

$$\mathbf{A}_\sigma\mathbf{Q}_l=\mathbf{Q}_l\mathbf{B}_l$$

Les valeurs $(\alpha_k,\beta_k)_{k=1,l}$ sont obtenues à partir du premier vecteur aléatoire et les valeurs suivantes $(\alpha_k,\beta_k)_{k=l+1,m}$ *via* un deuxième vecteur aléatoire. Pour détecter la nullité du terme extra-diagonal on utilise le critère numérique

$$|\beta_{l-1}|\leq PREC\_LANZOS|\alpha_l| \quad \text{alors} \quad \beta_{l-1}=0 \ ,$$

où PREC_LANCZOS est initialisé sous le mot-clé facteur CALC_FREQ.

Remarques:
•Un critère plus robuste retenu par les librairies mathématiques EISPACK, LAPACK et ARPACK utilise le terme diagonal précédent, un paramètre modulable c et la précision machine ε ,

$$|\beta_{l-1}|<c\,\varepsilon\left(|\alpha_{l-1}|+|\alpha_l|\right)$$

•c'est ce critère qui à été retenu pour IRAM (mais l'utilisateur ne peut modifier le paramètre c , celui-ci est fixé à une valeur standard par le code), différents messages prévenant d'ailleurs l'utilisateur de la détection d'un espace invariant.
•L'obtention d'un tel espace est tout à fait sympathique puisqu'elle nous assure de la très bonne qualité des premiers modes. De plus, on réduit la taille du problème en le scindant en deux parties: l'une résolue, l'autre à résoudre. De nombreuses techniques cherchent à reproduire artificiellement ce phénomène (cf. §4.1).

6.4.2 Stratégies de redémarrages

Pour **limiter les problèmes d'orthogonalisation** endémiques, **borner les pré-requis mémoire** et prendre en compte **dynamiquement l'information spectrale** déjà obtenue, des stratégies de redémarrages ont été couplées à l'algorithme de Lanczos. Il perd son caractère «simple» et devient «itératif». On itère les «restarts» jusqu'à satisfaire les critères de convergence souhaités. L'algorithme ne subit pas la convergence des modes imposés par le théorème 8 et il peut favoriser intéractivement celle de certains modes.

En théorie cependant, plus la taille du sous-espace est grande, meilleure est la convergence. Un compromis est donc à trouver entre la taille du sous-espace et la fréquence des redémarrages. Différents vecteurs de redémarrages peuvent être utilisés, s'écrivant généralement comme une somme pondérée des p modes propres recherchés (Y.Saad 1980)

$$\mathbf{q}_1=\sum_{i=1}^{p}\chi_i\mathbf{Q}_m\mathbf{x}_i$$

En effet, si on redémarre avec un vecteur initial appartenant au sous-espace invariant engendré par les modes recherchés, on est alors sûr d'obtenir (avec un bon algorithme d'orthogonalisation) une norme résiduelle de l'ordre de la précision machine et des modes propres presque exacts. Outre la décision de déclencher le redémarrage, toute la problématique réside dans la recherche de ces poids χ_i . Nous verrons qu'avec IRAM, les restarts mis en place via des filtres polynomiaux implicites résolvent élégamment et automatiquement cette question.

Remarques:
•Cette philosophie des restarts a été initiée par W.Karush(1951).
•Les redémarrages basés sur les vecteurs de Schur associés à la décomposition spectrale recherchée semblent plus stables (J.Scott 1993).
•Des critères d'efficacité ont été recherchés pour décider de l'opportunité d'un restart (B.Vital 1990).
•Au lieu d'une simple combinaison linéaire de vecteurs propres, on peut déterminer un vecteur de redémarrage via des polynômes (Tchebychev en réel et Faber en complexe) permettant d'axer la recherche modale dans telle ou telle zone. On parle alors d'accélérations polynomiales explicites[Sad93].

Le paragraphe suivant va reprendre certains des concepts présentés jusqu'alors pour expliciter la variante mise en place dans le code.

6.5 Implantation dans *Code_Aster*

6.5.1 Variante de Newmann & Pipano

Cette variante développée par M.Newmann & A.Pipano[CW85][NP77] en 1977, est une méthode de Lanczos simple, en arithmétique réelle, avec réorthogonalisation totale (*via* un GSM). Elle utilise le «shift and invert» classique mâtiné du pseudo-produit scalaire introduit par la matrice décalée $(\mathbf{A}-\sigma\mathbf{B})$

$$\underbrace{(\mathbf{A}-\sigma\mathbf{B})}_{\mathbf{A}_\sigma}^{-1}\mathbf{B}\,\mathbf{u}=\frac{1}{\underbrace{\mu-\sigma}_{\lambda}}\mathbf{u}$$

avec :

$$\left|\begin{array}{c}(\mathbf{x},\mathbf{y})=\mathbf{y}^{\mathbf{t}}(\mathbf{A}-\sigma\mathbf{B})\mathbf{x},\\ \delta=sign(\mathbf{x},\mathbf{x}),\\ \|\mathbf{x}\|=\delta|(\mathbf{x},\mathbf{x})|^{\frac{1}{2}}.\end{array}\right.$$

Ce choix rend le couple «opérateur de travail-produit scalaire» symétrique et il est adapté aux matrices particulières de la mécanique des structures. On peut ainsi traiter des problèmes:
•De structure libre et de fluide structure ($\mathbf{A}$ peut être singulière).
•De flambement ($\mathbf{B}$ est indéfinie).
Le pseudo-produit scalaire introduit par la matrice décalée est ainsi régulier (σ répond à cette prérogative) et il permet de chercher des vecteurs de Lanczos $(\mathbf{A}-\sigma\mathbf{B})$ -orthogonaux. La réorthogonalisation totale ne s'effectue que si elle s'avère nécessaire et ce critère est paramétrable.
Le prix à payer pour ce gain en robustesse est la perte de symétrie possible de la matrice de Rayleigh.

$$\mathbf{B}_m=\begin{bmatrix}\alpha_1 & \gamma_1 & 0 & 0\\ \beta_1 & \alpha_2 & \dots & 0\\ 0 & \dots & \dots & \gamma_{m-1}\\ 0 & 0 & \beta_{m-1} & \alpha_m\end{bmatrix} \text{ avec } \left|\begin{array}{c}\gamma_i=\delta_i\beta_i\\ \delta_i=sign(\mathbf{q}_{i+1},\mathbf{q}_{i+1})\end{array}\right.$$

Après avoir été équilibrée et mise sous forme de Hessenberg supérieure, $\mathbf{B}_m$ est diagonalisée par une méthode QL implicite si elle reste malgré tout symétrique ou par une méthode QR sinon (cf. Annexe 1). La différence en coût calcul entre ces solveurs reste négligeable face aux coûts des réorthogonalisations. Le schéma de Lanczos implanté devient alors:

Tirage aléatoire de $\mathbf{q}_{init}$.
$\tilde{\mathbf{q}}_1=\left(\mathbf{A}-\sigma\mathbf{B}\right)^{-1}\left(\mathbf{q}_{init}(i).\mathbf{u}_{lagr}(i)\right)_i$.
$\hat{\mathbf{q}}_1=\mathbf{A}_\sigma\left(\tilde{\mathbf{q}}_1(i).\mathbf{u}_{bloq}(i)\right)_i$.
$\delta_1=sign\left(\hat{\mathbf{q}}_1^T\left(\mathbf{A}-\sigma\mathbf{B}\right)\hat{\mathbf{q}}_1\right)$
$\mathbf{q}_1=\delta_1\hat{\mathbf{q}}_1\left|\hat{\mathbf{q}}_1^T\left(\mathbf{A}-\sigma\mathbf{B}\right)\hat{\mathbf{q}}_1\right|^{-\frac{1}{2}}$
$\mathbf{q}_0=0,\beta_0=0,d_0=0$.
Pour k=1,m faire
$\mathbf{z}=\mathbf{A}_\sigma\mathbf{q}_k-\delta_{k-1}\beta_{k-1}\mathbf{q}_{k-1}$,
$\alpha_k=\mathbf{q}_k^T\left(\mathbf{A}-\sigma\mathbf{B}\right)\mathbf{z}$,
$\mathbf{v}=\mathbf{z}-\delta_k\alpha_k\mathbf{q}_k$,
Réothogonalisation de $\mathbf{v}$ par rapport aux $\left(\mathbf{q}_i\right)_{i=1,k}$(IGSM),
$\beta_k=\mathbf{v}^{\mathbf{T}}\left(\mathbf{A}-\sigma\mathbf{B}\right)\mathbf{v}$,
$\delta_k=sign\left(\mathbf{v}^{\mathbf{T}}\left(\mathbf{A}-\sigma\mathbf{B}\right)\mathbf{v}\right)$,
Si $\beta_k\neq0$ alors
$\mathbf{q}_{k+1}=\frac{\delta_k\mathbf{v}}{\left|\beta_k\right|^{\frac{1}{2}}}$,
Sinon
Déflation ;
Fin si.
Fin boucle.

Algorithme 10. Méthode de Lanczos. Variante de Newman-Pipano.

Remarques:
•La $\mathbf{B}$ -orthogonalité et à fortiori celle au sens euclidien ne peuvent plus être que le fruit de configurations particulières. Ceci ne modifie pas les propriétés d'orthogonalités des modes propres (cf. proposition 2).
•Comme on l'a déjà signalé il existe toute une zoologie de couples «opérateur-produit scalaire», celui-ci n'étant qu'une possibilité parmi d'autres. Ainsi, les bibliothèques[Arp] classiques proposent de traiter les problèmes de flambement en «buckling mode» via le même «shift and invert» et le pseudo-produit scalaire introduit par $\mathbf{A}$. Mais du fait de l'introduction quasi-systématique de Lagranges, cette matrice devient indéfinie voire singulière, ce qui perturbe grandement le processus. Les mêmes causes produisent les mêmes effets lorsque pour un calcul de dynamique, on utilise le $\mathbf{B}$ -produit scalaire.

La réorthogonalisation s'effectue grâce à une variante «maison» de la Méthode de Gram-Schmidt Itératif (IGSM) suivant le processus suivant:

Pour $i=1,k$
$a=\mathbf{q}_{k+1}^T(\mathbf{A}-\sigma\mathbf{B})\mathbf{q}_i$,
Si $\left|\langle\mathbf{q}_{k+1}^T(\mathbf{A}-\sigma\mathbf{B})\mathbf{q}_i\rangle\right|\geq PREC\_ORTHO$ alors
Pour $j=1$,NMAX_ITER_ORTHO
$\mathbf{x}=\mathbf{q}_{k+1}-\left(\mathbf{q}_{k+1}^T(\mathbf{A}-\sigma\mathbf{B})\mathbf{q}_i\right)\mathbf{q}_i$,
$b=\mathbf{x}^{\mathbf{T}}(\mathbf{A}-\sigma\mathbf{B})\mathbf{q}_i$

Si $(|b| \leq PREC\_ORTHO)$ alors
$\mathbf{q}_{k+1} = \mathbf{x}$,
$i+1 \Rightarrow i$, exit;
Sinon
Si $b \leq a$ alors
$\mathbf{q}_{k+1} = \mathbf{x}$,
$a = b$;
Sinon
Échec, émission d'un message d'alarme,
$i+1 \Rightarrow i$, exit;
Fin si.
Fin si.
Fin boucle en j .
Fin si.
Fin boucle en i .

Algorithme 11. Procédure de réorthogonalisation de 'TRI_DIAG'.

Remarque:
•Après quelques tests, il semble que cette variante spécifique au code soit moins efficace que l'IGSM de type Kahan-Parlett[Par80] choisi pour l'IRAM (cf. Annexe 2).

6.5.2 Paramétrage

Pour pouvoir activer cette méthode, il faut initialiser le mot-clé METHODE à 'TRI_DIAG'. D'autre part, la taille du sous-espace de projection est déterminée, soit par l'utilisateur, soit empiriquement à partir de la formule:

$$m = min(max(4p, p+7), n_{ddl-actifs})$$

où p est le nombre de valeurs propres à calculer,
$n_{ddl-actifs}$ est le nombre de degrés de liberté actifs du système (cf. §3.2)

L'utilisateur peut toujours imposer lui-même la dimension en l'indiquant avec le mot-clé DIM_SOUS_ESPACE du mot-clé facteur CALC_FREQ. On peut préférer à une valeur par défaut, un coefficient multiplicatif par rapport au nombre de fréquences contenues dans l'intervalle d'étude. Ce type de fonctionnalité concerne le mot-clé COEF_DIM_ESPACE.

Les paramètres de la réorthogonalisation totale, PREC_ORTHO et NMAX_ITER_ORTHO, le nombre maximal d'itérations QR, NMAX_ITER_QR, et le critère de déflation, PREC_LANCZOS, sont accessibles par l'utilisateur sous le mot-clé facteur CALC_FREQ. Lorsque ce phénomène de déflation se produit, un message spécifique précise son rang 1.

6.5.3 Avertissement sur la qualité des modes

Rappelons que cette variante n'est pas itérative. A partir du nombre de fréquences souhaitées, on estime la dimension du sous-espace de calcul et on espère que les modes propres du problème standard projeté seront une bonne approximation de ceux du problème généralisé. On vérifie *a posteriori* la validité des résultats (cf. §3.5) mais on n'a aucun contrôle actif sur cette précision.
S'ils ne sont pas satisfaisants, on n'a comme seul recourt que d'effectuer un nouvel essai en augmentant la dimension du sous-espace.

Remarque:
•La méthode IRA que nous allons aborder propose un contrôle modulable et dynamique de la précision des résultats, c'est une méthode itérative.

6.5.4 Détection de modes rigides

Une **méthode algébrique calculant les valeurs propres nulles** du problème modal généralisé a été introduite. Physiquement celles-ci correspondent aux mouvements à énergie de déformation nulle d'une structure libre (cf. [BQ00] TP n°2). Hormis les difficultés numériques de manipulation de quantités quasi-nulles, du fait de leurs multiplicités, leur capture correcte était jusqu'à présent souvent problématique pour le Lanczos implanté dans *Code_Aster*: des modes fantômes apparaissaient correspondant à des multiplicités ratées !
Cet algorithme de détection des modes de corps rigide, que l'on active en initialisant OPTION à 'MODE_RIGIDE' (valeur par défaut 'SANS'), intervient en pré-traitement du calcul modal proprement dit (uniquement avec 'TRI_DIAG'). Elle est basée sur l'analyse de la matrice de rigidité et se décompose en trois phases:
•Détection des pivots nuls de cette matrice.
•Blocage de ces pivots.
•Résolution d'un système linéaire dont sont solutions les vecteurs propres associés.

Lors du processus de Lanczos il suffit dès lors d'orthogonaliser, au fur et à mesure de leur détermination, les vecteurs de base avec ces derniers.
Cependant, l'introduction de la méthode IRA réduit l'intérêt (si ce n'est à titre de comparaison) d'une telle option (qui n'est pas gratuite en complexité calcul puisqu'elle requiert des inversions de systèmes). A quoi bon se priver d'un tel algorithme qui n'est nullement affecté par la présence de ces modes multiples un peu particuliers !

Cette méthode reste néanmoins utile à titre de solution de secours en cas de défaillance d'IRAM. On peut aussi envisager de l'encapsuler dans un opérateur auxiliaire (comme INFO_MODE).

Remarque:
•Cette option 'MODE_RIGIDE' n'est activable qu'avec Lanczos pour des GEP à modes réels.

6.6 Périmètre d'utilisation

GEP à matrices symétriques réelles.
L'utilisateur peut spécifier la classe d'appartenance de son calcul (dynamique ou flambement) en initialisant le mot-clé TYPE_RESU. Suivant cette valeur, on renseigne le vecteur FREQ ou CHAR_CRIT.

6.7 Affichage dans le fichier message

L'exemple ci-dessous issu de la liste de cas tests du code (sdll112a) récapitule l'ensemble des traces gérées par l'algorithme. Le nombre d'itérations QR effectives (ou QL) ne peut être qu'identique pour toutes les valeurs propres. C'est un artefact d'information qui sera amené à disparaître.

```
-----------------------------------------------------------------------
LE NOMBRE DE DDL
TOTAL EST:                      86
DE LAGRANGE EST:                20
LE NOMBRE DE DDL ACTIFS EST:        56
-----------------------------------------------------------------------
L'OPTION CHOISIE EST: PLUS_PETITE
LA VALEUR DE DECALAGE EN FREQUENCE EST :  0.00000E+00
-----------------------------------------------------------------------
INFORMATIONS SUR LE CALCUL DEMANDE :
 NOMBRE DE MODES DEMANDES     :  10
LA DIMENSION DE L'ESPACE REDUIT EST :  0
 ELLE EST INFERIEURE AU NOMBRE DE MODES, ON LA PREND EGALE A  40
-----------------------------------------------------------------------
LES FREQUENCES CALCULEES INF. ET SUP. SONT:
      FREQ_INF :  1.54569E+01
      FREQ_SUP :  1.01614E+02
LA PREMIERE FREQUENCE SUPERIEURE NON RETENUE EST:  1.29375E+02
-----------------------------------------------------------------------
       CALCUL MODAL:  METHODE D'ITERATION SIMULTANEE
                      METHODE DE LANCZOS
NUMERO    FREQUENCE (HZ)    NORME D'ERREUR    ITER_QR
         1      1.54569E+01       1.29798E-12          4
         2      1.54569E+01       7.15318E-13          4
```

```
          3      3.35823E+01       3.98618E-13          4
          4      3.35823E+01       4.25490E-12          4
         ...
         10      1.01614E+02       3.18663E-12          4
 ---------------------------------------------------------------------
 VERIFICATION A POSTERIORI DES MODES
 DANS L'INTERVALLE ( 1.54182E+01, 1.16326E+02)
    IL Y A BIEN   10 FREQUENCE(S)
 ---------------------------------------------------------------------
```

Exemple 5. Trace de MODE_ITER_SIMULT dans le fichier message avec METHODE='TRI_DIAG'.

6.8 Récapitulatif du paramétrage

Récapitulons maintenant le paramétrage disponible de l'opérateur MODE_ITER_SIMULT avec ce solveur modal.

Opérande	Mot-clé	Valeur par défaut	Références
	TYPE_RESU ='DYNAMIQUE'	'DYNAMIQUE'	§3.1
	'MODE_FLAMB '		§3.1
	METHODE='TRI_DIAG'	'SORENSEN'	§6.5
	OPTION='MODE_RIGIDE'	'SANS'	§6.5
	'SANS'		§6.5
CALC_FREQ	**FREQ (si** CENTRE **)**		§5.4
	CHAR_CRIT (si CENTRE **)**		§5.4
	OPTION ='PLUS_PETITE'	'PLUS_PETITE' '	§5.4
	'BANDE'		§5.4
	'CENTRE'		§5.4
	NMAX_FREQ	10	§5.4
	DIM_SOUS_ESPACE COEF_DIM_ESPACE	Calculé	§6.5
	PREC_ORTHO	1.E-12	§6.5
	NMAX_ITER_ORTHO	1.E-04	§6.5
	PREC_LANCZOS	1.E-04	§6.5
	NMAX_ITER_QR	15	§6.5
	NMAX_ITER_SHIFT	3	[Boi12] §3.2
	PREC_SHIFT	0.05	[Boi12] §3.2
	SEUIL_FREQ	1.E-02	§3.7
VERI_MODE	STOP_ERREUR='OUI'	'OUI'	§3.7
	'NON'		§3.7
	PREC_SHIFT	5.E-03	§3.7
	SEUIL	1.E-06	§3.7
	STURM='OUI'	'OUI'	§3.7
	'NON'		§3.7

Tableau 6.8-1. Récapitulatif du paramètrage de MODE_ITER_SIMULT (GEP) avec METHODE='TRI_DIAG'.

Remarques:
•On retrouve toute la "tripaille" de paramètres liée aux pré-traitements du test de Sturm (NMAX_ITER_SHIFT, PREC_SHIFT) et aux post-traitements de vérification (SEUIL_FREQ, VERI_MODE).

•Lors des premiers passages, il est fortement conseillé de ne modifier que les paramètres principaux notés en gras. Les autres concernent plus les arcanes de l'algorithme et ils ont été initialisés empiriquement à des valeurs standards.
•En particulier, pour améliorer la qualité d'un mode, le seul paramètre modulable est la dimension du sous-espace (DIM_SOUS_ESPACE/COEF_DIM_ESPACE).

7 Algorithme IRA (METHODE='SORENSEN')

7.1 Introduction

Nous avons vu qu'un des problèmes cruciaux de la méthode de Lanczos est la perte d'orthogonalité inéluctable de ses vecteurs de base. Une **généralisation** de cet algorithme **au cas non hermitien** imaginée par **W.E.Arnoldi**[Arn51] **en 1951** permet de résoudre partiellement cette problématique. Elle a été remise au goût du jour par Y.Saad[Saa80] en 1980 et une abondante littérature couvre le sujet. Mis à part l'article fondateur et les papiers de Y.Saad et M.Sadkane[Sad93], on recommande la synthèse actualisée et exhaustive de J.L.Vaudescal[Vau00].

Cette méthode étant à la base de **l'algorithme IRAM dit «de Sorensen»** (IRAM pour 'Implicit Restarted Arnoldi Method'), nous allons tout d'abord détailler son fonctionnement, ses comportements et ses limitations. Par la suite, nous cernerons les enjeux auxquels doit répondre IRAM (et elle le fait dans la plupart des cas standards !) et nous nous pencherons sur ses arcanes théoriques et numériques. Nous conclurons par le récapitulatif de son paramétrage effectif dans *Code_Aster* et par un exemple de fichier message.

7.2 Algorithme d'Arnoldi

7.2.1 Principe

Son périmètre d'application recouvre **tous les couples «opérateur de travail-(pseudo)produit scalaire»**. Le pendant de cette ouverture est le remplissage de la matrice de Rayleigh qui devient de la forme **Hessenberg supérieure**. Ce n'est pas très préjudiciable, car on peut ainsi lui appliquer directement l'algorithme QR**.** La première étape d'un bon QR, hormis l'équilibrage, consiste justement à réduire la matrice de travail sous forme de Hessenberg. Cela permet de gagner un ordre de magnitude lors de la résolution proprement dite (cf. Annexe 1).
L'algorithme est très similaire à celui de Lanczos, il consiste à construire progressivement une famille de vecteur d'Arnoldi $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_m$ en projetant orthogonalement, à l'itération k , le vecteur $\mathbf{A}_\sigma \mathbf{q}_k$ sur les k vecteurs précédent. Le nouveau vecteur devient $\mathbf{q}_{k+1}$ et ainsi, de proche en proche, on assure l'orthogonalité de cette famille de vecteurs.
A la différence de Lanczos, l'orthogonalité du nouveau vecteur par rapport à tous les précédents est donc assurée explicitement et non implicitement. Celle-ci est gérée par l'algorithme de Gram-Schmidt Modifié (GSM) (cf. annexe 2) qui s'avère suffisamment robuste dans la plupart des cas.

En notant, $\mathbf{e}_m$ le m ième vecteur de la base canonique, le **vecteur résidu de la factorisation d'Arnoldi** s'écrit $\mathbf{R}_m = b_{m+1,m}\,\mathbf{q}_{m+1}\mathbf{e}_m^T$.

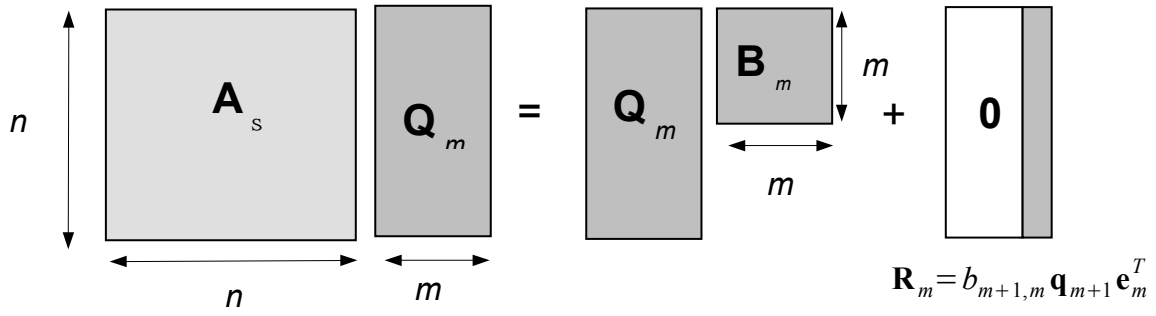


Figure 7.2-1. Factorisation d'Arnoldi.

Le processus itératif se résume comme suit:

Calcul de $\mathbf{q}_1/\|\mathbf{q}_1\|=1$.
Pour $k=1,m$ faire
  $\mathbf{z}=\mathbf{A}_\sigma \mathbf{q}_k$,
  Pour $l=1,k$ faire (GSM)
    $b_{lk}=(\mathbf{z},\mathbf{q}_l)$,
    $\mathbf{z}=\mathbf{z}-b_{lk}\mathbf{q}_l$,
  Fin boucle;
  $b_{k+1,k}=\|\mathbf{z}\|$,
  Si $b_{k+1,k}\neq 0$ alors
    $\mathbf{q}_{k+1}=\dfrac{\mathbf{v}}{b_{k+1,k}}$,
  Sinon
    Déflation ;
  Fin si.
Fin boucle.

Algorithme 13. Arnoldi théorique.

La **matrice de Rayleigh** s'écrit alors:

$$\mathbf{B}_m=\begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1m} \\ b_{21} & b_{22} & \cdots & b_{2m} \\ 0 & \cdots & \cdots & b_{m-1,m} \\ 0 & 0 & b_{m,m-1} & b_{mm} \end{bmatrix}$$

Hormis la forme de cette matrice et une moindre acuité aux problèmes d'orthogonalité, **cet algorithme nous assure les mêmes propriétés théoriques et numériques que Lanczos**. Cependant si on laisse croître indéfiniment la taille du sous-espace jusqu'à convergence des valeurs propres souhaitées, les effets d'arrondi vont malgré tout reprendre le dessus et on va au devant de gros ennuis. D'où la nécessité, comme pour Lanczos, de rendre ce processus itératif via des redémarrages.

Remarques:
•Cette méthode peut être vue comme une variante implicite de l'algorithme de Lanczos avec réorthogonalisation totale.
•L'orthogonalisation implicite de l'algorithme peut être conduite par des algorithmes plus coûteux mais plus robustes tels que QR ou IGSM. Ceci est fortement requis lorsque l'opérateur de travail présente un trop fort défaut de normalité.
•Du fait de la structure de $\mathbf{B}_m$, la complexité mémoire est plus sollicitée qu'avec Lanczos, par contre la complexité calcul reste du même ordre de grandeur $O(nm(c+3+m))$ (avec c le nombre moyen de termes non nuls sur les lignes de la matrice de travail).
•Pour améliorer ce premier point, Y.Saad[Saa80] a montré que la structure de Hessenberg supérieure peut voir s'annuler ses sur-diagonales extrêmes si on n'effectue que partiellement la réorthogonalisation.
•Les algorithmes vectoriels ayant une tendance naturelle à rater des multiplicités, on préfère souvent utiliser une version blocs (M.Sadkane 1993). Mais la taille de ceux-ci influent sur la qualité des résultats, c'est pour cette raison qu'on leur préfère les versions vectorielles ou blocs d'IRAM.
•Le choix du vecteur d'Arnoldi initial s'effectue de la même manière que pour Lanczos.

7.2.2 Estimations d'erreurs et de convergence

En ce qui concerne l'évaluation de la qualité d'approximation des modes propres obtenus, on a un critère aussi simple et efficace que pour Lanczos.

Propriété 5
La norme euclidienne du résidu de l'élément de Ritz $\left(\tilde{\lambda}, \tilde{\mathbf{u}}=\mathbf{Q}_m\mathbf{x}\right)$ est égale à

$$\|\mathbf{r}\|_2=\|(\mathbf{A}_\sigma-\tilde{\lambda}\mathbf{I})\tilde{\mathbf{u}}\|_2=|b_{m+1,m}||\mathbf{e}_m^T\mathbf{x}|$$

Preuve:
Triviale en prenant la norme euclidienne de la factorisation d'Arnoldi $\mathbf{A}_\sigma\mathbf{Q}_m\mathbf{x}=\mathbf{Q}_m\mathbf{B}_m\mathbf{x}+b_{m+1,m}\mathbf{q}_{m+1}\mathbf{e}_m^T\mathbf{x}$ et comme $\mathbf{q}_{m+1}$ est normé à l'unité.

Toujours en nous focalisant sur la norme du projecteur supplémentaire $\|\left(\mathbf{I}-\mathbf{P}_m\right)\mathbf{u}\|_2$ on peut généraliser le théorème de convergence 3 au cas non hermitien.

Théorème 6
Soit $\left(\lambda_1, \mathbf{u}_1\right)$ le premier (rangement classique, par ordre décroissant de module) mode propre dominant de $\mathbf{A}_\sigma$ diagonalisable et soit $\mathbf{q}_1=\sum_{k=1}^{n}\alpha_k\mathbf{u}_k$ le vecteur initial d'Arnoldi décomposé sur la base de vecteurs propres, il existe alors un mode de Ritz $\left(\tilde{\lambda}_1, \tilde{\mathbf{u}}_1\right)$ tel que: $|\lambda_1-\tilde{\lambda}_1|\leq\frac{\alpha^2}{\beta}\xi_1\delta_1^m$
avec $\alpha=\frac{\beta}{\|\mathbf{P}_m\mathbf{u}_i\|_2}$, β la constante du théorème 1, $\xi_1=\sum_{k=1,k\neq1}^{n}|\frac{\alpha_k}{\alpha_1}|$ et $\delta_1^m=\left(\sum_{j=2}^{m+1}\prod_{k=2,k\neq j}^{m+1}\left|\frac{\lambda_k-\lambda_1}{\lambda_k-\lambda_j}\right|\right)^{-1}$
. Ce résultat se décline de la même manière sur les autres modes.

Preuve:
En reprenant la démonstration de Y.Saad ([Saa91] pp209-210) et le résultat du théorème 1.

Ces majorations très différentes de celles obtenues avec Lanczos guident cependant les mêmes phénomènes:
•Si le vecteur initial n'a aucune contribution le long des vecteurs propres recherchés, on ne peut les capturer ($\xi_i\to+\infty$).
•On a **prioritairement convergence des modes périphériques** du spectre, et ce, d'autant mieux qu'il est séparé (propriété de λ_i^m).
•La décroissance de l'erreur est proportionnelle à l'augmentation de m (propriété de λ_i^m).

Remarques:
•Lorsqu'une valeur propre est mal conditionnée $\xi_i\to+\infty$ alors il faut augmenter m pour λ_i^m que décroisse.
•Des résultats analogues ont été vus dans le cas d'un opérateur défectif (cf.Zia 94).

Fort de ces enseignements, nous allons maintenant récapituler les enjeux auxquels doit répondre IRAM.

7.3 Les enjeux

L'algorithme IRA tente d'apporter un remède élégant aux problèmes récurrents soulevés par les autres approches:
•**Minimisation** de l' **espace de projection**: il propose à minima $m>p+1$ au lieu des $m=4p$ de Lanczos.
•Gestion optimale des surcoûts d'orthogonalisation établissant un **compromis** entre la **taille du sous-espace** et la **fréquence des redémarrages**.
•Gestion transparente, dynamique et efficace de ces restarts.
•Prise en compte automatique de **l'information spectrale** .
•Fixation des **pré-requis mémoire** et de la **qualité** des **résultats** .

On a donc plus de question à se poser concernant la stratégie de réorthogonalisation, la fréquence des restarts, leurs implantations, les critères de détection d'éventuels modes fantômes... «super-IRAM» se charge de tout !

En bref, il procure:
•Une meilleure **robustesse** globale.
•Des **complexités calculs** $O(4\mathrm{nm}(m-p))$ et **mémoires** $O(2\mathrm{np}+p^2)$ **améliorées** (surtout par rapport à un Lanczos simple tel celui de Newmann & Pipano) pour **une précision fixée**.
•Une capture plus rigoureuse des multiplicités, des clusters et des modes de corps rigides (donc **moins de modes parasites** !).

Remarques:
•Sur ce dernier point, seule une version par blocs d'IRAM ou une version incorporant du «purge and lock» (techniques de capture et de filtrage, cf. D.Sorensen & R.B.Lehoucq, 1997) peuvent nous garantir une détection correcte du spectre d'un opérateur standard (i.e. pas trop mal conditionné).
•Il semble, qu'en pratique dans Code_Aster, le rapport en complexité calcul entre IRAM et Lanczos/Bathe & Wilson soit à minima d'ordre 2 en faveur du premier. Avec des tailles de problèmes raisonnables (quelques dizaines de milliers de ddls et $p=O(100)$) celui-ci peut monter jusqu'à 10 (sans l'encapsulation de MACRO_MODE_MECA). Dans certains cas semi-industriels, il a permis de dérouler une recherche de spectre qui avait échoué avec Jacobi et qui était inabordable avec Lanczos (compte tenu des délais impartis).
•Une classe d'algorithme dit de «Jacobi-Davidson» (cf. R.B.Morgan 1990) semble encore plus prometteuse pour traiter des cas pathogènes. Elle utilise un algorithme de type Lanczos qu'elle préconditionne via une méthode de Rayleigh.

Dans le paragraphe suivant nous allons expliciter le fonctionnement d'IRAM.

7.4 Algorithme 'Implicit Restarted Arnoldi' (IRA)

Cet algorithme a été initié par D.C.Sorensen[Sor92] en 1992 et connaît un réel essor pour la résolution de grands systèmes modaux sur des super-calculateurs parallèles. Son cadre d'application est tout à fait général. Il traite aussi bien les problèmes réels que complexes, hermitiens ou non. Il se résume en une succession de factorisations d'Arnoldi dont les résultats pilotent automatiquement des redémarrages statiques et implicites, via des filtres polynomiaux modélisés par des QR implicites.

Tout d'abord il réalise une **factorisation d'Arnoldi d'ordre** $m=p+q$ (en théorie $q=2$ suffit, en pratique $q=p$ est préférable. C'est d'ailleurs cette dernière valeur par défaut qui a été retenue (cf. §7.5) de la matrice de travail. Puis une fois ce pré-traitement effectué il **itère un processus de filtrage de la partie du spectre indésirable** (numériquement et méthodologiquement, il est plus facile d'exclure que d'incorporer).

Il commence par déterminer le spectre de la matrice de Rayleigh (*via* l'indétronable QR) et il en dissocie la partie non désirée (en se référant aux critères fixés par l'utilisateur) qu'il utilise ensuite comme shift pour mettre en place une série de q QR implicite avec shift simple (cf. Annexe 1). La factorisation s'écrit alors:

$$\mathbf{A}_\sigma \mathbf{Q}_m^+ = \mathbf{Q}_m^+ \mathbf{B}_m^+ + \mathbf{R}_m \mathbf{Q}$$

où $\mathbf{Q}_m^+ = \mathbf{Q}_m \mathbf{Q}$, $\mathbf{B}_m^+ = \mathbf{Q}^T \mathbf{B}_m \mathbf{Q}$ et $\mathbf{Q}=\mathbf{Q}_1 \mathbf{Q}_2 .... \mathbf{Q}_q$ la matrice unitaire associée aux QRs. Après avoir mis à jour les matrices, on les tronque jusqu'à l'ordre p

$$\mathbf{A}_\sigma \mathbf{Q}_p^+ = Q_p^+ \mathbf{B}_p^+ + \mathbf{R}_p^+$$

et ainsi, au prix de q nouvelles itérations d'Arnoldi, on peut retrouver une factorisation d'Arnoldi d'ordre m qui soit viable. Toute la subtilité du processus repose sur ce dernier enchaînement. Notons:

$$\Phi\left(\mathbf{A}_\sigma\right) = \prod_{i=1}^{q} \left(\mathbf{A}_\sigma - \tilde{\lambda}_{p+i} \mathbf{Id}\right) ,$$

le polynôme matriciel d'ordre q engendré par l'opérateur de travail. En fait, les QRs implicites ont agit sur les p premières lignes de la matrice d'Arnoldi, de Rayleigh et du résidu, **de manière à ce que la**

factorisation complémentaire d'ordre q produise le même effet qu'une factorisation d'ordre m initiée par le vecteur

$$\tilde{\mathbf{q}}_1 = \Phi\left(\mathbf{A}_\sigma\right)\mathbf{q}_1 \ .$$

Le vecteur initial a été implicitement modifié (il n'y a pas construction et application effective du polynôme) afin qu'il engendre préférentiellement les modes souhaités et ce, en soustrayant les composantes jugées «impropres». Comme on l'a déjà fait remarquer ce type de restart permet de diminuer le résidu en amoindrissant les composantes du vecteur initial suivant les modes indésirables. En arithmétique exacte, on aurait immédiatement $\mathbf{R}_m = \mathbf{0}$ et le processus s'arrêterait là !

Itération après itération, on améliore donc la qualité des modes recherchés en s'appuyant sur des modes auxiliaires. Bien sûr celle-ci est estimée à chaque étape et conditionne l'opportunité de simuler un autre restart ou pas. Le processus peut se résumer (très macroscopiquement !) comme suit:

Factorisation d'Arnoldi d'ordre m: $\mathbf{A}\mathbf{Q}_m = \mathbf{Q}_m\mathbf{B}_m + \mathbf{R}_m$.
Pour k= 1, NMAX_ITER_SOREN faire
Calculer $\underbrace{\tilde{\lambda}_1, \tilde{\lambda}_2 \ldots \tilde{\lambda}_p}_{\textit{Conservées pour}\ \text{amélioration}}$, $\underbrace{\tilde{\lambda}_{p+1} \ldots \tilde{\lambda}_{p+q}}_{\text{Utilisées comme shifts}}$,
QR avec shifts implicites,
Mise à jour $\mathbf{Q}_m, \mathbf{B}_m$ et $\mathbf{R}_m$,
Troncature de ces matrices *à* l'ordre p,
$\mathbf{A}\mathbf{Q}_p = \mathbf{Q}_p\mathbf{B}_p + \mathbf{R}_p \Rightarrow \mathbf{A}\mathbf{Q}_m = \mathbf{Q}_m\mathbf{B}_m + \mathbf{R}_m$,
Estimation qualité des p modes.
Fin boucle.

Algorithme 14. Méthode IRA (dite de Sorensen).

Le nombre maximum d'itérations est piloté par le mot-clé NMAX_ITER_SOREN du mot-clé facteur CALC_FREQ. Il faut remarquer que l'algorithme Arnoldi est prudemment complété par une **réorthogonalisation totale** (déclenchée que si cela s'avère nécessaire). Ce surcoût est d'autant plus acceptable qu'elle est implantée via l'algorithme IGSM de Kahan-Parlett (cf. Annexe 2) qui est particulièrement efficace. Tout ceci permet de nous assurer de la bonne tenue de la projection vis-à-vis du spectre initial.
D'autre part, l'évaluation de la **qualité des modes** ne s'effectue pas simplement en construisant les p **résidus** $\|\mathbf{r}_i\|_2 = \|(\mathbf{A}_\sigma - \tilde{\lambda}\mathbf{I})\tilde{\mathbf{u}}_i\|_2$ *via* la propriété 10. On a déjà mentionné que dans le cas non hermitien, ils ne sauraient suffire à cette tâche, notamment en cas de fort défaut de normalité. Pour s'en acquitter, sans avoir recours à d'autres informations[23] *a priori*, on utilise la propriété précédente complétée par un critère dû à Z.Bai et al[BDK93]

$$|b_{m+1,m}||\mathbf{e}_m^T\mathbf{x}| < \max\left(\varepsilon\|\mathbf{B}_m\|, PREC\_SOREN\,|\tilde{\lambda}|\right)$$

où ε est la précision machine et PREC_SOREN est un mot-clé initialisé sous CALC_FREQ. L'utilisateur a donc un contrôle (partiel) de la qualité des modes, ce dont il ne disposait pas avec les autres méthodes implantées dans le code. Compte tenu des différentes normes utilisées, cette erreur est différente de celle résultant du post-traitement global (cf. §3.6) qui est affichée dans les colonnes résultats.

Remarques:
•La technique d'accélération polynomiale utilisée est plus efficace que celle de Tchebychev, puisque cette dernière est explicite et requiert m produits matrice-vecteur.
•Pour éviter la détérioration des vecteurs de Ritz (et donc des vecteurs propres approchés) par des valeurs propres de très grands modules (associées au noyau de $\mathbf{B}$ en «shift and invert») un filtrage de type Ericsson & Ruhe[ER80] a été implanté. Des techniques plus robustes existent mais elles nécessitent des informations a priori concernant notamment les blocs de Jordan associés au noyau de l'opérateur (cf. Meerbergen & Spence, 1996).

23 Pour obtenir un critère exacte il faudrait pouvoir estimer les conditionnements spectraux des espaces invariants et les angles qu'ils font entre eux. Ce qui est difficile à obtenir, même *a posteriori* !

•Cet algorithme peut être vu comme une forme tronquée de l'algorithme QR implicite de J.C.F.Francis (cf. Annexe 1).

Le paragraphe suivant va expliciter les choix qui ont conduit à la variante mise en place dans le code.

7.5 Implantation dans *Code_Aster*

7.5.1 ARPACK

Le package ARPACK(**ARPACK** pour ARnoldi PACKage) implémente la méthode IRA. C'est un produit public téléchargeable sur internet[Arp]. Ces concepteurs, D.Sorensen, R.Lehoucq et C.Yang de la Rice University de Houston, l'ont voulu à la fois:

- **Simple d'accès:** interfaçage Fortran/C/C++ *via* le modèle de «reverse communication».
- **Modulable:** basé sur les bibliothèques LINPACK, LAPACK et BLAS.
- **Riche en fonctionnalités numériques**: décomposition de Schur, shifts modulables, nombreuses transformations spectrales, critères d'arrêt paramétrables.
- **Couvrant large périmètre d'utilisation**: simple/double précision, réel/complexe, GEP à matrices symétriques/non symétriques...
- **Performant:** pour gagner en temps CPU et en occupation mémoire, le package se décline en PARPACK version parallélisée *via* MPI et BLACS. D'autre part, même avec ARPACK séquentiel, les résolutions de systèmes linéaires requis par la méthode peuvent être effectuées par un solveur linéaire parallèle. La gestion mémoire des gros objets requis par ARPACK est laissée au soin du code hôte.

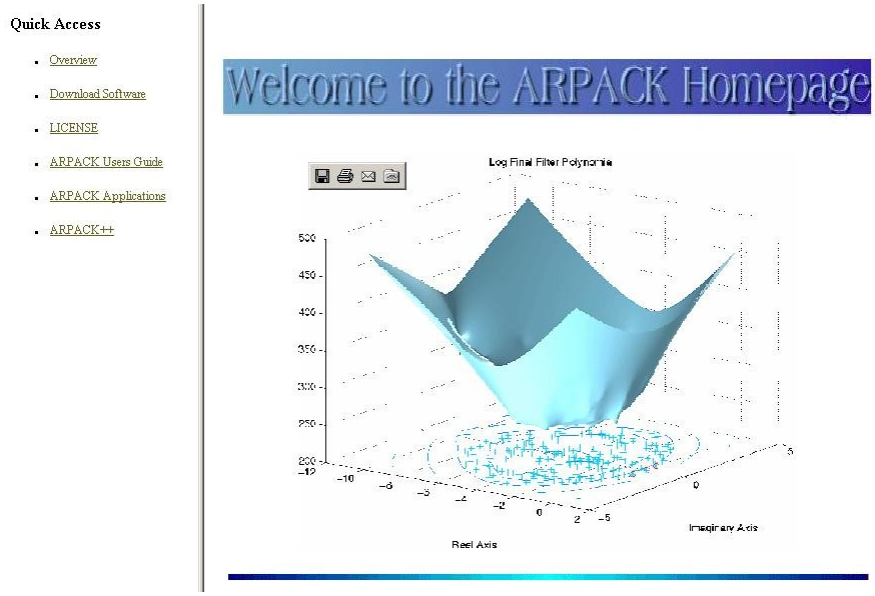


Figure 7.5-1. La page d'accueil du site web d'ARPACK[Arp].

Son efficacité est décuplée par l'utilisation de **BLAS** de niveau 2 et 3 **très optimisées** et par la mise en place de la «**reverse communication**». L'utilisateur est donc maître de ses structures de données et de ses procédures de traitement concernant l'opérateur et le produit scalaire de travail. C'est lui qui fournit aux routines ARPACK ce type d'information. Cela permet donc de gérer au mieux, avec les outils et les procédures *Code_Aster*, les produits matrice-vecteur et les résolutions de système linéaire.
Pour l'instant, la gestion mémoire des gros objets requis par ARPACK est confiée à JEVEUX. Elle bénéficie donc de ses facultés «out-of-core[24]». D'autre part, les opérateurs modaux d'*Aster* ayant accès aux mêmes solveur linéaires que ceux de statique, ils peuvent bénéficier eux aussi des avancées en temps et en mémoire que procure le parallélisme (SOLVEUR/METHODE='MUMPS' ou 'MULT-FRONT' cf. [U4.50.01][U2.08.06]).

24 «Out-Of-Core» (OOC) signifie que le progiciel de gestion mémoire peut décharger sur disque une partie des objets contenus dans la mémoire RAM. Contrairement à la traditionnelle gestion mémoire «In-Core» (IC) qui conserve tout en RAM, ce qui limite plus rapidement la taille des problèmes traitables.

Mais, attention, ces gains sont limités à la partie solveur linéaire du calcul modal. Les speed-ups atteignables sont donc moins prometteurs puisque cette partie peut représenter, dans certains cas, que 50% du temps calcul[25].

Le site d'ARPACK propose de la documentation (théorique et utilisation), des exemples d'utilisation et une page de téléchargement. La version finalisée d'ARPACK (v2.5 du 27/08/96) utilisée dans *Code_Aster* semble être la dernière. Malgré des milliers d'utilisation dans le monde (académique et industriel), un référencement avéré[HRTV07] et une reconnaissance certaine (cf. l'onglet «ARPACK applications» du site et les liens avec d'autres librairies telles PETSc et TRILINOS) ce projet de développement logiciel a été arrêté en 1997. Il a perduré quelques années de plus *via* la version C++ (ARPACK++) de D.Sorensen et F.Gomez.

Figure 7.5-2. Logo de la version C++ d'ARPACK.

7.5.2 Adaptations de l'algorithme de Sorensen

Pour traiter des problèmes modaux généralisés, ce package propose toute une série de transformations spectrales, en réel ou en complexe, en hermitien ou non. En hermitien, l'algorithme de Sorensen est basé sur le couple Lanczos-**QL** (IRLM pour Implicit Restarted Lanczos Method). Les deux approches (hermitienne ou non) ne sont d'ailleurs pas prévues pour traiter des pseudo-produits scalaires liés à des matrices indéfinies.
Justement, pour à la fois **circonscrire des problèmes numériques liés à leurs propriétés assez hétérogènes** dans le code et, d'autre part, s'assurer d'une **meilleure robustesse globale**, nous avons choisi de **travailler en non symétrique** (**IRAM** avec **Arnoldi** et **QR)**, sur **le couple opérateur de travail-produit scalaire** suivant:

$$\underbrace{\left(\mathbf{A}-\sigma\,\mathbf{B}\right)^{-1}}_{\mathbf{A}_\sigma}\mathbf{B}\,\mathbf{u}=\underbrace{\frac{1}{\mu-\sigma}}_{\lambda}\mathbf{u}$$

$$\left(\mathbf{x},\mathbf{y}\right)=\mathbf{y}^{\mathrm{T}}\mathbf{x}$$

On aurait pu traiter les problèmes de flambement en «buckling mode» *via* le même «shift and invert» et le pseudo-produit scalaire introduit par $\mathbf{A}$. Mais du fait de l'introduction quasi-systématique de Lagranges, cette matrice devient indéfinie voire singulière, ce qui perturbe grandement le processus. Les mêmes causes produisent les mêmes effets lorsque, pour un calcul de dynamique, on a recourt au $\mathbf{B}$-produit scalaire. Plutôt que de modifier tout le package en introduisant un pseudo-produit scalaire, nous avons donc opté pour un simple produit scalaire «euclidien» plus robuste et beaucoup moins coûteux.

Il a donc fallu modifier les procédures de «*reverse-communication*» du package, car il ne prévoyait pas cette option (avec des matrices standards on préfère classiquement enrichir les composantes avec un produit scalaire matriciel, même en non symétrique). Contrairement à la variante de Newmann & Pipano introduite pour Lanczos, nous nous sommes délibérément placés dans une configuration non symétrique. Mais afin d'éviter autant que faire ce peut les problèmes d'orthogonalité récurrents, même en symétrique, nous aurions opté pour la version d'IRAM utilisant Arnoldi.

L'inconvénient de cette démarche est qu'il faut, en post-traitement préliminaire d'IRAM, $\mathbf{B}$-orthonormaliser les vecteurs propres approchés pour retrouver numériquement la propriété 2 exploitées par les recombinaisons modales. Cette étape ne perturbe pas la base de modes propres exhumée et elle est très efficacement réalisée *via* l'IGSM de Kahan-Parlett.

25 Dans ce cas extrême de seulement 50% du temps elapsed passé dans la partie solveur linéaire, le parallélisme, même sur des dizaines de processeurs, n'amènera au mieux qu'un gain (ou accélération ou 'speed-up' en anglais) d'un facteur 2 (en temps) ! En mémoire RAM, les gains sont eux aussi bornés par le ratio entre les besoins propres d'ARPACK et ceux du solveur linéaire.

La **prise en compte des conditions limites** et, en particulier des doubles dualisations, a été conduite comme pour Lanczos suivant le mode opératoire décrit en §3.2. En particulier, une fois que le vecteur initial se trouve dans l'espace admissible on lui applique l'opérateur de travail. Ce procédé classique permet de purger les vecteurs de Lanczos (et donc les vecteurs de Ritz) des composantes du noyau.

D'autre part, dans certaines configurations pour lesquelles le **nombre de fréquences demandées** p est **égal au nombre de degrés de liberté actifs**, on a dû bluffer l'algorithme qui s'arrêtait en erreur fatale ! En effet, il détectait généralement bien l'espace invariant attendu (de taille p), mais du fait de la structure particulière des vecteurs propres associés aux Lagranges (cf. preuve de la propriété 4, §3.5) il avait beaucoup de mal à générer un vecteur initial qui leur soit proportionnel.

Il aurait fallu des traitements particuliers tenant compte de la numérotation de ces Lagranges, qui auraient été d'autant plus coûteux qu'ils ne sont pas foncièrement nécessaires pour résoudre le problème demandé ! Toute l'information spectrale étant déjà présente au plus profond de l'algorithme, ce n'est donc pas la peine d'achever les deux itérations restantes (lorsque l'utilisateur demande $p=n_{ddl-actifs}$, on impose automatiquement $m=p+2$). Il suffit de court-circuiter le fil naturel de l'algorithme, de retirer les modes de Ritz intéressants et de les post-traiter pour revenir dans l'espace initial.

Remarque:
•Ce type de cas de figure dans lequel on recherche un nombre de modes propres très proche du nombre de degrés de liberté sort du périmètre d'utilisation «idéal» de ce type d'algorithme. Un bon QR serait sans nul doute plus efficace, mais c'est un bon moyen de tester l'algorithme.

7.5.3 Paramétrage

Pour pouvoir activer cette méthode, il faut initialiser le mot-clé METHODE à 'SORENSEN'. La taille du sous-espace de projection est déterminée, soit par l'utilisateur, soit empiriquement à partir de la formule:

$$m=min\left(max\left(2\,p\,,\,p+2\right),n_{ddl_{actifs}}\right)$$

où p est le nombre de valeurs propres à calculer,
$n_{ddl-actifs}$ est le nombre de degrés de liberté actifs du système (cf. [§3.2])

L'utilisateur peut toujours imposer lui-même la dimension en l'indiquant avec le mot-clé DIM_SOUS_ESPACE du mot-clé facteur CALC_FREQ. On peut préférer à une valeur par défaut, un coefficient multiplicatif par rapport au nombre de fréquences contenues dans l'intervalle d'étude. Ce type de fonctionnalité concerne le mot-clé COEF_DIM_ESPACE.
Le paramètre de l'IGSM de Kahan-Parlett (cf. Annexe 2) PARA_ORTHO_SOREN, le nombre maximal d'itérations du processus global, NMAX_ITER_SOREN, et le critère de contrôle de qualité des modes, PREC_SOREN, sont accessibles par l'utilisateur sous le mot-clé facteur CALC_FREQ. Lorsque ce dernier mot-clé est nul, l'algorithme l'initialise à la précision machine.

7.6 Périmètre d'utilisation

GEP à matrices réelles quelconques (symétriques ou non) ou à matrice $\mathbf{A}$ complexe et $\mathbf{B}$ réelle symétriques.
L'utilisateur peut spécifier la classe d'appartenance de son calcul (dynamique ou flambement si matrices réelles symétriques) en initialisant le mot-clé TYPE_RESU. Suivant cette valeur, on renseigne le vecteur FREQ ou CHAR_CRIT.

7.7 Affichage dans le fichier message

L'exemple ci-dessous issu de la liste de cas tests du code (ssll103b) récapitule l'ensemble des traces gérées par l'algorithme. On retrouve notamment, pour chaque charge critique (ou fréquence), l'estimation de sa qualité *via* la norme d'erreur.

Ici, IRAM n'a itéré qu'une seule fois et a utilisé 30 IGSM (dans sa première phase). La résolution globale a consommé 91 produits (en fait, moins que cela du fait de l'introduction «implicite» du produit scalaire euclidien) matrice-vecteur et 31 inversions de système (juste la remontée car l'opérateur de travail est déjà factorisé).

```
-------------------------------------------------------------
LE NOMBRE DE DDL
TOTAL EST:                      68
DE LAGRANGE EST:                14
LE NOMBRE DE DDL ACTIFS EST:       47
-------------------------------------------------------------
L'OPTION CHOISIE EST: PLUS_PETITE
LA VALEUR DE DECALAGE CHARGE CRITIQUE EST :  0.00000E+00
-------------------------------------------------------------
INFORMATIONS SUR LE CALCUL DEMANDE :
 NOMBRE DE MODES DEMANDES     :  10
LA DIMENSION DE L'ESPACE REDUIT EST :  30
=============================================
     =       METHODE DE SORENSEN (CODE ARPACK)   =
     =       VERSION :  2.4                      =
     =          DATE :  07/31/96                 =
     ===========================================
     NOMBRE DE REDEMARRAGES                     =     1
     NOMBRE DE PRODUITS OP*X                    =    31
     NOMBRE DE PRODUITS B*X                     =    91
     NOMBRE DE REORTHOGONALISATIONS  (ETAPE 1)  =    30
     NOMBRE DE REORTHOGONALISATIONS  (ETAPE 2)  =     0
     NOMBRE DE REDEMARRAGES DU A UN V0 NUL      =     0
-------------------------------------------------------------
LES CHARGES CRITIQUES CALCULEES INF. ET SUP. SONT:
      CHARGE_CRITIQUE_INF : -9.96796E+06
      CHARGE_CRITIQUE_SUP : -6.80007E+05
-------------------------------------------------------------
       CALCUL MODAL:  METHODE D'ITERATION SIMULTANEE
                      METHODE DE SORENSEN
NUMERO    CHARGE CRITIQUE    NORME D'ERREUR
         1     -6.80007E+05       5.88791E-12
         2     -7.04572E+05       1.53647E-12
         3     -7.09004E+05       1.16735E-12
        ...
        10     -9.96796E+06       3.55014E-12
-------------------------------------------------------------
VERIFICATION A POSTERIORI DES MODES
DANS L'INTERVALLE (-1.00178E+07,-6.76607E+05)
   IL Y A BIEN   10 CHARGE(S) CRITIQUE(S)
-------------------------------------------------------------
```

Exemple 6. Trace de `MODE_ITER_SIMULT` dans le fichier message avec `METHODE='SORENSEN'`.

Remarque:
•L'introduction de cette méthode a permis de solder de nombreuses fiches d'anomalies liées à des multiplicités, des clusters ou des recherches de valeurs propres d'ordres de grandeur très différents sur lesquels Lanczos et Bathe & Wilson achoppaient.

7.8 Récapitulatif du paramétrage

Récapitulons maintenant le paramétrage disponible de l'opérateur MODE_ITER_SIMULT avec ce solveur modal.

Mot-clé facteur	Mot-clé	Valeur par défaut	Références
	TYPE_RESU ='DYNAMIQUE'	'DYNAMIQUE'	§3.1
	'MODE_FLAMB'		§3.1
	METHODE='SORENSEN'	'SORENSEN'	§7.4
CALC_FREQ	**FREQ (si CENTRE)**		§5.4
	CHAR_CRIT (si CENTRE)		§5.4
	OPTION ='PLUS_PETITE'	'PLUS_PETITE''	§5.4
	'BANDE'		§5.4

Mot-clé facteur	Mot-clé	Valeur par défaut	Références
	'CENTRE'		§5.4
	NMAX_FREQ	10	§5.4
	AMOR_REDUIT	0.	§5.4
	DIM_SOUS_ESPACE **COEF_DIM_ESPACE**	calculé	§7.5
	PREC_SOREN	0.	§7.4, §7.5
	NMAX_ITER_SOREN	20	§7.4, §7.5
	PARA_ORTHO_SOREN	0.717	§7.5, Annexe 2
	NMAX_ITER_SHIFT	3	[Boi12] §3.2
	PREC_SHIFT	0.05	[Boi12] §3.2
	SEUIL_FREQ	1.E-02	§3.7
VERI_MODE	STOP_ERREUR='OUI'	'OUI'	§3.7
	'NON'		§3.7
	PREC_SHIFT	5.E-03	§3.7
	SEUIL	1.E-06	§3.7
	STURM='OUI'	'OUI'	§3.7
	='NON'		§3.7

Tableau 7.8-1. Récapitulatif du paramètrage de MODE_ITER_SIMULT (GEP) avec METHODE='SORENSEN'.

Remarques:
•On retrouve toute la "tripaille" de paramètres liée aux pré-traitements du test de Sturm (NMAX_ITER_SHIFT, PREC_SHIFT) et aux post-traitements de vérification (SEUIL_FREQ, VERI_MODE).
•Lors des premiers passages, il est fortement conseillé de ne modifier que les paramètres principaux notés en gras. Les autres concernent plus les arcanes de l'algorithme et ils ont été initialisés empiriquement à des valeurs standards.
•En particulier, pour améliorer la qualité d'un mode, le paramètre fondamental est la dimension du sous-espace (DIM_SOUS_ESPACE/COEF_DIM_ESPACE).

8 Méthode de Bathe et Wilson (METHODE='JACOBI')

8.1 Principe

La méthode de Bathe et Wilson est une **méthode d'itérations simultanées** qui consiste à **étendre l'algorithme des itérations inverses**. On travaille à partir du problème décalé $\mathbf{A}^{\sigma}\mathbf{x}:=(\lambda-\sigma)\mathbf{B}\mathbf{x}$ avec $\mathbf{A}^{\sigma}:=\mathbf{A}-\sigma\mathbf{B}$ (cf. §5.4). On distingue quatre parties dans l'algorithme[Bat71]:

•Choisir p vecteurs initiaux indépendants $\mathbf{x}_{1,}\mathbf{x}_{2,}...,,\mathbf{x}_{p}$ et construire la matrice $\mathbf{X}$ qu'ils engendrent.

•Calculer les éléments propres, via la méthode globale de Jacobi (cf. Annexe 3), dans le sous-espace de Ritz en résolvant

$$(\bar{\mathbf{A}}-\bar{\lambda}_i\bar{\mathbf{B}})\mathbf{u}_i=0 \quad \text{.où} \quad \bar{\mathbf{A}}=\mathbf{Q}^{\mathrm{T}}\mathbf{A}^{\sigma}\mathbf{Q} \quad \text{et} \quad \bar{\mathbf{B}}=\mathbf{Q}^{\mathrm{T}}\mathbf{B}\mathbf{Q}$$

•On revient ensuite à l'espace initial (pour les vecteurs propres) via la transformation $\mathbf{X}=\mathbf{Q}\mathbf{U}$ .où $\mathbf{U}=[\{\mathbf{u}_i\}]$

•Tester la convergence des modes propres λ_i .

8.2 Tests de convergence

La méthode de Bathe et Wilson converge vers les p plus petites valeurs propres à condition que les p vecteurs initiaux ne soient pas $\mathbf{B}$ -orthogonaux à l'un des vecteurs propres. Par ailleurs, les matrices $\bar{\mathbf{A}}$ et $\bar{\mathbf{B}}$ tendent vers des matrices diagonales. Pour cette raison et comme les matrices sont pleines, on utilise la méthode de Jacobi (cf. Annexe 3) pour trouver les éléments propres du sous-espace de Ritz.

Pour tester la convergence des valeurs propres, on les classe après chaque itération par ordre croissant en valeur absolue et on regarde si, pour chaque valeur propre, le test suivant est vérifié

$$|\lambda^{k+1}-\lambda^{k}|\leq PREC\_BATHE|\lambda^{k+1}|$$

où l'exposant k indique le nombre d'itérations. Si après NMAX_ITER_BATHE itérations, on n'a pas convergé pour toutes les valeurs propres, un message d'alarme est émis dans le fichier message.

8.3 Implantation dans *Code_Aster*

8.3.1 Dimension du sous-espace

Si on désire calculer p valeurs propres, il est recommandé d'utiliser un sous espace de dimension q supérieure. On ne vérifiera la convergence que pour les r plus petites valeurs propres où $p\leq r\leq q$. Il semble que $r=p$ ne soit pas suffisant: on peut trouver les bonnes valeurs propres mais les vecteurs propres ne sont pas corrects (la convergence est plus lente pour les vecteurs propres que pour les valeurs propres). $r=(p+q)/2$ semble un bon choix. Pour q on prend habituellement[Bat71]

$$q=min(p+8,2p)$$

8.3.2 Choix des vecteurs initiaux

Pour choisir les q vecteurs initiaux, on opère de la façon suivante:

•Premier vecteur tel que $x_{1i}=\dfrac{\mathbf{A}^{\sigma}_{ii}}{\mathbf{B}_{ii}}$,

• Pour les autres vecteurs de 2 à $(q-1)$

$$x_2=\begin{Bmatrix}0\\1\\0\end{Bmatrix}--\text{ligne}\quad i_1 \qquad x_{q-1}=\begin{Bmatrix}0\\1\\0\\0\end{Bmatrix}--\text{ligne}\quad i_2,\ldots$$

où les i sont les indices correspondant aux plus petites valeurs successives de $\frac{\mathbf{A}_{ii}^{\sigma}}{\mathbf{B}_{ii}}$,

•Dernier vecteur $\mathbf{x}_q$, vecteur aléatoire.

8.4 Périmètre d'utilisation

GEP à matrices réelles symétriques.
L'utilisateur peut spécifier la classe d'appartenance de son calcul en initialisant le mot-clé TYPE_RESU. Suivant cette valeur, on renseigne le vecteur FREQ ou CHAR_CRIT.

8.5 Récapitulatif du paramétrage

Récapitulons maintenant le paramétrage disponible de l'opérateur MODE_ITER_SIMULT avec solveur modal. Pour pouvoir utiliser la méthode de Bathe et Wilson, il faut choisir la commande MODE_ITER_SIMULT et sélectionner METHODE='JACOBI'. Les deux paramètres concernant directement la convergence de la méthode sont accessibles sous le mot-clé facteur CALC_FREQ à l'aide des mots-clés PREC_BATHE et NMAX_ITER_BATHE. On y trouve aussi ceux gérant la méthode interne de résolution modale, PREC_JACOBI et NMAX_ITER_JACOBI .

Mot-clé facteur	Mot-clé	Valeur par défaut	Références
	TYPE_RESU='DYNAMIQUE'	'DYNAMIQUE'	§3.1
	'MODE_FLAMB'		
	METHODE 'JACOBI'	'SORENSEN'	Annexe 3
CALC_FREQ	**FREQ (si CENTRE)**		§5.4
	CHAR_CRIT (si CENTRE)		§5.4
	OPTION ='PLUS_PETITE'	'PLUS_PETITE'	§5.4
	'BANDE'		
	'CENTRE'		
	NMAX_FREQ	10	§5.4
	DIM_SOUS_ESPACE COEF_DIM_ESPACE	Calculé	§8.3
	PREC_BATHE	1.E-10	§8.2
	NMAX_ITER_BATHE	40	§8.2
	PREC_JACOBI	1.E-12	Annexe 3
	NMAX_ITER_JACOBI	12	Annexe 3
	NMAX_ITER_SHIFT	3	[Boi12] §3.2
	PREC_SHIFT	0.05	[Boi12] §3.2
	SEUIL_FREQ	1.E-02	§3.7
VERI_MODE	STOP_ERREUR='OUI'	'OUI'	§3.7
	='NON'		
	PREC_SHIFT	5.E-03	§3.7
	SEUIL	1.E-06	§3.7
	STURM='OUI'	'OUI'	§3.7
	'NON'		§3.7

Tableau 8.3-1. Récapitulatif du paramètrage de MODE_ITER_SIMULT (GEP) avec METHODE='JACOBI'.

Remarques:
•On retrouve toute la "tripaille" de paramètres liée aux pré-traitements du test de Sturm (NMAX_ITER_SHIFT, PREC_SHIFT) et aux post-traitements de vérification (SEUIL_FREQ, VERI_MODE).
•Lors des premiers passages, il est fortement conseillé de ne modifier que les paramètres principaux notés en gras. Les autres concernent plus les arcanes de l'algorithme et ils ont été initialisés empiriquement à des valeurs standards.

9 Méthode globale QZ (METHODE='QZ')

9.1 Introduction

La plupart des méthodes modales ont recours à une transformation spectrale de type «shift and invert» pour faciliter la recherche d'une partie du spectre et transformer le GEP initial en un SEP classique(cf. §3.7). Cette démarche est habile mais elle implique des factorisations numériques de matrices dynamiques du type $\mathbf{A}_\sigma = \left(\mathbf{A}-\sigma\,\mathbf{B}\right)^{-1}\mathbf{B}$ (cf. §3.8). Le résultat de cette factorisation n'est qu'un intermédiaire de calcul maint fois utilisé dans l'algorithmique des solveurs modaux. Or **ces factorisations sont forcément entachées d'erreurs numériques** liées aux conditionnements des matrices et à l'erreur inverse[26] du solveur direct(cf. [R6.02.03] [Boi08] §2.3). D'où l'idée, pour construire une méthode modale robuste, d'éviter ce type de mécanisme et de ne recourir qu'à:

- Des **transformations qui produisent/propagent peu d'erreurs d'arrondis** (dites «backward stables») du type Householder ou Givens,
- Des **matrices canoniques faciles à manipuler** (triangulaires, diagonales ou de Hessenberg) et/ou **très bien conditionnées** (orthogonales en réel ou unitaires en complexe).

9.2 Les GEP et la méthode QZ

C'est le principe de la méthode QZ qui traite directement le GEP

$$\begin{array}{c}\text{Trouver } \left(\lambda\,,\mathbf{u}\right) \text{ tel que}\\ \left(\mathbf{A}-\lambda\,\mathbf{B}\right)\,\mathbf{u}=\mathbf{0}\end{array} \qquad (9.2\text{-}1)$$

plutôt qu'un SEP issu de sa modification *via* une transformation spectrale. Cette méthode cherche en fait à construire la décomposition de Schur généralisée suivante (pendant de la décomposition de Schur standard cf. §2.1.5).

Théorème 12 (Décomposition Généralisée de Schur)
Soient $\mathbf{A}$ et $\mathbf{B}$ deux matrices carrées complexes, alors il existe deux matrices carrées unitaires $\mathbf{Q}$ et $\mathbf{Z}$ telles que $\mathbf{Q}^*\mathbf{A}\mathbf{Z}=\mathbf{T}$ et $\mathbf{Q}^*\mathbf{B}\mathbf{Z}=\mathbf{S}$ soient des matrices triangulaires supérieures. Les valeurs propres du GEP (9.1-1) se calculent alors facilement grâce aux termes diagonaux (complexes) des matrices $\mathbf{T}$ et $\mathbf{S}$:

$$\lambda\left(\mathbf{A}\,,\mathbf{B}\right):=\left\{\lambda_i:=\mathbf{T}_{ii}/\mathbf{S}_{ii} \text{ avec } \mathbf{S}_{ii}\neq 0\,;\,\lambda_i=\infty \text{ si } \mathbf{S}_{ii}=0\right\} \qquad (9.2\text{-}2)$$

Preuve:
Cf. [GL89] pp396 et [MS73].

Ce théorème indique donc que, une fois la Décomposition Généralisée de Schur exhumée, **résoudre le GEP (9.2-1) revient à résoudre le GEP équivalent** (et beaucoup plus simple car triangulaire)

$$\begin{array}{c}\text{Trouver } \left(\lambda\,,\mathbf{v}\right) \text{ tel que}\\ \left(\mathbf{T}-\lambda\,\mathbf{S}\right)\,\mathbf{v}=\mathbf{0}\end{array} \qquad (9.2\text{-}3)$$

Ils ont le même spectre, $\lambda\left(\mathbf{A}\,,\mathbf{B}\right)=\lambda\left(\mathbf{T}\,,\mathbf{S}\right)$, et on passe des vecteurs propres de l'un à ceux de l'autre, *via* la transformation canonique

$$\mathbf{u}=\mathbf{Z}\mathbf{v} \qquad (9.2\text{-}4)$$

D'autre part, la **manipulation de couples du type** $\left(\alpha_i\,,\beta_i\right)=\left(\mathbf{T}_{ii}\,,\mathbf{S}_{ii}\right)$, plutôt que de directement une valeur propre $\lambda_i:=\dfrac{\alpha_i}{\beta_i}$, est une astuce numérique permettant de mieux appréhender la complexité qu'induit la résolution directe d'un GEP. En effet, suivant par exemple le rang de la matrice $\mathbf{B}$, le spectre peut être:

- fini $\lambda\left(\mathbf{A}\,,\mathbf{B}\right)=\left\{\lambda_1\ldots\lambda_n\right\}$, avec éventuellement des valeurs propres de valeurs très grandes $\pm\infty$,

26 L'erreur inverse mesure la propention de l'algorithme à transmettre/amplifier les erreurs d'arrondi.

• vide ($\lambda(\mathbf{A},\mathbf{B})=\emptyset$),
•infini ($\lambda(\mathbf{A},\mathbf{B})=\mathbb{R}$ ou $\mathbb{C}$).

Dans les deux premiers cas, le système est dit «régulier» par opposition à la dernière situation où il est dit «singulier». Pour s'en convaincre, les exemples ci-dessous illustrent les trois cas de figures :

$$\begin{aligned}
&\mathbf{A}:=\begin{bmatrix}1 & 2\\ 0 & 3\end{bmatrix},\ \mathbf{B}:=\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix} \Rightarrow \lambda(\mathbf{A},\mathbf{B})=\{1\}\\
&\mathbf{A}:=\begin{bmatrix}1 & 2\\ 0 & 3\end{bmatrix},\ \mathbf{B}:=\begin{bmatrix}0 & 0\\ 1 & 0\end{bmatrix} \Rightarrow \lambda(\mathbf{A},\mathbf{B})=\emptyset\\
&\mathbf{A}:=\begin{bmatrix}1 & 2\\ 0 & 0\end{bmatrix},\ \mathbf{B}:=\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix} \Rightarrow \lambda(\mathbf{A},\mathbf{B})=\mathbb{C}
\end{aligned} \qquad (9.2\text{-}5)$$

La représentation des solutions en (α,β) permet alors de gérer plus proprement ces cas de figure. Par exemple, la valeur $\beta\approx 0$ peut être interprétée comme une valeur propre infinie $\lambda\approx\infty$ (probablement à ne pas retenir dans le spectre de travail !). Si de plus $\alpha\approx 0$, on a une indétermination pour calculer la valeur propre correspondante et cela illustre un système singulier ou très mal conditionné.

Dans le cas des GEPs résolus par *Code_Aster* ce cas de figure apparaît lorsqu'on **traite les valeurs propres associées aux ddls bloqués et à ceux de Lagranges** (cf. § 3.2). La décomposition (α,β) fournie alors un cadre pour filtrer ces artefacts numériques du spectre physique véritablement recherchés:

$$\lambda_{physique}(\mathbf{A},\mathbf{B}):=\left\{-\infty<<|\lambda_i|<<+\infty,\ i=1\ldots n_{ddl_{actifs}}\right\} \qquad (9.2\text{-}6)$$

Remarques:
•La méthode QZ est «backward stable» pour résoudre un GEP standard. Mais elle perd cette propriété lorsqu'on l'utilise sur un GEP linéarisé (cf. §2.1), intermédiaire de calcul pour appréhender un QEP (cf. [TM01] §5.1).
•Dans le cas de matrices réelles (cas le plus courant dans Code_Aster), on a une décomposition similaire dite «Généralisée Réelle» avec des matrices carrées orthogonales $\mathbf{Q}$ et $\mathbf{Z}$ telles que $\mathbf{Q}^{\mathbf{T}}\mathbf{A}\mathbf{Z}=\mathbf{T}$ soit quasi-triangulaire supérieure (.i.e. diagonale par blocs 1×1 ou 2×2) et $\mathbf{Q}^{\mathbf{T}}\mathbf{B}\mathbf{Z}=\mathbf{S}$ soit triangulaires supérieures. La philosophie du calcul reste la même, en introduisant des couples (α,β) comme intermédiaires, avant le calcul proprement-dit des valeurs propres λ . Une différence notable avec le cas complexe provient du fait que l'on peut, même en présence de modes complexes, continuer à travailler en arithmétique réelle. Cette stratégie exploitée par certaines routines de LAPACK a été utilisée avec la méthode QZ de `MODE_ITER_SIMULT`.
•Dans certains cas (matrices SPD, structures bandes…), l'approche peut se décliner plus efficacement. Par exemple, si le GEP est à matrices réelles symétriques avec $\mathbf{B}$ en plus définie positive (donc il existe une factorisation de Cholesky telle que $\mathbf{B}=\mathbf{L}\mathbf{D}\mathbf{L}^{\mathbf{T}}$), le GEP (9.2-1) se résout directement via le SEP symétrique réel équivalent

• $$(\mathbf{GEP})\begin{cases}\text{Trouver } (\lambda,\mathbf{u}) \text{ tel que}\\ (\mathbf{A}-\lambda\mathbf{B})\,\mathbf{u}=0\end{cases} \Leftrightarrow (\mathbf{SEP})\begin{cases}\text{Trouver } (\lambda,\mathbf{u}) \text{ tel que}\\ \left(\mathbf{L}^{-1}\mathbf{A}\mathbf{L}^{-T}\right)\left(\mathbf{L}^{\mathrm{T}}\mathbf{u}\right)=\lambda\left(\mathbf{D}\mathbf{L}^{T}\mathbf{u}\right)\end{cases} \qquad (9.2\text{-}7).$$

•Ce SEP canonique se résout alors de manière robuste et plus efficace via un QR adapté (cf. annexe 1). Cette stratégie exploitée par certaines routines de LAPACK a été utilisée avec la méthode QZ de `MODE_ITER_SIMULT` (`METHODE='QZ'+TYPE_QZ='QZ_QR'`).

9.3 La méthode QZ

Dans le cas général, **l'algorithme de la méthode QZ peut se décomposer en trois étapes**:
•*Via* des transformations de Givens (cf. annexe 3) ou de Housholder 2x2, les deux matrices complexes $\mathbf{A}$ et $\mathbf{B}$ (resp. réelles) sont décomposées sous une forme dite «Généralisée Hessenberg Supérieure». C'est-à-dire qu'on détermine deux matrices carrées unitaires (resp. orthogonales) $\mathbf{U}_1$ et $\mathbf{V}_1$ telles

que $\mathbf{A}=\mathbf{U}_1\mathbf{H}\,\mathbf{V}_1^*$ et $\mathbf{B}=\mathbf{U}_1\mathbf{R}\,\mathbf{V}_1^*$ (resp. $\mathbf{A}=\mathbf{U}_1\mathbf{H}\,\mathbf{V}_1^T$ et $\mathbf{B}=\mathbf{U}_1\mathbf{R}\,\mathbf{V}_1^T$) avec $\mathbf{H}$ de Hessenberg supérieure et $\mathbf{R}$ triangulaire supérieure.

Par exemple, avec $\mathbf{A}:=\begin{bmatrix}10 & 1 & 2\\ 1 & 2 & -1\\ 1 & 1 & 2\end{bmatrix}$ et $\mathbf{B}:=\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\end{bmatrix}$, on trouve

$$\mathbf{U}_1:=\begin{bmatrix}-0.12 & -0.99 & 0.03\\ -0.49 & -0.02 & -0.86\\ -0.86 & 0.12 & 0.49\end{bmatrix},\ \mathbf{V}_1:=\begin{bmatrix}1.00 & 0.00 & 0.00\\ 0.00 & -0.89 & -0.44\\ 0.00 & 0.44 & -0.89\end{bmatrix},$$

$$\mathbf{H}:=\begin{bmatrix}-2.58 & 1.54 & 2.42\\ -9.76 & 0.08 & 1.92\\ 0.00 & 2.72 & -0.76\end{bmatrix} \text{ et } \mathbf{R}:=\begin{bmatrix}-8.12 & 3.63 & 14.20\\ 0.00 & 0.00 & 1.87\\ 0.00 & 0.00 & 0.00\end{bmatrix}.$$

•En généralisant l'algorithme à double shifts implicites de Francis (à base de transformations de Householder, cf. annexe 1), on finalise la transformation. On exhume alors deux matrices carrées unitaires (resp. orthogonales) $\mathbf{U}_2$ et $\mathbf{V}_2$ telles que $\mathbf{H}=\mathbf{U}_2\mathbf{T}\,\mathbf{V}_2^*$ et $\mathbf{R}=\mathbf{U}_2\mathbf{S}\,\mathbf{V}_2^*$ (resp. $\mathbf{H}=\mathbf{U}_2\mathbf{T}\,\mathbf{V}_2^T$ et $\mathbf{R}=\mathbf{U}_2\mathbf{S}\,\mathbf{V}_2^T$) avec $\mathbf{T}$ et $\mathbf{S}$ triangulaires supérieures (resp. $\mathbf{T}$ quasi-triangulaire supérieure). En regroupant ces deux étapes, on établit le résultat recherché, comme le produit de transformations unitaires reste unitaire,

$$\underbrace{\left(\mathbf{U}_1\mathbf{U}_2\right)}_{\mathbf{Q}^T}\mathbf{A}\underbrace{\left(\mathbf{V}_1\mathbf{V}_2\right)}_{\mathbf{Z}}=\mathbf{T} \text{ et } \underbrace{\left(\mathbf{U}_1\mathbf{U}_2\right)}_{\mathbf{Q}^T}\mathbf{B}\underbrace{\left(\mathbf{V}_1\mathbf{V}_2\right)}_{\mathbf{Z}}=\mathbf{S} \qquad (9.3\text{-}1)$$

•En utilisant les termes diagonaux des matrices triangulaires obtenues $\mathbf{T}$ et $\mathbf{S}$, on déduit (après filtrage et quelques tests de validité) les valeurs propres physiques du problème vibratoire modélisé. Les vecteurs de Schur généralisé à droite et à gauche (vecteurs colonne de $\mathbf{Q}$ et $\mathbf{Z}$) permettent de remonter aux vecteurs propres du GEP équivalent (9.2-3) puis aux vecteurs propres d'origine (*via* la transformation (9.2-4)).

Cette méthode QZ est préconisée lorsqu'on cherche à obtenir de manière robuste tout le spectre d'un GEP. Cependant, elle est très coûteuse en temps CPU $\vartheta\left(30\,n^3\right)$ contre $\vartheta\left(nm\left(m-p\right)\right)$ pour IRAM avec n la taille du problème, p le nombre de modes propres souhaités et m la taille de l'espace de projection. De plus, elle est très gourmande en capacité mémoire car elle requiert le stockage plein des matrices. D'où une complexité mémoire en $\vartheta\left(n^2\right)$ contre, par exemple, $\vartheta\left(2np+p^2\right)$ pour IRAM. Bref, **cet algorithme est à réserver aux petits GEP** s (inférieur à 10^3 degrés de liberté), plutôt denses, **dont on souhaite obtenir une estimation fiable du spectre** . Comme pour l'algorithme QR dans Lanczos/IRAM ou pour Jacobi avec Bathe & Wilson, il est plutôt à exploiter comme brique élémentaire d'un processus algorithmique qui aura (considérablement) réduit la taille du problème initial.

Pour de plus amples informations sur la méthode, on pourra consulter G.H.Golub[GL89] ou le papier fondateur de C.B.Moler et G.W.Stewart[MS73] (1973)

9.4 Implantation dans *Code_Aster*

9.4.1 LAPACK

Des routines de l'excellente **bibliothèque d'algèbre linéaire LAPACK** [Lap] (cf. § 2.3) propose des routines dédiées (et des drivers les encapsulant) à la résolution de GEP directement *via* un QZ. Et ce, pour un large panel de problèmes: SPD, réel non symétrique, complexe hermitien ou quelconque, simple/double précision, stockage plein ou par bande… D'autre part, la librairie propose toute une panoplie de routines pour décomposer la résolution, pré et post-traiter les données manipulées voire estimer les erreurs numériques qui

entachent les solutions. Elle reste cependant très liées au stockage plein des matrices, elle ne travaille qu'en In-Core et n'exploite aucun parallélisme[27].

D'autre part, il ne semble pas qu'il existe de package du domaine public implémentant plus efficacement[28] que LAPACK une méthode de type QZ. C'est donc sur **cette librairie que se base la méthode QZ disponible dans MODE_ITER_SIMULT** . Notons que beaucoup d'autres produits on fait le même choix de LAPACK et de son QZ: module `eigenvalues` de Python/linear algebra, fonctions `eig` et `qz` pour Matlab, `spec` pour Scilab, MSC/Nastran..

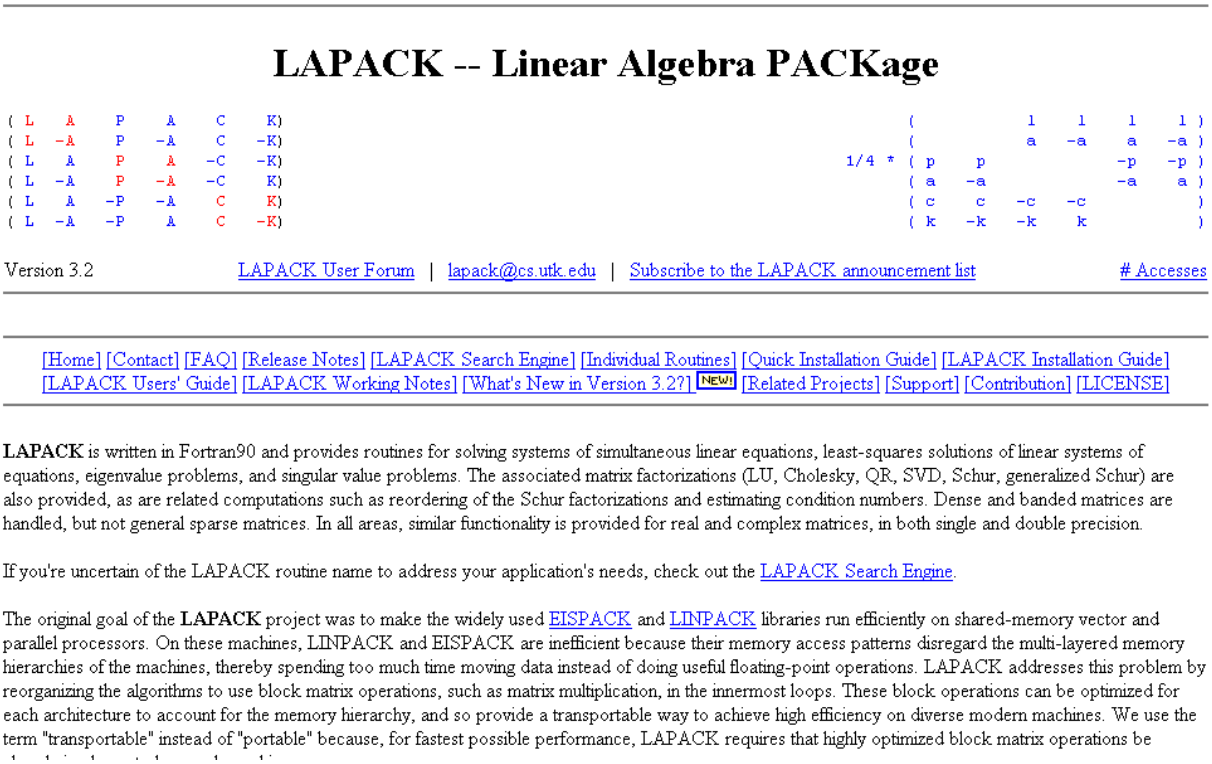


Figure 9.4-1. La page d'accueil du site web de LAPACK[Lap].

9.4.2 Intégration et post-vérifications

Dans l'implémentation qui a été retenue dans *Code_Aster* , les **GEP à modes réels** (matrices symétriques réelles) **sont traités en arithmétique réelle *via* les routines DGGEV et DGGEVX** . La première routine est à utiliser lors d'un run standard (`TYPE_QZ='QZ_SIMPLE'`, valeur par défaut). La seconde (`TYPE_QZ='QZ_EQUI'`), plus sophistiquée, est à réserver aux résolutions difficiles. En particulier, elle équilibre et permute les termes des matrices d'entrée afin d'améliorer les conditionnements spectraux et donc de diminuer l'erreur numérique sur les solutions. Parfois ce pré-traitement peut être préjudiciable en augmentant démesurément des termes proches de la précision machine, faussant ainsi les résultats. Pour cette raison, cette option n'est pas activée par défaut.

Dans le cas très particulier de **GEP symétrique réel à matrice $\mathbf{B}$, en plus, définie positive**[29] , le recours au driver LAPACK `DSYGV` est plus efficace (cf. formule § 9.2.7) que les `DGGEV/X`. Il doit être activé explicitement *via* le mot-clé `TYPE_QZ='QZ_QR'`. Pour certaines situations (flambement, double Lagrange, matrice complexe ou non symétrique, QEP linéarisé), la caractère non SPD de $\mathbf{B}$ peut être détecté *a priori*. Une erreur fatale arrête alors le calcul si la valeur '`QZ_EQUI`' a été choisie malgré tout.

27 LAPACK a une version parallèle (dont le périmètre est différent): ScaLAPACK[Sca].
28 C'est-à-dire avec un stockage creux, voire en mode parallèle.
29 Par exemple un calcul dynamique sans amortissement hystérétique avec des conditions limites de blocage modélisées uniquement par élimination (sans le recours au double Lagrange).

Dans le troisième et dernier cas de figure de **GEP à modes complexes** (matrices complexes symétriques[30] et/ ou réelles non symétriques), la résolution est directement effectuée en arithmétique complexe *via* les drivers LAPACK ZGGEV et ZGGEVX. Comme en réel, le premier est le mode standard par défaut (activé par TYPE_QZ='QZ_SIMPLE') et le deuxième, le mode expert (TYPE_QZ='QZ_EQUI').

A l'issu du calcul modal, QZ renvoie à *Code_Aster* des couples (α_i, β_i) et leur vecteur propre associé $\mathbf{u}_i$, que l'on va trier et filtrer suivant les préconisations suivantes:

•**Vérifications mécaniques:** Si $|\beta_i|$ est proche de la précision machine, le mode n'est pas retenu (il correspond sans doute à un ddls bloqué ou à un Lagrange de blocage) et un message d'alarme est émis (si INFO =2).

Une fois tous les tris effectués, on s'assure que le nombre de modes physiquement acceptables (cf. formule (9.1-6)) correspond bien au nombre de ddls_actifs $n_{ddl-actifs}$. Celui-ci est déterminé dans la phase d'initialisation et est indépendant du solveur modal (cf. § 3.2). Si le GEP est à modes réels, la non vérification de ce critère entraine une erreur fatale. Dans les autres cas, il est seulement signalé par un affichage dédié. Il n'est alors pas forcément synonyme de dysfonctionnements. Tout dépend du tri ultérieur des valeurs propres complexes conjuguées !

•**Vérifications numériques:** Si $|\alpha_i| \geq \|\mathbf{A}\|$ ou $|\beta_i| \geq \|\mathbf{B}\|$, on ne retient pas le mode et un message d'alarme est émis.

Dans le cas d'un *GEP symétrique réel* (avec TYPE_QZ='QZ_SIMPLE'/'QZ_EQUI'), on vérifie de plus que toutes les valeurs propres ont une partie imaginaire très petite (inférieure à la valeur f_{corrig} renseignée par SEUIL_FREQ , valeur par défaut égale à 10^{-2}). Dans le cas contraire, une alarme est émise.

Dans le cas d'un *GEP symétrique définie positive* réel traité *via* 'QZ_QR', on exclus les valeurs propres inférieures à $-2 f_{corrig}$ et supérieure à $0.5 * 10^{308}$ (0.5*valeur maximale représentable en machine[31]). Ces valeurs arbitraires résultent d'expériences numériques sur des cas-tests modaux de la base *Aster*. Lorsqu'on trouve une valeur propre en dehors de cette bande, une alarme est émise si INFO=2.

Puis, une fois les modes issus de QZ filtrés, ceux-ci vont être triés et réordonnés suivant les *desiderata* de l'utilisateur:

•CALC_FREQ/OPTION='BANDE': option licite seulement avec un GEP à modes reels. On sélectionne les valeurs propres situées dans la bande fréquentielle souhaitée par l'utilisateur (mot-clé FREQ/CHAR_CRIT).

•CALC_FREQ/OPTION='PLUS_PETITE': on retient la valeur propre la plus petite en module.

•CALC_FREQ/OPTION='CENTRE': on sélectionne les valeurs propres les plus proches en module du shift paramétré par le mot-clé FREQ/CHAR_CRIT. On en retient NMAX_FREQ.

•CALC_FREQ/OPTION='TOUTE': on conserve toutes les valeurs propres licites issues des filtres précédents.

Pour finir, comme pour tous les solveurs modaux du code (cf. §3.8), les modes retenus vont subir deux dernières vérifications:

•Test sur la norme du résidu du mode $(\lambda, \mathbf{u})$.

•Dans le cas standard d'un GEP à modes réels, test de Sturm de comptage de valeurs propres dans la bande fréquentielle *ad hoc*.

Remarque:

•Ce travail d'intégration d'un solveur QZ dans Code_Aster poursuit et industrialise les développements entrepris par M.Begon[Beg06]. Il a été effectué pour répondre[Boi08b] notamment aux nouveaux besoins de robustesse (sur de petits problèmes) des fonctionnalités dynamiques du code CADYRO[32].

9.5 Périmètre d'utilisation

GEP à matrices réelles quelconques (symétriques ou non) ou à matrice $\mathbf{A}$ complexe et $\mathbf{B}$ réelle symétriques.

30 Ce cas de figure se rencontre avec la modélisation *Aster* de l'amortissement hystérétique.

31 Renvoyée par la variable d'environnement *Aster* R8MIEM.

32 Logiciel effectuant des analyses vibratoires sur des lignes d'arbres de machines tournantes.

L'utilisateur peut spécifier la classe d'appartenance de son calcul (dynamique ou flambement) en initialisant le mot-clé TYPE_RESU. Suivant cette valeur, on renseigne le vecteur FREQ ou CHAR_CRIT.

9.6 Affichage dans le fichier message

Dans le fichier message sont mentionnées les informations relatives à la méthode choisie (ici QZ), à sa variante (ici 'QZ_SIMPLE') et aux modes retenus. Dans le cas le plus courant d'un GEP à modes propres réels, on précise la liste des fréquences f_i retenues (FREQUENCE) et leur norme d'erreur (NORME D'ERREUR).

```
-----------------------------------------------
      CALCUL MODAL:  METHODE GLOBALE DE TYPE QR
                     ALGORITHME QZ_SIMPLE

   NUMERO     FREQUENCE (HZ)     NORME D'ERREUR
        1       1.67638E+02        2.47457E-11
        2       1.67638E+02        1.48888E-11
        3       1.05060E+03        2.00110E-12
        4       1.05060E+03        1.55900E-12
.....
```

Exemple 7a. Trace de MODE_ITER_SIMULT dans le fichier message avec METHODE='QZ'. Extrait du cas-test forma11a.

Pour un GEP à modes complexes, contrairement aux QEP[Boi09], *Code_Aster* ne filtre pas les valeurs propres conjuguées. Il les conserve et les affiche par ordre croissant de partie réelle. Ainsi, les colonnes FREQUENCE et AMORTISSEMENT regroupent, respectivement, $\mathrm{Re}(f_i)$ et $\frac{\mathrm{Im}(f_i)}{2\,\mathrm{Re}(f_i)}$.

```
-----------------------------------------------------------------
-----
      CALCUL MODAL:  METHODE GLOBALE DE TYPE QR
                     ALGORITHME QZ_EQUI

   NUMERO     FREQUENCE (HZ)     AMORTISSEMENT     NORME D'ERREUR
        1       6.44568E+00        5.00000E-02        1.28280E-15
        2       1.55613E+01        5.00000E-02        6.26512E-16
 NORME D'ERREUR MOYENNE:  0.95466E-15
.....
```

Exemple 7b. Trace de MODE_ITER_SIMULT dans le fichier message avec METHODE='QZ'. Extrait du cas-test sdld313c.

9.7 Récapitulatif du paramétrage

Récapitulons maintenant le paramétrage disponible de l'opérateur MODE_ITER_SIMULT avec cette option METHODE='QZ'.

Mot-clé facteur	Mot-clé	Valeur par défaut	Références
	TYPE_RESU='DYNAMIQUE'	'DYNAMIQUE'	§3.1
	'MODE_FLAMB'		§3.1
	METHODE ='QZ'	'SORENSEN'	§9.2
CALC_FREQ	**FREQ** (si CENTRE)		§9.4
	CHAR_CRIT (si CENTRE)		§9.4
	OPTION='PLUS_PETITE'	'PLUS_PETITE''	§9.4
	'BANDE'		§9.4
	'CENTRE'		§9.4
	'TOUTE'		§9.4
	NMAX_FREQ	10	§9.4
	AMOR_REDUIT	0.	§5.4

Mot-clé facteur	Mot-clé	Valeur par défaut	Références
	TYPE_QZ ='QZ_SIMPLE'	'QZ_SIMPLE'	§9.4
	'QZ_EQUI'		§9.4
	'QZ_QR		§9.4
	NMAX_ITER_SHIFT	3	[Boi12] §3.2
	PREC_SHIFT	0.05	[Boi12] §3.2
	SEUIL_FREQ	1.E-02	§3.7
VERI_MODE	STOP_ERREUR='OUI'	'OUI'	§3.7
	'NON'		§3.7
	PREC_SHIFT	5.E-03	§3.7
	SEUIL	1.E-06	§3.7
	STURM='OUI'	'OUI'	§3.7
	'NON'		§3.7

Tableau 9.7-1. Récapitulatif du paramètrage de MODE_ITER_SIMULT (GEP) avec METHODE='QZ'.

Remarques:
•On retrouve toute la "tripaille" de paramètres liée aux pré-traitements du test de Sturm (NMAX_ITER_SHIFT, PREC_SHIFT) et aux post-traitements de vérification (SEUIL_FREQ, VERI_MODE).
•Lors des premiers passages, il est fortement conseillé de ne modifier que les paramètres principaux notés en gras. Les autres concernent plus les arcanes de l'algorithme et ils ont été initialisés empiriquement à des valeurs standards.
•L'algorithme QZ est censé fournir les valeurs «les plus fiables possibles» des modes recherchés. Toutefois, lorsqu'il calcule tout le spectre d'un problème de taille non négligeable (> 100 ddls), il peut fournir des valeurs «approchées». N'étant pas un algorithme avec redémarrage (cf. IRAM) ou avec une étape de projection (cf. Lanczos/IRAM), l'utilisateur ne dispose d'aucun «levier» numérique pour améliorer la convergence des modes. Tout au plus, peut il changer d'option de calcul (TYPE_QZ), en rajoutant ou en enlevant les pré-traitements d'équilibrage (cf. §9.4).

10 Parallélisme et calcul intensif

10.1 Opérateurs intégrés

Les opérateurs intégrés MODE_ITER_SIMULT/INV et INFO_MODE ne peuvent bénéficier que d'un seul niveau de parallélisme: celui intrinsèque du solveur linéaire MUMPS.
Cependant, l'efficacité parallèle de MUMPS dans un calcul modal est plus limitée que pour d'autres types d'analyses. On constate, en général, une efficacité parallèle en temps de l'ordre de 0.2 à 0.3 et ce, pour une faible gamme de processeurs: de 2 à 16. Au delà on ne gagne plus rien.
Cela s'explique notamment par:
- la quasi-singularité de certaines matrices dynamiques de travail,
- un ratio « nombre de descente-remontées/nombre de factorisations » très défavorable,
- le coût important de la phase d'analyse par rapport à celui de la factorisation,
- un ratio « coût en temps/mémoire du solveur linéaire »/« coût en temps/mémoire du solveur modal » très défavorable.

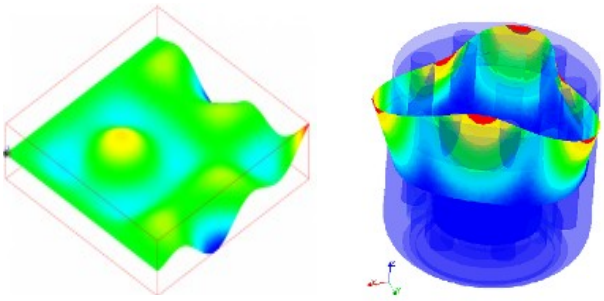
Pour plus d'informations techniques et fonctionnelles, on peut consulter les documentations [R6.02.03], [U4.50.01] et [U2.08.03/06].

Afin de progresser en terme de performances, on propose de décomposer le calcul initial en sous-calculs plus performants et plus précis: c'est l'objet de la macro-commande MACRO_MODE_MECA détaillée dans paragraphe suivant. De plus, cette réécriture algorithmique du problème exhibe deux niveaux de parallélisme plus pertinents et plus efficaces pour **« booster » les calculs modaux de Code_Aster**.

10.2 Macro-commande MACRO_MODE_MECA

Pour traiter efficacement de gros problèmes modaux (en taille de maillage et/ou en nombre de modes recherchés), on conseille l'usage de la macro-commande: **MACRO_MODE_MECA**. Elle décompose le calcul modal d'un GEP standard (symétrique et réel), en une succession de sous-calculs indépendants, moins coûteux, plus robustes et plus précis.
Rien qu'en séquentiel, les **gains peuvent être notables**: facteurs 2 à 5 en temps, 2 ou 3 en pic RAM et 10 à 10^4 sur l'erreur moyenne des modes.
De plus, **son parallélisme multi-niveaux,** en réservant une soixantaine de processeurs, **peut procurer des gains supplémentaires** de l'ordre de 20 en temps et 2 en pic RAM (cf. tableaux 10-1). Et ce, sans perte de précision, ni restriction de périmètre et avec le même comportement numérique.

Cas-test perf016a (N=4M, 50 modes) découpage en **8 sous-bandes**	**Temps elapsed**	**Pic mémoire RAM**
1 processeur	5524s	16.9Go
8 processeurs	1002s	19.5Go
32 processeurs	643s	13.4Go
découpage en **4 sous-bandes**		
1 processeur	3569s	17.2Go
4 processeurs	1121s	19.5Go
16 processeurs	663s	12.9Go



Etude sismique (N=0.7M, 450 modes) découpage en **20 sous-bandes**	**Temps elapsed**	**Pic mémoire RAM**
1 processeur	5200s	10.5Go
20 processeurs	407s	12.1Go
80 processeurs	270s	9.4Go
découpage en **5 sous-bandes**		
1 processeur	4660s	8.2Go
5 processeurs	1097s	11.8Go
20 processeurs	925s	9.5Go

Figures-Tableaux 10-1a/b. Quelques résultats de tests de MACRO_MODE_MECA parallèle avec les paramètres par défaut (+ SOLVEUR=MUMPS en IN_CORE et RENUM='QAMD').
Code_Aster v11.3.11 sur la machine IVANOE (1 ou 2 processus MPI par noeud).

Le principe de MACRO_MODE_MECA[U4.52.02] repose sur le fait que les coûts calculs et mémoires des algorithmes modaux dépendent plus que linéairement du nombre de modes recherchés. Donc, comme pour la décomposition de domaines|R6.01.03], on va décomposer la recherche de centaines de modes en paquets de taille plus raisonnables.
Un paquet de l'ordre de quarante modes semble un être un optimum empirique en séquentiel. En parallèle, on peut continuer à améliorer les performances en descendant jusqu'à la quinzaine.
L'exemple de la figure 10-2 illustre ainsi un calcul MODE_ITER_SIMULT global dans la bande

$$[freq_{min}, freq_{max}]$$

qui est souvent avantageusement remplacé par dix calculs MODE_ITER_SIMULT ciblés sur des sous-bandes contigües équivalentes

$$[freq_1 = freq_{min}, freq_2], [freq_2, freq_3], ... [freq_{10}, freq_{11} = freq_{max}] \quad .$$

D'autre part, ce type de décomposition permet de:
Réduire les problèmes de robustesse,
Améliorer et homogénéiser les erreurs modales.

En pratique, **cet opérateur modal dédié au HPC** est décomposé en quatre étapes principales:

1.**Précalibration modale** (*via* INFO_MODE) des sous-bandes paramétrées par l'utilisateur:
Donc potentiellement, une boucle de nb_freq calculs indépendants sur chaque position modale de fréquences (cf. [R5.01.04]).

2.**Calcul modal effectif** (*via* MODE_ITER_SIMULT+'BANDE'+TABLE_FREQ) des modes contenus dans chaque sous-bande non vide (en mutualisant les calibrations modales de l'étape n°1):
Donc potentiellement, une boucle de nb_sbande_nonvide < nb_freq calculs indépendants.

3.**Post-vérification** avec un test de Sturm sur les bornes extrêmes des modes calculées (*via* INFO_MODE):
Donc potentiellement, une boucle de 2 calculs indépendants.

4.**Post-traitements** sur tous les modes obtenus: normalisation (*via* NORM_MODE) et filtrage (*via* EXTR_MODE).

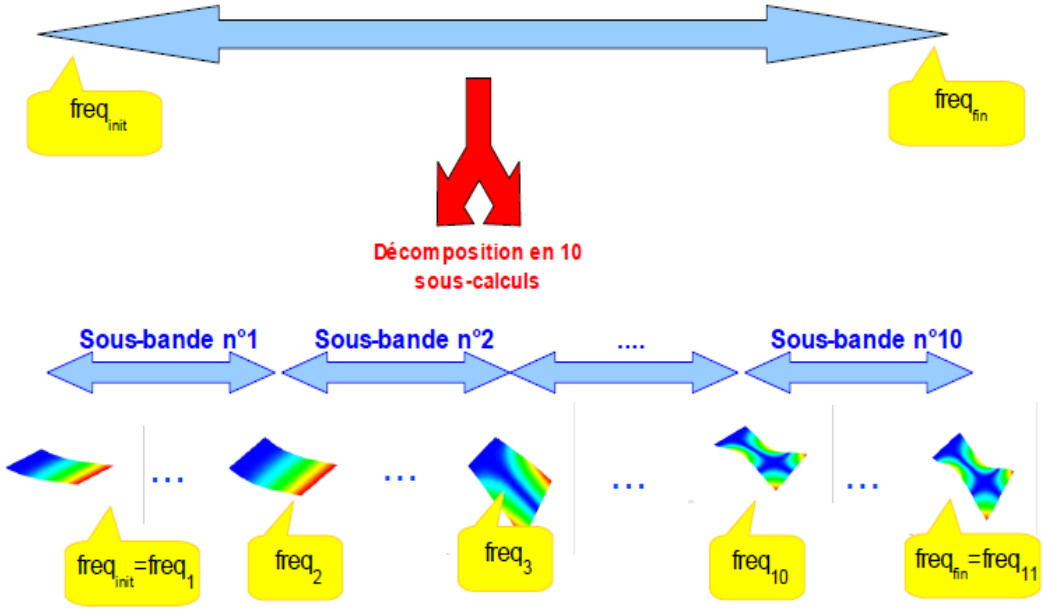


Figure 10-2. Principe de la décomposition des calculs de MACRO_MODE_MECA.

En parallèle, **chacune des étapes de calcul peut dégager au moins un niveau de parallélisme:**
Les deux premières en distribuant les calculs de chaque sous-bande sur le même nombre de paquets de processeurs.
La troisième, en distribuant les positions modales des bornes de l'intervalle de vérification sur deux paquets de processeurs.
La quatrième étape, peu coûteuse, reste séquentielle.

Si le nombre de processeurs et le paramétrage le permettent (notamment, si on utilise le solveur linéaire MUMPS), on peut exploiter un deuxième niveau de parallélisme.
La figure 10-3 illustre un calcul modal cherchant à tirer profit de 40 processeurs en décomposant le calcul initial en dix sous-bandes de recherche. Chacune bénéficie de l'appui de 4 occurrences MUMPS pour les inversions de systèmes linéaires intensivement requises par les solveurs modaux.

Pour une présentation exhaustive de ce parallélisme multi-niveaux, de ses enjeux et de quelques détails techniques et fonctionnels, on peut consulter les documentations [R5.01.04], [U4.52.01/02] et [U2.08.06].

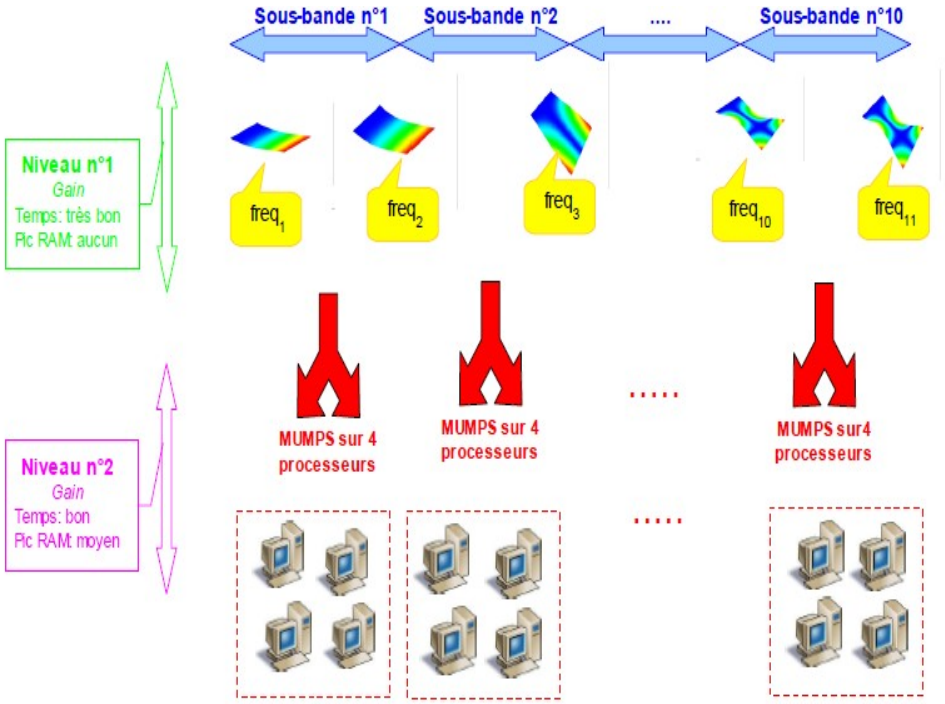


Figure 10-3. Exemple de deux niveaux de parallélisme dans l'INFO_MODE de pré-traitement et dans la boucle des MODE_ITER_SIMULT de MACRO_MODE_MECA.
Distribution sur nb_proc=40 processeurs avec un découpage en 10 sous-bandes (parallélisme dit « 10x4 »).
On utilise ici le solveur linéaire MUMPS et la paramétrage du parallélisme par défaut ('COMPLET').

Remarques:
•En parallélisme MPI, les principales étapes concernent la distribution des tâches et leurs communications. Pour MACRO_MODE_MECA, la distribution s'effectue dans le python de la macro ainsi que dans le fortran. Les deux communiquent par des mot-clés cachés: PARALLELISME_MACRO. Mais tous les appels MPI sont limités aux seules couches F77/F90.
•Les communications globales du premier niveau, celles des valeurs et vecteurs propres s'effectuent en fin de MODE_ITER_SIMULT. A un niveau intermédiaire, entre la simple communication de résultats d'algèbre linéaire (du type de ce qui est fait autour de MUMPS/PETSc) et la communication de structures de données Aster dans le python après filtrage (optimale en terme de performance mais beaucoup plus compliquée à mettre en oeuvre).
•L'idéal aurait été de pouvoir équilibrer empiriquement les sous-bandes de fréquences pour limiter les déséquilibrages de charge liés à la répartition des modes par sous-bandes et ceux liés au calcul modal proprement dit. Comme pour FETI, on aurait pu ainsi prévoir 2, 4 ou 8 calculs de sous-bandes par processeur. Cela permettrait aussi de bénéficier des gains de la décomposition de calcul de la macro, même sur peu de processeurs. Mallheureusement, des contingences informatiques de manipulation de concepts utilisateurs potentiellement vides n'ont pas permis de valider ce scénario plus ambitieux.

11 Bibliographie

11.1 Livres/articles/proceedings/thèses…

[AHLT05] P.Arbenz, U.L.Hetmaniuk, R.B.Lehoucq & R.S.Tuminaro. *A comparison of eigensolvers for large-scale 3d modal analysis using AMG-Preconditioned Iterative Methods* . Int. J. Numer. Meth. Engng, vol. 64, pp204-236 (2005).
[Arn51] W.E.Arnoldi. *The principle of minimized iterations in the solution of the matrix eigenvalue problem*. Quart. Appl. Math, vol 9, pp17-19 (1951).
[Bat71] K.J.Bathe *Solution methods for large generalized eigenvalue problems in structural engineering.* Ed. California university Berkeley (1971).
[Ber99] O.Bertrand. *Procédures de dénombrement de valeurs propres*. Thèse de l'université de Rennes 1 (1999).
[BDK93] Z.Bai, J.Demmel & A.M.Kenney. *On computing condition numbers for the nonsymmetric eigenproblem* . ACM transactions on mathematical software, vol 19, pp202-223 (1993).
[BP02] Bergamaschi & M.Putti. *Numerical comparison of iterative eigensolvers for large symmetric positive definite matrices*. Comput. Methods Appli. Mech. Engrg. 191 5233-5247 (2002).
[Cha88] F.Chatelin. *Valeurs propres de matrices.* Ed. Masson (1988).
[CW85] J.K.Cullum & R.A.Willoughby. *Lanczos algorithms for large symetric eigenvalue computations* . Vol 1 Theory, Ed Birkhäuser (1985).
[DMW71] A.Dubrulle, R.S.Martin & J.H.Wilkinson. *The Implicit **QL** Algorithm* . Handbook for automatic computation. Vol 2, Linear algebra, Springer-Verlag (1971).
[ER80] T.Ericsson & A.Ruhe. *The spectral transformation Lanczos method for the numerical solution of large sparse generalised symmetric eigenvalue problems* . Mathematics of computations, vol 35, pp1251-1268 (1980).
[GL89] G.H.Golub & C.F.Van Loan. *Matrix computations* . The Johns Hopkins university press (1989).
[GSF92] G.Gambolati, F.Sartoretto & P.Florian. *An orthogonal accelerated deflation technique for large symmetric eigenproblems*. Comp. Meth. Appl. Mech. Engrg., 94 13-23 (1992).
[Hau80] Y.Haugazeau. *Application du théorème de Sylsvester à la localisation des valeurs propres*. RAIRO analyse numérique, vol. 14, 1, pp25-41 (1980).
[HRTV07] V.Hernandez, J.E.Roman, A.Tomas & V.Vidal. *A survey of software for sparse eigenvalue problems*. Rapport SLEPc disponible à http://www.grycap.upv.es/slepc.
[Imb91] J.F.Imbert. *Analyse des structures par éléments finis*. Ed. CEPADUES (1991).
[Kny91] A.V.Knyazev. *Toward the optimal preconditioned eigensolver: locally optimal block preconditioned conjugate gradient method* . SIAM J. Sci. Comput., 23 pp517-541 (2001).
[Lan50] C. Lanczos. *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators* . Journal. of research. of the national bureau of standards, vol 45, n°4 pp255-282 (1950).
[LS96] R.B.Lehoucq & J.A.Scott. *An evaluation of software for computing eigenvalues of sparse nonsymmetric matrices*. Rapport Argonne National Laboratory MCS-P547-1195.
[LT86] P.Lascaux & R.Theodor. *Analyse numérique matricielle appliquée à l'Art de l'ingénieur*. Masson (1986).
[MPW71] R.S.Martin, G.Peters & J.H.Wilkinson. *The **QR** Algorithm for Real Hessenberg Matrices* . Handbook for automatic computation,. Vol. 2, Linear algebra, Springer-Verlag (1971).
[MS73] C.B.Moler & G.W.Stewart. *An algorithm for GEP* . SIAM J. Num. Anal., 10, 241-256 (1973).
[MS93] R.B.Morgan & D.S.Scott. *Preconditionning the Lanczos algorithm for sparse symmetric eigenvalue problems* . SIAM J. Sci. Comp, vol 14 (1993).
[NP77] M.Newmann & A.Pipano. *Fast modal extraction in NASTRAN via FEER computer programs, NASTRAN, user manuel* , NASA Langley Research Center pp485-506 (1977).
[Osb60] E.E.Osborne. *On pre-conditioning of matrices* . J.Assoc.Comput.Mach., vol 7, pp338-345 (1960).
[Pai00] C.C.Paige. *Accuracy and effectivness of the Lanczos algorithm for the symmetric eigenproblem* . Linear algebra and its applications, vol 34 (1980).
[Par80] B.N.Parlett. *The symetric eigenvalue problem* . Prentice Hall (1980).
[PR71] B.N.Parlett & C.Reinsch. *Balancing a matrix for calculation of eigenvalues and eigenvectors* . Handbook for automatic computation. Vol. 2, Linear algebra, Springer-Verlag (1971).
[Saa80] Y.Saad. *On the rates of convergences of the Lanczos and the Block-Lanczos methods* . SIAM J. Numer. Anal., vol 17 N° 5 , pp687-706 (1980).
Y.Saad. *Variation on Arnoldi's method for computing eigenelements of large unsymetric matrices* . Linear algebra and its applications, vol 34, pp269-2395 (1980).
[Saa91] Y.Saad. *Numerical methods for large eigenvalue problems* . Ed. Manchester university (1991).
[Sad93] M. Sadkane. *A block Arnoldi-Tchebychev method for computing the leading ergenpairs of large sparse unsymmetric matrices* . Numerische mathematik, vol 64, pp181-193 (1993).
[Sor92] D.C.Sorensen. *Implicit applications of polynomial filters in a k-step Arnoldi method*. SIAM J. Matrix Anal. Appl., vol 13, pp357-385 (1992).
[SV96] G.L.G.Sleijpen & H.Van der Vorst. *A jacobi-Davidson method for linear eigenvalue problems.* SIAM J. Matrix Anal. Appl., 17 pp401-425 (1996).
[TM01] F.Tisseur & K.Meerbergen. *The quadratic eigenvalue problem.* SIAM Review vol 43, n°2, pp235-286 (2001).
[Wat07] D.S.Watkins. *The matrix eigenvalue problem.* SIAM (2007).
[WR71] J.H.Wilkinson & C.Reinsch. *Handbook for automatic computation*, vol 2, Linear Algebra, Springer-Verlag (1971).

11.2 Rapports/compte-rendus EDF

[Beg06] M.Begon. *Implantation de solveurs modaux dans Code_Aster*. Rapport de Master de l'INSTN (2006).
[BQ00] O.Boiteau & B.Quinnez. *Méthode d'analyse spectrale dans Code_Aster* . Fascicule du Cours de dynamique disponible sur le site internet http://www.code-aster.org (2000).
[Boi08] O.Boiteau. *Généralités sur les solveurs linéaires directs et utilisation du produit MUMPS*. Documentation de Référence *Code_Aster* R6.02.03 (2008).

[Boi08b] O.Boiteau. *Intégration d'un solveur modal de type QZ dans Code_Aster* et *Extension des solveurs QZ et Sorensen aux problèmes modaux non symétriques de Code_Aster*. Comptes-rendu internes EDF R&D CR-I23/2008/030 et CR-I23/2008/044 (2008).
[Boi09] O.Boiteau. *Résolution du problème modal quadratique (QEP)*. Documentation de Référence *Code_Aster* R5.01.02 (2009).
[Boi09b] O.Boiteau. *Généralités sur le gradient conjugué: GCPC Aster et utilisation du produit PETSc*. Documentation de Référence *Code_Aster* R6.01.02 (2009).
[Boi12] O.Boiteau. *Procédure de dénombrement de valeurs propres*. Documentation de Référence *Code_Aster* R5.01.04 (2012).
[Boy07] E.Boyère. *Opérateur* DYNA_TRAN_MODAL. Documentation d'Utilisation Code_Aster U4.53.21 (2007).
[Pel01] J.Pellet. *Dualisation des conditions limites.* Documentation de Référence Code_Aster R3.03.01 (2001).
[Ros07] C.Rose. *Méthode multifrontale*. Documentation de Référence *Code_Aster* R6.02.02 (2007).
[Vau00] J.L.Vaudescal. *Introduction aux méthodes de résolution de problèmes aux valeurs propres de grande taille*. Note EDF R&D HI-72/00/01 (2000).

11.3 Ressources internet

[Arp] Site web d'ARPACK : http://www.caam.rice.edu/software/ARPACK/ .
[Lap] Site web de LAPACK: http://www.netlib.org/lapack/.
[MaMa] Site web de MatrixMarket : http://math.nist.gov/MatrixMarket/index.html .
[Sca] Site web de Scalapack: http://www.netlib.org/scalapack/ .

12 Description des versions du document

Version Aster	Auteur(s) Organisme(s)	Description des modifications
3	D.SELIGMANN-R&D/MMN	Texte initial
...	...	
5	O.BOITEAU EDF-R&D/MMN	
V8.4	O.BOITEAU EDF-R&D/SINETICS O.NICOLAS EDF-R&D/AMA	
9.4	O.BOITEAU EDF-R&D/SINETICS	Refonte du document, et extension du périmètre du solveur QZ au non symétrique
v10.4	O.BOITEAU EDF-R&D/SINETICS	Gros travail de remise en forme (formule, figure, légende, remarque, caractère blancs, fonte...). Mise à jour concernant les paramètres (en particulier NPREC, COEF_DIM_ESPACE), les aspects solveurs linéaires et le parallélisme.
V11.2	O.BOITEAU EDF-R&D/SINETICS	Gros travail de remise en forme (2ième couche) + corrections fautes de réécriture de formules. Mise en cohérence avec la nouvelle doc. R5.01.04 (procédure de dénombrement de valeurs propres).
V11.2.13	O.BOITEAU EDF-R&D/SINETICS	Encore quelques remises en forme et prise en compte EL16710.
V11.2.19	O.BOITEAU EDF-R&D/SINETICS	EL19643
V11.3.20	O.BOITEAU EDF-R&D/SINETICS	Paragraphe sur MACRO_MODE_MECA + le parallélisme mono et bi-niveaux (EL20309/17078/20604).

13 Annexe 1. Généralités sur l'algorithme QR

13.1 Principe

Les algorithmes de type QR ont été pressentis par H.Rutishauser(1958) et formalisés concurremment par J.C.Francis et V.N.Kublanovskaya(1961). Cette méthode fondamentale est souvent impliquée dans les autres approches mieux adaptées pour traiter les problèmes de grandes tailles (en particulier les méthodes de projection).

Pour $k=1,\ldots$ faire
$\mathbf{H}_k=\mathbf{Q}_k\mathbf{R}_k$ (factorisation $\mathbf{QR}$),
$\mathbf{H}_{k+1}=\mathbf{R}_k\mathbf{Q}_k$,
Fin boucle.

Algorithme 1.1. QR théorique.

Le **processus conduit itérativement** vers une matrice $\mathbf{H}_k$ **triangulaire supérieure** (ou triangulaire par blocs) dont les termes diagonaux sont les valeurs propres de l'opérateur initial $\mathbf{H}=\mathbf{H}_1$. La notation $\mathbf{H}$ n'est en fait pas innocente, car on a tout intérêt à préalablement transformer orthogonalement (de manière à ne modifier que la forme de l'opérateur shifté et non son spectre) l'opérateur de travail $\mathbf{A}$ sous forme de Hessenberg supérieur, soit en arithmétique réelle

$$\mathbf{H}_1=\mathbf{Q}_0^{\mathrm{T}}\mathbf{A}\mathbf{Q}_0$$

Ceci peut s'effectuer *via* diverses transformations orthogonales (Householder, Givens, Gram-Schmidt...) et leur coût (de l'ordre de $O(10\mathrm{n}^3/3)$) est négligeable comparé au gain qu'elles permettent de réaliser à chaque itération du processus global: $O(n^2)$ (avec Householder ou Fast-Givens) contre $O(n^3)$. Ce gain d'un ordre de magnitude peut même être amélioré lorsque l'opérateur est tridiagonal symétrique (c'est le cas de Lanczos avec un vrai produit scalaire): $O(20\mathrm{n})$.

La convergence vers une matrice triangulaire simple ne s'effectue que si toutes les valeurs propres sont de modules distincts et que si la matrice initiale n'est pas «pathologiquement» trop pauvre dans les directions propres. La convergence du i ème mode (rangés classiquement par ordre décroissant de module) s'effectue alors en:

$$\max_{j^1 i}\frac{|\lambda_j|}{|\lambda_i|}$$

ce qui peut s'avérer très lent si aucun processus complémentaire n'est mis en œuvre.

Remarques:
•*La détermination du spectre de la matrice de Rayleigh avec la variante de Newmann & Pipano est effectuée via un QR (ou un QL en symétrique) simple de ce type (avec au préalable une procédure d'équilibrage). Le seul paramètre accessible par l'utilisateur est le nombre maximum d'itérations admissibles* `NMAX_ITER_QR`.
•*Pour IRAM, ce calcul est réalisé via une méthode QR avec double shift explicite alors que les filtres polynomiaux gérant les restarts utilisent un QR avec double shift implicite. Par prudence, aucun paramètre n'est accessible pour l'utilisateur dans Code_Aster !*
•*Il ne faut pas confondre la méthode, l'algorithme QR, et un de ses outils conceptuels, la factorisation QR.*
•Cette classe d'algorithme est très utilisée pour déterminer le spectre complet d'un opérateur, car elle est très robuste (c'est la référence dans ce domaine). Cependant elle est très gourmande en place mémoire ce qui rend son utilisation rédhibitoire sur de grands systèmes.
•Le périmètre d'application de l'algorithme QR est beaucoup plus général que celui de Jacobi (c'est le deuxième algorithme standard fournissant tout le spectre d'un opérateur pour peut que l'on soit prêt à le stocker entièrement) qui est limité aux matrices hermitiennes.

Pour accélérer la convergence de l'algorithme simple qui peut être très lente (en présence de clusters par exemple) une multitude de variantes, basées sur le choix de shifts répondant à certains critères, ont vu le jour.

13.2 La stratégie du shift

Cette stratégie consiste à **susciter artificiellement un phénomène de déflation** (cf. §4.1, §6.4) au sein de la matrice de travail. Cela offre le triple avantage:
•de pouvoir **isoler une valeur propre** réelle voire deux valeurs propres complexes conjuguées,
•tout en **réduisant la taille** du problème à traiter,
•et en **accélérant la convergence**.

Dans sa version avec simple shift explicite, la méthode se réécrit alors sous la forme suivante:

Pour $k=1,\ldots$ faire
  Choisir le shift μ,
  $\mathbf{S}_k=\mathbf{H}_k-\mu\mathbf{I}$,
  $\mathbf{S}_k=\mathbf{Q}_k\mathbf{R}_k$ (factorisation $\mathbf{QR}$),
  $\mathbf{H}_{k+1}=\mathbf{R}_k\mathbf{Q}_k+\mu\mathbf{I}$,
Fin boucle.

Algorithme 1.2. QR avec simple shift explicite.

Remarque:
•Ce processus se généralise intuitivement à plusieurs shifts. On construit alors, pour chaque itération globale k, autant de matrices auxiliaires $\mathbf{S}_k^i$ que de shift μ_i.

La convergence du processus est grandement améliorée dans le sens où les termes sous-diagonaux s'annulent asymptotiquement en

$$\max_{j\neq i}\frac{|\lambda_j-\mu|}{|\lambda_i-\mu|}$$

En théorie, si ce shift μ est valeur propre du problème, alors la déflation est exacte. En pratique, les effets d'arrondi perturbent ce phénomène, c'est ce qu'on appelle la **propriété d'instabilité directe de l'algorithme**. La difficulté principale réside dans le choix du (ou des) shifts. D'autre part, on ne garde pas le même shift pour toutes les itérations. On doit en changer lorsqu'il est associé à une valeur propre convergée. En effet, il aura numériquement provoqué son éviction du spectre de travail en suscitant une déflation à l'itération précédente.

Depuis les années soixante toute **une zoologie de shifts s'est développée**. Tandis que la plus simple utilise le dernier terme diagonal $\mathbf{H}_{n,n}$ celle de **J.H. Wilkinson**[WR71] consiste à déterminer analytiquement la valeur propre μ du bloc diagonal

$$\begin{bmatrix}\mathbf{H}_{n-1,n-1} & \mathbf{H}_{n-1,n}\\ \mathbf{H}_{n,n-1} & \mathbf{H}_{n,n}\end{bmatrix}$$

la plus proche de ce terme. Cette technique permet d'obtenir une convergence quadratique voire cubique (dans le cas symétrique) et elle s'avère particulièrement efficace pour capturer des modes doubles ou des valeurs propres distinctes de même module. Cependant cette stratégie peut se révéler inefficace en présence de modes propres complexes conjugués.

Le même auteur a alors proposé une **variante incluant un double shift** correspondant aux deux valeurs propres complexes μ_1 et μ_2 (déterminées préalablement) du bloc incriminé. Mais outre les difficultés numériques à réfréner l'apparition de composantes complexes envahissantes (qui en théorie n'ont pas lieu d'être), on a d'autre part beaucoup de mal à conserver le caractère de «Hessenberg supérieure» de la matrice de travail. Au mieux, cela ralentie la convergence de l'algorithme QR, au pire, cela fausse complètement son fonctionnement.

Afin de remédier à ces inconvénients numériques, J.C.Francis a mis au point une version avec **double shift implicite**. Elle est très bien explicitée dans [GL89](pp377-81) et on se contentera ici de la résumer.

Pour minimiser les effets d'arrondis, il faudrait constituer directement la matrice auxiliaire $\mathbf{S}_k$ résultant de l'application simultanée des deux shifts μ_1 et μ_2

$$\mathbf{S}_k=\mathbf{H}_k^2-\left(\mu_1+\mu_2\right)\mathbf{H}_k+\left(\mu_1\mu_2\right)\mathbf{I}$$

avant de la factoriser sous forme QR et de construire le nouvel itéré $\mathbf{H}_{k+1}$

$$\mathbf{S}_k=\mathbf{Q}_k\mathbf{R}_k,$$
$$\mathbf{H}_{k+1}=\mathbf{Q}_k^T\mathbf{H}_k\mathbf{Q}_k.$$

Mais le seul coût de l'assemblage initial (de l'ordre de $O(n^3)$) rend la tactique inopérante. Cette variante, basée sur le théorème $\mathbf{Q}$ -implicite, consiste à appliquer à la matrice de travail des transformations de Householder particulières permettant de retrouver une matrice de Hessenberg «essentiellement» égale à $\mathbf{H}_{k+1}$, c'est à dire du type:

$$.\ \tilde{\mathbf{H}}_{k+1}=\mathbf{E}^{-1}\mathbf{H}_{k+1}\mathbf{E}\ \text{avec}\ \mathbf{E}=\text{diag}\left(\pm1,\ldots\pm1\right)$$

La **matrice de travail conserve** ainsi, au **moindre coût**, **sa structure particulière et son spectre**.

Toutes ces variantes sont en fait très sensibles aux techniques d'équilibrage implantées pour préconditionner l'opérateur initial. Le paragraphe suivant va résumer cette technique de «mise à l'échelle» («balancing» pour les Anglo-saxons) des termes de la matrice de travail qui est très employée en calcul modal.

13.3 Équilibrage

Il s'agit **d'atténuer les effets d'arrondi** en bridant le **périmètre d'expansion des termes de l'opérateur de travail**, c'est à dire d'éviter qu'ils ne deviennent trop petits ou, au contraire, trop grands. À ce sujet, E.E.Osborne [Osb60] (1960) a remarqué que généralement l'erreur sur le calcul des éléments propres de $\mathbf{A}$ est de l'ordre de $\varepsilon\|\mathbf{A}\|_2$ où ε est la précision machine. Il proposa alors de transformer la matrice initiale en une matrice

$$\tilde{\mathbf{A}}=\mathbf{D}^{-1}\mathbf{A}\,\mathbf{D}\quad\text{(avec }\mathbf{D}\text{ une matrice diagonale),}$$

telle que:

$$\|\tilde{\mathbf{A}}\|_2\ll\|\mathbf{A}\|_2$$

En fait on calcule itérativement une succession de matrices

$$\mathbf{A}_k=\mathbf{D}_{k-1}^{-1}\mathbf{A}_{k-1}\mathbf{D}_{k-1}$$

telles que:

$$\mathbf{A}_k\ \underset{k\to\infty}{\to}\ \mathbf{A}_f$$

vérifiant $\|\mathbf{a}^i\|_2=\|\mathbf{a}_i\|_2$ avec $\mathbf{a}^i$ et $\mathbf{a}_i$, respectivement, les i ème colonne et ligne de $\mathbf{A}_f$.

Remarques:
•Cette technique a été généralisée à toute norme matricielle induite par une norme discrète de[33] l^q.
•Son emploi est très répandu en calcul scientifique et notamment parmi les solveurs directs de système d'équations linéaires.

La base de calcul de l'ordinateur, notée β , intervient dans la détermination des termes des matrices $\mathbf{D}_k$. Afin de minimiser les erreurs d'arrondi, on choisit les éléments de $\mathbf{D}_k$ afin qu'ils soient des puissances de cette base.

33 L'espace fonctionnel l^q est l'ensemble des suites complexes $(u_n)_n$ telles que $\sum_n|u_n|^q<\infty$.

Soient $\mathbf{R}_k$ et $\mathbf{C}_k$ les p normes (en pratique on prend souvent $p=2$), respectivement des ligne et colonne i de la matrice $\mathbf{A_{k-1}}$ (i est choisi de telle façon que $i-1\equiv k-1[n]$). En supposant que $\mathbf{R}_k\mathbf{C}_k\neq 0$, on montre alors qu'il existe un unique entier signé σ tel que:

$$\beta^{2\sigma-1}<\frac{\mathbf{R}_k}{\mathbf{C}_k}\leq\beta^{2\sigma+1}$$

Soit $f=\beta^{\sigma}$, on définit la matrice $\bar{\mathbf{D}}_\mathbf{k}$ telle que:

$$\bar{\mathbf{D}}_k=\begin{cases}\mathbf{I}+(f-1)\mathbf{e}_i\mathbf{e}_i^T & \text{si } (\mathbf{C}_k f)^p+(R_k/f)^p<\gamma(\mathbf{C}_k^p+\mathbf{R}_k^p)\\ \mathbf{I} & sinon\end{cases}$$

où $0<\gamma\leq 1$ est une constante et $\mathbf{e}_i$ le i ème vecteur canonique. On construit alors itérativement :

$$\begin{cases}\mathbf{D}_k=\bar{\mathbf{D}}_k\mathbf{D}_{k-1},\\ \mathbf{A}_k=\bar{\mathbf{D}}_k^{-1}\mathbf{A}_{k-1}\bar{\mathbf{D}}_k\end{cases}$$

en initialisant $\mathbf{D}_0=\mathbf{I}$. Le processus s'arrête dès que $\bar{\mathbf{D}}_k=\mathbf{I}$.

Remarques:
•Avant d'effectuer l'équilibrage, une recherche des valeurs propres isolées est effectuée en détectant la présence de lignes et de colonnes quasi-nulles (tous les termes sont nuls, sauf celui placé sur la diagonale). Lorsqu'il en existe on peut, en effectuant des permutations idoines, mettre la matrice de travail sous la forme blocs suivante:

$$\left(\begin{array}{ccc|c|c} * & & * & & \\ & * & & \mathbf{Y} & \mathbf{Z} \\ 0 & & * & & \\ \hline & 0 & & \mathbf{X} & \mathbf{T} \\ \hline & & & * & * \\ & 0 & & 0 & * \\ & & & 0 & * \end{array}\right)$$

•

•Il n'est alors nécessaire d'effectuer l'équilibrage que sur le bloc central $\mathbf{X}$ car les termes diagonaux des deux matrices triangulaires supérieures sont des valeurs propres de la matrice de travail.
•Dans Code_Aster, on a choisi $p=1$ et $\gamma=0.95$ (cf. [PR71]).
•Avant d'appliquer la méthode QR, on commence par équilibrer la matrice de travail et ensuite, on la transforme sous forme de Hessenberg supérieure. Une fois le calcul QR effectué, on remonte des vecteurs de Ritz aux vecteurs propres via l'inverse des transformations orthogonales dues à la mise sous forme Hessenberg et à l'équilibrage. C'est ce procédé qui est mis en place aussi bien dans Lanczos que dans IRAM. Cependant outre le fait qu'il nécessite le stockage de ces matrices orthogonales, il est aussi et surtout extrêmement sensible aux effets d'arrondis. Ainsi, il serait grandement préférable d'estimer les vecteurs propres via une méthode des puissances inverses initialisée par les valeurs propres exhumées.

13.4 La méthode QL

La factorisation QL d'une matrice $\mathbf{A}$ consiste à orthonormaliser ses vecteurs colonnes en commençant par le dernier (contrairement à l'algorithme QR qui débute par le premier) donnant ainsi une matrice $\mathbf{L}$ triangulaire inférieure régulière. On montre d'ailleurs que l'algorithme QL appliqué à l'opérateur $\mathbf{A}$ inversible est équivalent à l'algorithme QR appliqué à $\mathbf{A}^{-*}$ (matrice transposée conjuguée de l'inverse de $\mathbf{A}$).

La méthode mise en place dans *Code_Aster* est identique à la méthode de simple shift de J.H.Wilkinson vue précédemment en adaptant la recherche de ce shift au mineur 2×2 le plus haut.

Remarque:
•Contrairement à QR qui capture les valeurs propres par ordre croissant de module, on obtient ici préférentiellement les modes dominants, puis les autres, par ordre décroissant de module.

14 Annexe 2. Orthogonalisation de Gram-Schmidt

14.1 Introduction

On a vu à plusieurs reprises dans ce document que la **qualité d'orthogonalisation** d'une **famille de vecteurs** est **cruciale** pour le bon **déroulement des algorithmes** et la **qualité des modes** obtenus. Cette tâche est d'ailleurs à réaliser en permanence d'où l'importance d'un algorithme rapide.

Les algorithmes d'orthonormalisation simples sont déduits du processus classique de Gram-Schmidt mais ils sont souvent «conditionnellement stables» (la qualité de leur travail dépend du conditionnement matriciel de la famille à orthonormaliser). Ce défaut de robustesse peut s'avérer problématique pour traiter des situations particulièrement mal conditionnées. On leur préfère alors des algorithmes plus onéreux mais robustes, à base de projections ou de rotations: les transformations spectrales de Householder et de Givens.

En pratique, pour l'implantation de la méthode IRA dans *Code_Aster*, nous avons retenu une **version itérative du processus de Gram-Schmidt** (l'IGSM de Kahan-Parlett[Par80]) Elle réalise un **bon compromis entre la robustesse et la complexité calcul** puisqu'elle est inconditionnellement stable et permet d'orthogonaliser à la précision machine près, en, au maximum, deux fois plus de temps qu'un Gram-Schmidt classique (GS).

Dans les paragraphes suivants nous allons détailler le fonctionnement de l'algorithme de base, ainsi que celui de ces deux principales variantes. La première a été mise en place dans MODE_ITER_INV et la seconde est utilisée dans MODE_ITER_SIMULT. Un comparatif très percutant de ces méthodes est décliné dans [Vau00] (pp33-36) à partir d'un exemple très simple.

14.2 Algorithme de Gram-Schmidt (GS)

Étant donné k **vecteurs indépendants** de $\mathbb{R}^n$, $(\mathbf{x}_i)_{i=1,k}$ on désire obtenir k **vecteurs orthonormaux** (par rapport à un produit scalaire quelconque) $(\mathbf{y}_i)_{i=1,k}$ de l'espace qu'ils engendrent. En d'autres termes, on souhaite obtenir une autre famille orthonormale engendrant le même espace. Le procédé d'orthonormalisation classique de Gram-Schmidt est le suivant:

Pour $i = 1, k$ faire
  Pour $j = 1, i$-1 faire
    calcul de $r_{ji} = (\mathbf{y}_j, \mathbf{x}_i)$
  Fin boucle.
  calcul de $\hat{\mathbf{y}}_i = \mathbf{x}_i - \sum_{j=1,i-1} r_{ji}\mathbf{y}_j$,
  calcul de $r_{ii} = \|\hat{\mathbf{y}}_i\|$,
  calcul de $\mathbf{y}_i = \frac{\hat{\mathbf{y}}_i}{r_{ii}}$
Fin boucle.

Algorithme 2.1. Algorithme de Gram-Schmidt (GS).

Ce procédé est simple mais très instable du fait des erreurs d'arrondi, ce qui a pour effet de produire des vecteurs non orthogonaux. En particulier lorsque les vecteurs initiaux sont presque dépendants cela crée des écarts de magnitude importants dans la deuxième étape du processus

$$\hat{\mathbf{y}}_i = \mathbf{x}_i - \sum_{j=1,i-1} r_{ij}\mathbf{y}_j$$

D'un point de vue numérique, la gestion de ces écarts est très difficile.

Remarque:
•En notant $\mathbf{Q}$ la matrice engendrée par les $(\mathbf{y}_i)_{i=1,k}$, on a donc construit explicitement une factorisation QR de la matrice initiale $\mathbf{X}$ liée aux $(\mathbf{x}_i)_{i=1,k}$. C'est en fait le but de tout procédé d'orthogonalisation.

14.3 Algorithme de Gram-Schmidt Modifié (GSM)

Afin d'évacuer ces instabilités numériques, on réorganise l'algorithme précédent. Mathématiquement équivalent au procédé précédent, celui-ci est alors beaucoup plus robuste car il évite les écarts de magnitude importants entre les vecteurs manipulés dans l'algorithme.

Dans le procédé initial, les vecteurs orthogonaux $\mathbf{y}_i$ sont obtenus sans tenir compte des $i-1$ orthogonalisations précédentes. Avec le Gram-Schmidt Modifié, on **orthogonalise plus progressivement en tenant compte des altérations précédentes** suivant le processus ci-dessous.

Pour $i = 1,k$ faire
  $\hat{\mathbf{y}}_i = \mathbf{x}_i$
  Pour $j = 1,i$-1 faire
    calcul de $r_{ji} = (\mathbf{y}_j, \mathbf{x}_i)$,
    calcul de $\hat{\mathbf{y}}_i = \mathbf{x}_i - r_{ji}\mathbf{y}_j$,
  Fin boucle.
  calcul de $r_{ii} = \|\hat{\mathbf{y}}_i\|$,
  calcul de $\mathbf{y}_i = \dfrac{\hat{\mathbf{y}}_i}{r_{ii}}$
Fin boucle.

Algorithme 2.2. Algorithme de Gram-Schmidt Modifié (GSM).

L'orthonormalité des vecteurs de base est bien meilleure avec ce procédé et elle peut même s'obtenir à la précision machine et à une constante (dépendant du conditionnement de $\mathbf{X}$) près. Cependant, pour traiter des situations particulièrement mal conditionnées, cette stabilité «conditionnelle» peut s'avérer rapidement problématique, d'où le recours à l'algorithme itératif suivant.

Remarque:
•GSM est deux fois plus efficace qu'une méthode de Householder pour obtenir une factorisation QR de la matrice initiale $\mathbf{X}$. Il requiert seulement $O(2nk^2)$ opérations (avec n le nombre de lignes de la matrice).

14.4 Algorithme de Gram-Schmidt Itératif (IGSM)

Pour **s'assurer** néanmoins de l'**orthogonalité à la précision machine près**, on préconise de réaliser une **seconde orthogonalisation**. Et si, à l'issue de cette dernière, l'orthogonalité n'est pas assurée ce n'est plus la peine de recommencer, les quantités manipulées sont alors certainement très proches et leurs écarts oscillent autour de zéro. Cette approche est basée sur une analyse théorique due à W.Kahan et reprise par B.N.Parlett[Par80] (cf. pp105-110).

Pour i = 1, k faire
$\hat{\mathbf{y}}_i = \mathbf{x}_i$,
Pour j = 1, i - 1 faire
Calcul de $r_{ji} = \left(\mathbf{y}_j, \hat{\mathbf{y}}_i\right)$,
Calcul de $\tilde{\mathbf{y}}_i = \hat{\mathbf{y}}_i - r_{ji}\,\mathbf{y}_j$,
Si $\left\|\tilde{\mathbf{y}}_i\right\| \geq \alpha \left\|\hat{\mathbf{y}}_i\right\|$ alors
$\hat{\mathbf{y}}_i \Leftarrow \tilde{\mathbf{y}}_i$,
Exit boucle en j;
Sinon
Calcul de $\tilde{r}_{ji} = \left(\mathbf{y}_j, \tilde{\mathbf{y}}_i\right)$,
Calcul de $\tilde{\tilde{\mathbf{y}}}_i = \tilde{\mathbf{y}}_i - \tilde{r}_{ji}\,\mathbf{y}_j$,
Si $\left\|\tilde{\tilde{\mathbf{y}}}_i\right\| \geq \alpha \left\|\tilde{\mathbf{y}}_i\right\|$ alors
$\hat{\mathbf{y}}_i \Leftarrow \tilde{\tilde{\mathbf{y}}}_i$,
Exit boucle en j;
Sinon
$\hat{\mathbf{y}}_i \Leftarrow \mathbf{0}$,
Passage au i suivant;
Fin si.
Fin boucle.
Calcul de $r_{ii} = \left\|\hat{\mathbf{y}}_i\right\|$,
Calcul de $\mathbf{y}_i = \dfrac{\hat{\mathbf{y}}_i}{r_{ii}}$;
Fin boucle.

Algorithme 2.3. Algorithme de Gram-Schmidt Itératif de type Kahan-Parlett (IGSM).

Lors de l'utilisation d'IRAM dans *Code_Aster*, le critère α est paramétré par le mot-clé PARA_ORTHO_SOREN (cf. §7.5). On montre que son intervalle de validité est $[1.2\,\varepsilon, 0.83-\varepsilon]$ avec ε la précision machine et on lui attribue généralement la valeur 0.717 (par défaut).

Remarques:
•Plus la valeur du paramètre □ est grande, moins la réorthogonalisation se déclenche, mais cela affecte la qualité du processus.
•Contrairement à la version «maison» développée pour Lanczos ici les critères d'arrêts se concentrent sur les normes des vecteurs orthogonalisés plutôt que sur les produits scalaires inter-vecteurs qui ont plus tendance à répercuter les effets d'arrondi. Cela, joint à la suppression des itérations supérieures à deux, donc inutiles, peut expliquer le surcroît d'efficacité de la version de Kahan-Parlett.
•D'après D.C.Sorensen la paternité de cette méthode est plutôt à attribuer à J.Daniel (papier de 1976 soumis à Mathematics of Computation, vol.30, pp772-795).

15 Annexe 3. Méthode de Jacobi

15.1 Principe

La méthode de Jacobi[LT86] permet de calculer toutes les valeurs propres d'un problème généralisé dont les matrices sont définies positives et symétriques (les matrices obtenues à chaque itération par la méthode de Bathe &Wilson vérifient ces propriétés; cf. §8). Elle consiste à transformer les matrices $\mathbf{A}$ et $\mathbf{B}$ du GEP $\mathbf{A}\mathbf{u}=\lambda\mathbf{B}\mathbf{u}$ en des matrices diagonales, en utilisant successivement des transformations semblables orthogonales (matrices de rotation de Givens). Le processus peut se schématiser de la manière suivante :

$$\mathbf{A}^1=\mathbf{A} \qquad \mathbf{B}^1=\mathbf{B}$$
$$\mathbf{A}^2=\mathbf{Q}_1^T\mathbf{A}^1\mathbf{Q}_1 \qquad \mathbf{B}^2=\mathbf{Q}_1^T\mathbf{B}^1\mathbf{Q}_1$$
$$\ldots$$
$$\mathbf{A}^k=\mathbf{Q}_{k-1}^T\mathbf{A}^{k-1}\mathbf{Q}_{k-1} \qquad \mathbf{B}^k=\mathbf{Q}_{k-1}^T\mathbf{B}^{k-1}\mathbf{Q}_{k-1}$$
$$\mathbf{A}^k \xrightarrow[k\to\infty]{} \mathbf{A}^d \text{ matrice diagonale} \qquad \mathbf{B}^k \xrightarrow[k\to\infty]{} \mathbf{B}^d \text{ matrice diagonale}$$

Algorithme 3.1. Processus de Jacobi

Les valeurs propres sont données par $\lambda=\mathbf{A}^d(\mathbf{B}^d)^{-1}$ soit $\lambda=\dfrac{\mathbf{A}_{ii}^d}{\mathbf{B}_{ii}^d}$ et la matrice des vecteurs propres vérifie:

$$\mathbf{X}=\mathbf{Q}^1\mathbf{Q}^2\ldots\mathbf{Q}^k\ldots\begin{pmatrix}1/\sqrt{\mathbf{A}_{11}^d} & & \\ & 1/\sqrt{\mathbf{A}_{22}^d} & \\ & & 1/\sqrt{\mathbf{A}_{nn}^d}\end{pmatrix}$$

Chaque matrice $\mathbf{Q}_k$ est choisie de manière à ce qu'un terme (i,j) diagonal et non nul de $\mathbf{A}_k$ ou de $\mathbf{B}_k$ soit nul après la transformation.

15.2 Quelques choix

Dans cet algorithme, on se rend compte que les points importants sont les suivants:
•Comment choisir les termes à annuler ?
•Comment mesurer le caractère diagonal des matrices quand k tend vers l'infini ?
•Comment mesurer la convergence ?

15.2.1 Termes non diagonaux à annuler

Pour le choix des termes à annuler, il existe plusieurs méthodes:
•La première consiste à chaque étape k, à choisir le plus grand élément en module non diagonal de la matrice $\mathbf{A}_k$ ou $\mathbf{B}_k$ et à l'annuler en effectuant une rotation. Ce choix assure la convergence de la méthode de Jacobi mais coûte relativement cher (recherche de l'élément maximal).
•La deuxième solution consiste à annuler successivement tous les éléments non diagonaux de ces matrices en suivant l'ordre naturel $\mathbf{A}_{13}^k,\ldots.,\mathbf{A}_{1n}^k,\mathbf{A}_{23}^k,\ldots.$. Quand on arrive à $\mathbf{A}_{n-1,n}^k$, on recommence le cycle. Cette méthode converge lentement.
Une variante de cette méthode, consiste tout en suivant l'ordre naturel des termes, à annuler seulement ceux qui sont supérieurs à une précision ε_k donnée. Au bout d'un cycle, on diminue la valeur de ce critère et on recommence. C'est cette stratégie qui est utilisée dans *Code_Aster*.

15.2.2 Test de convergence

Pour tester la convergence et le caractère diagonal des matrices, on opère ainsi. On vérifie que tous les facteurs de couplage définis par:

$$f_{\mathbf{A}_{ij}}=\frac{|\mathbf{A}_{ij}|}{\sqrt{|\mathbf{A}_{ii}\mathbf{A}_{jj}|}} \qquad f_{\mathbf{B}_{ij}}=\frac{|\mathbf{B}_{ij}|}{\sqrt{|\mathbf{B}_{ii}\mathbf{B}_{jj}|}} \qquad i\neq j$$

sont inférieurs à une précision donnée (caractère diagonal des matrices). On contrôle également la convergence des valeurs propres via l'indicateur

$$f_{\lambda}=\max_{i}\frac{|\lambda_i^k-\lambda_i^{k-1}|}{\lambda_i^{k-1}}$$

afin qu'il reste inférieur à une précision donnée ε_{jaco} .

15.2.3 Algorithme implanté dans *Code_Aster*

L'algorithme mis en oeuvre dans *Code_Aster* se résume à:

Initialisation de la matrice des vecteurs propres à la matrice identité.
Initialisation des valeurs propres.
Définir la précision de convergence dynamique requise.
Définir la précision globale ε_{glob} .
Pour chaque cycle $k=1,\mathrm{n\_max}_{jaco}$
  Définir la tolérance dynamique : $\varepsilon_k=(\varepsilon_{dyn})^k$,
  $l=0$,
  Pour chaque ligne $i=1,n$
    Pour chaque colonne $j=1,n$
  Calcul des facteurs de couplage, $f_{\mathbf{A}_{ij}^{k,l}}=\frac{|\mathbf{A}_{ij}^{k,l}|}{\sqrt{|\mathbf{A}_{ii}^{k,l}\mathbf{A}_{jj}^{k,l}|}}$ $f_{\mathbf{B}_{ij}^{k,l}}=\frac{|\mathbf{B}_{ij}^{k,l}|}{\sqrt{|\mathbf{B}_{ii}^{k,l}\mathbf{B}_{jj}^{k,l}|}}$ $i\neq j$
      Si $f_{\mathbf{A}_{ij}^{k,l}}$ ou $f_{\mathbf{B}_{ij}^{k,l}}\geq\varepsilon_k$ alors
        Calcul des coefficients de la rotation de Givens,
        Transformation des matrices $\mathbf{A}^{k,l}$ et $\mathbf{B}^{k,l}$,
        Transformation des vecteurs propres,
        $l=l+1$
      Fin si.
    Fin boucle.
  Fin boucle.
  Calcul des valeurs propres $\lambda_i^k=\frac{\mathbf{A}_{ii}^{k,l}}{\mathbf{B}_{ii}^{k,l}}$.
  Calcul de $f_{\lambda}=\max_{i}\frac{|\lambda_i^k-\lambda_i^{k-1}|}{\lambda_i^{k-1}}$.
  Calcul des facteurs de couplage globaux

$$f_{\mathbf{A}}=\max_{\substack{i,j\\ i\neq j}}\frac{|\mathbf{A}_{ij}^{k,l}|}{\sqrt{|\mathbf{A}_{ii}^{k,l}\mathbf{A}_{jj}^{k,l}|}} \qquad f_{\mathbf{B}}=\max_{\substack{i,j\\ i\neq j}}\frac{|\mathbf{B}_{ij}^{k,l}|}{\sqrt{|\mathbf{B}_{ii}^{k,l}\mathbf{B}_{jj}^{k,l}|}}$$

  Si $f_{\mathbf{A}}\leq\varepsilon_{glob}$ et $f_{\mathbf{B}}\leq\varepsilon_{glob}$ et $f_{\lambda}\leq\varepsilon_{glob}$ alors
    Correction des modes propres (divisons par $\sqrt{\mathbf{B}_{ii}^k}$),
    Exit;
  Fin si.
Fin boucle.

Algorithme 3.1. Méthode de Jacobi implantée dans Code_Aster.

On note $\mathrm{n\_max}_{jaco}$ le nombre maximum d'itérations permises.

Remarque:
•Les matrices $\mathbf{A}$ et $\mathbf{B}$ étant symétriques, seulement la moitié des termes est stockée sous forme d'un vecteur.

15.2.4 Matrice de rotation de Givens

On cherche à chaque étape à annuler les termes se trouvant en position i et j $(i \neq j)$ des matrices $\mathbf{A^{k,l}}$ et $\mathbf{B^{k,l}}$ en les multipliant par une matrice de rotation qui a la forme suivante:

$$\mathbf{G}_{ll}=1 \quad l=1,n \;\; ; \;\; \mathbf{G}_{ij}=a \;\; ; \;\; \mathbf{G}_{ji}=b$$

les autres termes étant nuls. On souhaite avoir $\mathbf{A}_{ij}^{k,l+1}=\mathbf{B}_{ij}^{k,l+1}=0$ ce qui conduit à:

$$\begin{cases} a\,\mathbf{A}_{ii}^{k,l}+(1+ab)\,\mathbf{A}_{ij}^{k,l}+b\,\mathbf{A}_{jj}^{k,l}=0 \\ a\,\mathbf{B}_{ii}^{k,l}+(1+ab)\,\mathbf{B}_{ij}^{k,l}+b\,\mathbf{B}_{jj}^{k,l}=0 \end{cases}$$

car $\mathbf{A}^{k,l+1}=\mathbf{G}^T\mathbf{A}^{k,l}\mathbf{G}$ et $\mathbf{B}^{k,l+1}=\mathbf{G}^T\mathbf{B}^{k,l}\mathbf{G}$. Si les deux équations sont proportionnelles on a:

$$a=0 \quad \text{et} \quad b=\frac{-\mathbf{A}_{ij}^{k,l}}{\mathbf{A}_{jj}^{k,l}}$$

sinon:

$$a=\frac{c_2}{d} \qquad b=\frac{-c_1}{d}$$

avec:

$$c_1=\mathbf{A}_{ii}^{k,l}\mathbf{B}_{ij}^{k,l}-\mathbf{B}_{ii}^{k,l}\mathbf{A}_{jj}^{k,l} \qquad c_2=\mathbf{A}_{jj}^{k,l}\mathbf{B}_{ij}^{k,l}-\mathbf{B}_{jj}^{k,l}\mathbf{A}_{ij}^{k,l}$$

$$c_3=\mathbf{A}_{ii}^{k,l}\mathbf{B}_{jj}^{k,l}-\mathbf{B}_{ii}^{k,l}\mathbf{A}_{jj}^{k,l} \qquad d=\frac{c_3}{2}+sign(c_3)\sqrt{\left(\frac{c_3}{2}\right)^2+c_1c_2}$$

Remarque:
•Si $d=0$ alors on est dans le cas proportionnel.