Manuel de Référence
Fascicule R5.01 : Analyse modale
Document : R5.01.01

Algorithme de résolution pour le problème généralisé

Résumé

Dans ce document, nous présentons les algorithmes de résolution pour les problèmes modaux généralisés qui sont implantés dans le *Code_Aster* via les opérateurs MODE_ITER_INV et MODE_ITER_SIMULT :

- la méthode des itérations inverses,
- la méthode de Lanczos,
- la méthode IRA (dite de 'Sorensen'),
- la méthode de Bathe & Wilson.

On donne au lecteur les propriétés et les limitations, théoriques et pratiques, des méthodes modales abordées tout en reliant ces considérations, qui peuvent parfois paraître un peu «éthérées», à un paramètrage précis des opérateurs. Pour chaque méthode, on récapitule sous forme de tableaux ledit paramètrage avec ses valeurs par défaut et des références aux paragraphes du document.

Table des matières

1 Introduction - Description du document

Une majorité d'étude concernant le **comportement dynamique de structures** sont réalisées en effectuant une **analyse transitoire sur base modale**. Pour exhumer ces modes de vibrations, une kyrielle d'algorithmes ont été développés depuis une cinquantaine 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 du *Code_Aster*.

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 a fait indiqué. Il regroupe des algorithmes heuristiques et ceux de type puissances (cf. [§3]).
Par contre, pour **capturer une partie significative du spectre,** on a recourt à **MODE_ITER_SIMULT**. Ce dernier fédère les méthodes dites de «sous-espace» (Lanczos [§4], [§5], IRAM [§6], Bathe & Wilson [§7]) qui projettent l'opérateur de travail afin d'obtenir un spectre approximé de taille plus réduite (traité alors par une méthode globale de type **QR** ou Jacobi).

Jusqu'à présent, ces algorithmes achoppaient régulièrement sur les mêmes écueils: la détection correcte de modes multiples, de modes de corps rigide et de manière générale, le traitement de spectre tassé. Tout ceci conduisait à l'apparition de **modes «fantômes»** parfois mal aisés à détecter (modes correspondants à des multiplicités ratées et pouvant générer des résidus corrects au sens d'Aster, et ce, d'autant plus que les critères des vérifications post-modales étaient parfois permissifs (résidu en 10^{-2} au lieu du 10^{-6} actuel), désactivés voire insuffisants (test de Sturm limité aux valeurs propres positives) qui suscitent des distorsions de résultats en aval du calcul, lors des projections sur base modale.

Pour s'affranchir des problèmes récurrents à ce type d'approche, on a donc proposé d'enrichir MODE_ITER_SIMULT (à partir de la V5) de **l'algorithme IRA** ('Implicit Restarted Arnoldi' [§6]). Cette variante d'Arnoldi, initiée par D.C. Sorensen en 1992, connaît un réel essor pour la résolution de grands systèmes modaux sur des super-calculateurs parallèles.
Elle tente d'apporter un remède élégant aux problèmes numériques récurrents soulevés par les autres approches. En bref, IRAM procure une meilleure robustesse globale Dans la V5, il a permis de solder toutes les d'anomalies logicielles liées aux problèmes modaux généralisés (dans la V5, il a permis de solder toutes les d'anomalies logicielles liées aux problèmes modaux généralisés) tout en améliorant les complexités calcul et mémoire pour une précision fixée, et, elle permet un réel contrôle sur la qualité des modes via un paramètrage idoine. **Son utilisation (méthode par défaut dans MODE_ITER_SIMULT) est donc à conseiller dans tous les cas de figures**.

Ce document s'articule autour des parties suivantes :

- dans un premier temps, on rappelle le contexte du calcul modal dans le *Code_Aster* au travers des matrices et des conditions limites utilisées, des propriétés particulières de modes propres exhumés et des difficultés d'estimation du spectre. On récapitule aussi ce qu'il faut savoir à minima sur les transformations spectrales, sur les familles d'algorithmes modaux et sur les grandes lignes du déroulement d'un calcul modal dans le code,
- dans un second temps, on décrit la méthode des puissances inverses et ses phases préliminaires de localisations de valeurs propres (méthode de bissection et de la sécante) mises en place dans MODE_ITER_INV,
- la troisième partie traite le cadre général des méthodes de sous-espace mises en place dans MODE_ITER_SIMULT. On détaille notamment les implications du choix du shift et du type de sous-espace de projection dans le paramètrage de l'opérateur,
- la trois chapitres suivants reprennent, dans l'ordre, les trois méthodes de cet opérateur: Lanczos, IRAM et Bathe & Wilson,
- dans les annexes on aborde plus en détails des processus «de second niveau» auxquels font appel les trois méthodes précédentes. Il s'agit des algorithmes **QR** et Jacobi qui sont dits généralistes car ils permettent de capturer tout le spectre d'un opérateur. Le premier, l'algorithme **QR**, est fondamental car il intervient dans la plupart des méthodes. C'est l'algorithme de référence qui offre une très grande robustesse mais des complexités calcul et mémoire prohibitives. On aborde aussi les différents algorithmes d'orthonormalisation mis en

place dans le code. En effet, on ne cessera de marteler, tout au long de ce document, que leur qualité et leur robustesse sont cruciales pour le bon déroulement des algorithmes.

Cette dernière version du document a été presque entièrement réécrite en s'inspirant des indices précédents rédigés, respectivement par D. Seligmann [R5.01.01] indice A et B. Quinnez [R5.01.01] indice B, ceci afin d'essayer de se rapprocher du code tout en explicitant plus en détails les caractéristiques des méthodes et les phénomènes numériques sous-jacents. Un effort particulier a été apporté pour mettre en perspective les choix conduits dans le *Code_Aster* par rapport à la recherche, passée et actuelle, ainsi que pour expliciter la philosophie générale d'un calcul modal.

On donne au lecteur les propriétés et les limitations, théoriques et pratiques, des méthodes modales abordées tout en reliant ces considérations, qui peuvent parfois paraître un peu «éthérées», à un paramètrage précis des opérateurs. Pour chaque méthode, on récapitule sous forme de tableaux ledit paramètrage avec ses valeurs par défaut et des références aux paragraphes du document.
Lors des premiers passages, on engage fortement l'utilisateur a ne modifier que les paramètres principaux notés en gras dans ces tableaux et à lire les traces du fichier message. Les autres, concernant plus les arcanes des algorithmes, ont été initialisés empiriquement à des valeurs standards.

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 [bib3], G.H. Golub [bib6], P. Lascaux [bib11], B.N. Parlett [bib18] et Y. Saad [bib32], et on recommande tout particulièrement la synthèse actualisée et exhaustive commise par J.L. Vaudescal [bib23].

2 Contexte

2.1 Problématique

Nous considérons le problème généralisé aux valeurs propres :
Trouver (λ,$\mathbf{u}$) tels que $\mathbf{A}\,\mathbf{u} = \lambda\,\mathbf{B}\,\mathbf{u}$, $\mathbf{u} \neq 0$, où $\mathbf{A}$ et $\mathbf{B}$ sont des matrices symétriques à coefficients réels. 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, la matrice $\mathbf{A}$ est la matrice de rigidité, notée $\mathbf{K}$, (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 $\mathbf{M}$. Les valeurs propres obtenues sont les carrés des pulsations associées aux fréquences cherchées.

  Le système à résoudre peut s'écrire: $\left(\mathbf{K}+\mathbf{K}_g\right)\mathbf{u} = \omega^2\ \mathbf{M}\ \mathbf{u}$ où $\omega = 2\pi\ f$ est la pulsation, f la fréquence propre et $\mathbf{u}$ le vecteur de déplacement propre associé.

- **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 problème généralisé aux valeurs propres de la forme: $\left(\mathbf{K}+\lambda\,\mathbf{K}_g\right)\mathbf{u} = \mathbf{0}$ avec $\mathbf{K}$ matrice de rigidité et $\mathbf{K}_g$ matrice de rigidité géométrique.

Remarques :

- *ce type de problème modal généralisé est traité dans le Code_Aster par deux opérateurs: MODE_ITER_INV et MODE_ITER_SIMULT. Chacun ayant son périmètre d'application, ses fonctionnalités et ses limitations,*
- *dans la V5, 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 fréquences alors que dans le second, on parlera de charge critique,*
- *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:* $\left(\mathbf{K}+i\,\omega\,\mathbf{C}-\omega^2\ \mathbf{M}\right)\mathbf{u} = \mathbf{0}$. *Il est résolu par les deux opérateurs modaux et fait l'objet d'une note spécifique [R05.01.02].*

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.

2.2 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 ddls de Lagrange [R3.03.01], permet de vérifier
  $\mathbf{C}\,\mathbf{u} = \mathbf{0}$ (CLL pour Condition Limite Linéaire),
  avec $\mathbf{C}$ matrice réelle de taille p x n ($\mathbf{K}$ et $\mathbf{M}$ sont d'ordre n). Les matrices de rigidité et de masse dualisées ont alors la forme
  $$\tilde{\mathbf{K}} = \begin{pmatrix} \mathbf{K} & \beta\mathbf{C}^T & \beta\mathbf{C}^T \\ \beta\mathbf{C} & -\alpha\,\mathbf{Id} & \alpha\,\mathbf{Id} \\ \beta\mathbf{C} & \alpha\,\mathbf{Id} & -\alpha\,\mathbf{Id} \end{pmatrix} \qquad \tilde{\mathbf{M}} = \begin{pmatrix} \mathbf{M} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}$$
  avec α et β réels strictement positifs qui seront initialisés à 1 sans que cela entraîne une perte de généralité.
  La dimension du problème a été augmentée de $2p$, car aux n ddls dits "physiques", on a rajouté des ddls de Lagrange. Il y a deux ddls de Lagrange 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 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 :
  $$\tilde{\mathbf{K}} = \begin{pmatrix} \overline{\mathbf{K}} & \mathbf{0} \\ \mathbf{0} & \mathbf{Id} \end{pmatrix} \qquad \tilde{\mathbf{M}} = \begin{pmatrix} \overline{\mathbf{M}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}$$
  La dimension du problème reste inchangée mais il faut cependant retirer les participations des ddls bloqués aux composantes des matrices initiales ($\overline{\mathbf{K}}$ est obtenue à partir de $\mathbf{K}$ en éliminant les lignes et les colonnes des ddls qui sont bloqués; idem pour $\overline{\mathbf{M}}$).

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

- $n_{ddl-actifs} = n - \dfrac{3p'}{2} \quad avec\ p' = 2p$ (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 : Gestion des ddls

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} ddl n'est pas bloqué $\mathbf{u}_{bloq}(i) = 1$,
- sinon $\mathbf{u}_{bloq}(i) = 0$,

$$\mathbf{u}^1(i) = \mathbf{u}^0(i) \,.\, \mathbf{u}_{bloq}(i) \quad (i = 1...n) \Rightarrow \mathbf{u}^1$$

Si on utilise la méthode de double dualisation, on a besoin d'un vecteur de positionnement des ddls 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 :

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

Par la suite, pour simplifier les notations, nous ne ferons 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.

2.3 Propriétés des matrices

Comme nous l'avons écrit précédemment les matrices considérées sont **symétriques** et à **coefficients réels**. Suivant les cas de figure répertoriés dans le tableau ci-dessous, elles peuvent être définies positives (noté > 0), semi-définies positives (≥ 0), indéfinies (≤ 0 ou ≥ 0) voire singulières (S).

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

Tableau 2.3-a : Propriétés des matrices du problème généralisé

***** Cette colonne concerne bien sûr les propriétés des matrices dualisées constituées à partir des matrices initiales [R3.03.01]

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 $\tilde{\mathbf{K}}_g$) devenir potentiellement indéfinies.

Remarque :

Cet éventail de propriétés doit être pris en compte lors du choix du couple opérateur de travail - (pseudo) 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 le Code_Aster.

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

2.4 Propriétés des modes propres

Rappelons tout d'abord que si la matrice du problème modal standard $\mathbf{A}\,\mathbf{u} = \lambda\ \mathbf{u}$ 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 problème généralisé $\mathbf{A}\,\mathbf{u} = \lambda\,\mathbf{B}\,\mathbf{u}$, 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 $\lambda_{\pm} = \frac{1}{2}\left(1 \pm i\,\sqrt{3}\right)$ 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 $\alpha \in \Re$ et $\beta \in \Re$ 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 $\left(\alpha\,\mathbf{A} + \beta\,\mathbf{B}\right)\mathbf{u} = \left(\lambda\,\alpha + \beta\right)\mathbf{B}\,\mathbf{u}$. Comme $\left(\alpha\,\mathbf{A} + \beta\,\mathbf{B}\right)$ est définie positive, elle admet une décomposition de Cholesky unique sous la forme $\mathbf{C}\mathbf{C}^T$ avec $\mathbf{C}$ matrice régulière.

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

❑

Remarque

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.*

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

$$\begin{cases} \mathbf{u}_i^T \mathbf{B}\, \mathbf{u}_j = \delta_{ij}\ a_j \\ \mathbf{u}_i^T \mathbf{A}\, \mathbf{u}_j = \lambda_j\ \delta_{ij}\ a_j \end{cases}$$

où a_j est un scalaire dépendant de la norme du j$^{\text{ième}}$ 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. [bib9]).

❑

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),*
- *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 [bib25]),*
- *cette propriété simplifie les **calculs de recombinaisons modales** (*DYNA_TRAN_MODAL *[R5.06.01]), lorsqu'on manipule des matrices de rigidité et de masse généralisées qui sont diagonales. Les quantités $k_j = \lambda_j a_j$ et $m_j = a_j$ sont appelées, respectivement, rigidité modale et masse modale du j$^{\text{ième}}$ mode.*

Sachant que les modes sont réels, nous allons maintenant nous préoccuper de leur estimation.

2.5 Estimation du spectre réel

Du fait de l'appartenance du spectre à l'axe réel, les problèmes de **comptage de valeurs propres** vont se trouver grandement **simplifiés**. On n'a pas à traiter de régionnement du plan complexe et l'on peut s'appuyer sur le corollaire de la loi d'inertie de Sylvester suivant.

Corollaire 3

Soient $\mathbf{A}$ et $\mathbf{B}$ deux matrices réelles symétriques, $\mathbf{B}$ étant de plus définie positive. Le nombre de valeurs propres, strictement inférieure à σ, du problème généralisé $\mathbf{A}\,\mathbf{u} = \lambda\,\mathbf{B}\,\mathbf{u}$ est alors égal au nombre de coefficients diagonaux strictement négatifs de la matrice $\mathbf{D}$ telle que $\left(\mathbf{A} - \sigma\,\mathbf{B}\right) = \mathbf{L}\mathbf{D}\mathbf{L}^T$.

Preuve :

Cf. paragraphe n°1 de l'article de Y. Haugazeau [bib7].

❑

Remarques

- *ce corollaire s'étend aux matrices hermitiennes et résulte des propriétés des suites de Sturm et du principe d'inclusion (*on note $p^{(n)}(\xi) = dét\left(\mathbf{A}^{(n)} - \xi\, \mathbf{B}^{(n)}\right)$ le polynôme caractéristique du problème généralisé $\mathbf{A}^{(n)}\ \mathbf{u}^{(n)} = \lambda^{(n)}\ \mathbf{B}^{(n)}\ \mathbf{u}^{(n)}$ obtenu en supprimant les *n* dernières lignes et colonnes des matrices $\mathbf{A}$ et $\mathbf{B}$. La suite de polynômes $\left(p^{(n)}\right)_n$ constitue une suite de Sturm car les racines du i^{ème} polynôme encadrent celles du (i+1)^{ème}. Cette propriété d'entrelacement du spectre de sous-matrices réelles symétriques est appelée «principe d'inclusion») *[bib7], [bib9],*
- *les valeurs propres multiples éventuelles sont comptées avec leur multiplicité,*
- *par la suite, on appellera position modale de σ et on la notera $pm(\sigma)$, ce nombre de coefficients diagonaux strictement négatifs.*

Ce corollaire permet donc de déterminer facilement le nombre de valeurs propres contenues dans un intervalle $\left[\sigma\ ,\ \mu\right]$ et la position modale de ces valeurs propres dans le spectre. Il suffit d'effectuer deux décompositions $\mathbf{LDL}^T$, celle de $\left(\mathbf{A} - \sigma\ \mathbf{B}\right)$ et celle de $\left(\mathbf{A} - \mu\ \mathbf{B}\right)$ et de comptabiliser la différence du nombre de termes strictement négatifs entre les deux matrices diagonales. Dans le jargon du code, on désigne (improprement d'ailleurs !) ce test sous le vocable de **"test de Sturm"**.

Il doit cependant être étendu aux formes bien particulières des matrices rencontrées dans le *Code_Aster*, et notamment, il faut pouvoir prendre en compte le flambement avec ou sans Lagrange. Pour ce faire, on a **élargi le critère à une matrice $\mathbf{B}$ quelconque** et on l'a **généralisé** en prenant en compte les **matrices dualisées**.
Rappelons que l'on définit habituellement la **signature** d'une matrice (et celle de la forme quadratique associée) comme le triplet d'entiers naturels *(r,s,t)* où *r* désigne le nombre de valeurs propres > 0, *s* le nombre de valeurs propres nulles et *t* celles < 0. Ce dernier entier est donc ce que l'on note $\mathrm{pm}(\sigma)$ dans notre cas de figure.

Propriété 4

Soient $\mathbf{A}$ et $\mathbf{B}$ les deux matrices réelles symétriques (d'ordre *n*) liées au problème modal généralisé (S): $\mathbf{A}\ \mathbf{u} = \lambda\ \mathbf{B}\ \mathbf{u}$. Notons $\tilde{\mathbf{A}}$ et $\tilde{\mathbf{B}}$, leurs matrices associées résultant de la double dualisation de p Lagranges permettant de vérifier $\mathbf{C}\ \mathbf{u} = \mathbf{0}$ (CLL), avec $\mathbf{C}$ matrice réelle de taille p x n.
Alors, $\forall\ \sigma \in \Re$, la signature de $\left(\tilde{\mathbf{A}} - \sigma\ \tilde{\mathbf{B}}\right)$ s'écrit, en notant $\mathrm{card}_\lambda\left[a,b\right]$ le nombre de valeurs propres du problème généralisé incluses dans l'intervalle $\left[a,b\right]$:

si $\mathbf{B}$ est indéfinie et $\mathbf{A}$ est définie positive
alors $\mathrm{card}_\lambda\{0\} = 0$ et

$$\textbf{si } \sigma < 0 \textbf{ :} \begin{cases} r = \mathrm{card}_\lambda\left]-\infty,\sigma\right[+ \mathrm{card}_\lambda\left[0,+\infty\right[\\ s = p + \mathrm{card}_\lambda\{\sigma\} \\ t = \mathrm{pm}(\sigma) = \mathrm{card}_\lambda\left]\sigma,0\right] + p \end{cases} \qquad \textbf{éq 2.5-1}$$

$$\text{si } \sigma = 0 : \begin{cases} r = \text{card}_\lambda \left]-\infty,+\infty\right[\\ s = p \\ t = \text{pm}(\sigma) = p \end{cases} \qquad \text{éq 2.5-2}$$

$$\text{si } \sigma > 0 : \begin{cases} r = \text{card}_\lambda \left]-\infty,0\right] + \text{card}_\lambda \left]\sigma,+\infty\right[\\ s = \text{card}_\lambda \{\sigma\} + p \\ t = \text{pm}(\sigma) = \text{card}_\lambda [0,\sigma[+ p \end{cases} \qquad \text{éq 2.5-3}$$

si $\mathbf{B}$ est définie positive et si $\mathbf{A}$ est semi-définie positive
alors $\text{card}_\lambda \left]-\infty,0\right[= 0$ et

$$\text{si } \sigma = 0 : \begin{cases} r = \text{card}_\lambda \left]0,+\infty\right[\\ s = p + \text{card}_\lambda \{0\} \\ t = \text{pm}(\sigma) = p \end{cases} \qquad \text{éq 2.5-4}$$

$$\text{si } \sigma > 0 : \begin{cases} r = \text{card}_\lambda \left]\sigma,+\infty\right[\\ s = p + \text{card}_\lambda \{\sigma\} \\ t = \text{pm}(\sigma) = \text{card}_\lambda [0,\sigma[+ p \end{cases} \qquad \text{éq 2.5-5}$$

Preuve :

Pour imposer des conditions aux limites linéaires, on utilise une technique de double dualisation [R3.03.01] qui conduit au système généralisé transformé

$$\left(\tilde{\mathbf{A}} - \sigma \tilde{\mathbf{B}}\right)\tilde{\mathbf{u}} = \left(\begin{pmatrix} \mathbf{A} & \mathbf{C}^T & \mathbf{C}^T \\ \mathbf{C} & -\mathbf{Id} & \mathbf{Id} \\ \mathbf{C} & \mathbf{Id} & -\mathbf{Id} \end{pmatrix} - \sigma \begin{pmatrix} \mathbf{B} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}\right)\begin{pmatrix} \mathbf{u} \\ \mathbf{v} \\ \mathbf{w} \end{pmatrix} = \mathbf{0} \quad \left(\tilde{S}\right)$$

On notera par la suite $\mathbf{v}_i^j$ le vecteur colonne nul de taille p (i= 1..p; j= 2,3) sauf à l'indice i, pour lequel il vaut 1, et $\mathbf{0}_n$, le vecteur colonne nul de taille n. Maintenant considérons les trois familles de vecteurs suivantes:

- $n - p$ vecteurs $\mathbf{V}_i^1 = \begin{pmatrix} \mathbf{u}_i^1 \\ \mathbf{v}_i^1 \\ \mathbf{v}_i^1 \end{pmatrix}$ qui sont les vecteurs propres du système $(\tilde{S})$. Clairement, d'après la proposition 2, ils sont $\tilde{\mathbf{A}}$ et $\tilde{\mathbf{B}}$ - orthogonaux (d'après les propriétés de $\mathbf{A}$ et $\mathbf{B}$) et on note leurs valeurs propres λ_i.
- p vecteurs indépendants $\mathbf{V}_i^2 = \begin{pmatrix} \mathbf{0}_n \\ \mathbf{v}_i^2 \\ \mathbf{v}_i^2 \end{pmatrix}$,
- p vecteurs indépendants $\mathbf{V}_i^3 = \begin{pmatrix} \mathbf{0}_n \\ \mathbf{v}_i^3 \\ -\mathbf{v}_i^3 \end{pmatrix}$.

Ces $n+p$ vecteurs forment une base *B* de l'espace $E=\left\{\tilde{\mathbf{u}}=\begin{pmatrix}\mathbf{u}\\ \mathbf{v}\\ \mathbf{w}\end{pmatrix} / \quad \mathbf{C}\mathbf{u}=\mathbf{0}\right\}$ des solutions admissibles du problème dualisé (en tant que famille libre d'un espace de dimension finie).

Considérons, pour un nombre réel σ donné, la forme quadratique associée à la matrice $\left(\tilde{\mathbf{A}}-\sigma\,\tilde{\mathbf{B}}\right)$

$$\Phi_\sigma(\tilde{\mathbf{u}})=\left\langle\left(\tilde{\mathbf{A}}-\sigma\,\tilde{\mathbf{B}}\right)\tilde{\mathbf{u}},\tilde{\mathbf{u}}\right\rangle \quad ,\tilde{\mathbf{u}}\in E$$

En décomposant sur la base *B* engendrée par les vecteurs précédents

$$\tilde{\mathbf{u}}=\sum_{i=1,n-p} a_i^1\,\mathbf{V}_i^1 \;+\sum_{i=1,p} a_i^2\,\mathbf{V}_i^2 \;+\sum_{i=1,p} a_i^3\,\mathbf{V}_i^3 \;,$$

on obtient

$$\Phi_\sigma(\tilde{\mathbf{u}})=\sum_{i=1,n-p}\left(a_i^1\right)^2\left(\lambda_i-\sigma\right)\left(\mathbf{u}_i^{1^T}\mathbf{B}\,\mathbf{u}_i^1\right)+\sum_{i=1,p}(-4)\left(a_i^3\right)^2$$

(on a utilisé en particulier les propriétés d'orthogonalité des familles de vecteurs $\mathbf{V}_i^j$ et la relation $\left(\mathbf{u}_i^1\,,\mathbf{C}^T\mathbf{v}_i^2\right)=\left(\mathbf{C}\,\mathbf{u}_i^1\,,\mathbf{v}_i^2\right)=0$). D'où, en notant L_i la forme linéaire qui associe à $\tilde{\mathbf{u}}$ sa $i^{\text{ème}}$ coordonnée dans la base B

$$\Phi_\sigma(\tilde{\mathbf{u}})=\sum_{i=1,n-p}\mathrm{L}_i^2(\tilde{\mathbf{u}})\left(\lambda_i-\sigma\right)\left(\mathbf{u}_i^{1^T}\mathbf{B}\,\mathbf{u}_i^1\right)+\sum_{i=1,p}0\;\mathrm{L}_{n-p+i}^2(\tilde{\mathbf{u}})\;+\sum_{i=1,p}(-4)\;\mathrm{L}_{n+i}^2(\tilde{\mathbf{u}})$$

Clairement les $\left(\mathrm{L}_i\right)_{i=1,n+p}$ sont linéairement indépendantes d'où une lecture immédiate des termes de la signature suivant le signe des facteurs. D'ailleurs, on remarque que cette décomposition comporte obligatoirement *p* zéros et *p* signes moins. Les relations de la propriété se déduisent alors tout naturellement, en utilisant le principe d'inertie de Sylvester (qui nous assure que cette décomposition est invariante), les relations de $\mathbf{A}$ et $\mathbf{B}$ - orthogonalité de la propriété 2, les propriétés du tableau [Tableau 2.3-a] et en remarquant que :

$$\tilde{\mathbf{u}}_i^T\,\tilde{\mathbf{B}}\,\tilde{\mathbf{u}}_j=\mathbf{u}_i^T\,\mathbf{B}\,\mathbf{u}_j=\frac{\tilde{\mathbf{u}}_i^T\,\tilde{\mathbf{A}}\,\tilde{\mathbf{u}}_j}{\lambda_j}=\frac{\mathbf{u}_i^T\,\mathbf{A}\,\mathbf{u}_j}{\lambda_j}\quad \text{pour } \lambda_j\neq 0\ .$$

La restriction $\sigma\geq 0$ du cas de figure (2) provient du fait que si $\mathbf{B}$ est définie positive alors le spectre du problème (S) est positif et donc le shift avec une valeur strictement négative n'a aucun intérêt (on se retrouve dans le cadre d'application du corollaire 3).

❑

D'après le tableau [Tableau 2.3-a] cette propriété s'applique aux matrices manipulées par le *Code_Aster* et on peut construire le corollaire suivant.

Corollaire 5 (Sturm étendu théorique)

Dans les configurations matricielles du *Code_Aster*, le nombre de modes propres du problème généralisé (S) dont le vecteur propre vérifie les conditions limites linéaires (CLL) et dont la valeur propre est contenue dans l'intervalle $]\sigma,\mu[$ est

Si $\sigma\ \mu \geq 0$ $\quad \mathrm{card}_\lambda\,]\sigma,\mu[= \mathrm{pm}(\mu) - \mathrm{pm}(\sigma)$,

sinon $\quad \mathrm{card}_\lambda\,]\sigma,\mu[= \mathrm{pm}(\mu) + \mathrm{pm}(\sigma) - 2p$.

Si aucune dualisation de Lagranges n'est requise, on pose p= 0.

Preuve :

Elle est immédiate en combinant les résultats du tableau à ceux de la propriété 4, et, en remarquant que les configurations modales généralisées du *Code_Aster* ne peuvent être que de deux types :

- flambement ⇨ $\mathbf{A}$ est définie positive et $\mathbf{B}$ est indéfinie, le spectre peut être négatif et on utilise les relations de type (1),
- dynamique ⇨ $\mathbf{A}$ est semi-définie positive et $\mathbf{B}$ est définie positive, le spectre est positif et on peux utiliser indifféremment les relations de type (1) ou (2).

Bien sûr, si aucune dualisation de Lagranges n'est requise, le même raisonnement s'applique en posant p = 0, $\tilde{\mathbf{A}} = \mathbf{A}, \tilde{\mathbf{B}} = \mathbf{B}$ et $\tilde{\mathbf{u}} = \mathbf{u}$.
Pour finir, le principe d'inertie de Sylvester nous assure que la signature de la factorisation $\left(\tilde{\mathbf{A}} - \sigma\,\tilde{\mathbf{B}}\right) = \tilde{\mathbf{L}}\tilde{\mathbf{D}}\tilde{\mathbf{L}}^T$ est identique à celle de la matrice shiftée. Malgré cette transformation, les positions modales demeurent donc bien des informations pérennes.

❑

Cependant ce corollaire est tributaire de l'arithmétique exacte, en pratique il faut l'adapter et contrôler son application.

2.6 Implantation du test de Sturm

Ce test est confronté en arithmétique finie à deux problèmes concomitants :

- la factorisation de la matrice shiftée lorsque σ est très proche d'une valeur propre,
- le décompte des termes strictement négatifs de $\tilde{\mathbf{D}}$ pour évaluer la position modale $\mathrm{pm}(\sigma)$.

Le premier point nécessite, au préalable, la détermination d'un critère d'appartenance du shift au spectre du problème. Dans le *Code_Aster* celui ci est fondé sur la perte de décimales lors de la factorisation de la matrice $\left(\tilde{\mathbf{A}} - \sigma\,\tilde{\mathbf{B}}\right) = \tilde{\mathbf{L}}\tilde{\mathbf{D}}\tilde{\mathbf{L}}^T$. Plus précisément, σ est considéré comme étant une valeur propre, si lors de la factorisation on perd plus de NPREC_SOLVEUR décimales (en toute rigueur ce n'est qu'un test de mauvais conditionnement numérique). Il faut alors modifier la valeur de σ en décalant ce shift de PREC_SHIFT % suivant l'algorithme:

Pour i = 1, *NMAX_ITER_SHIFT*
  Si perte de plus de *NPREC_SOLVEUR* décimales
    alors $\sigma \Leftarrow \sigma\left(1 + PREC\_SHIFT\right)$,
  Sinon exit;
Fin boucle.

Algorithme 1 : Décalage du shift

Si au bout de NMAX_ITER_SHIFT tentatives, la matrice n'est toujours pas numériquement inversible, on continue tout de même le calcul avec la dernière valeur shiftée. L'encadré ci-dessous montre la trace d'un tel décalage dans le fichier message.

```
ON MODIFIE LA VALEUR DE DECALAGE DE:  5.00000E+00POURCENT
LA VALEUR DE DECALAGE DEVIENT:  5.96840E+04
ON MODIFIE LA VALEUR DE DECALAGE DE:  5.00000E+00POURCENT
LA VALEUR DE DECALAGE DEVIENT:  5.66998E+04

VALEUR_MIN EN FREQUENCE EST :    2.00000E-01
VALEUR_MAX EN FREQUENCE EST :    5.78813E+01
LA VALEUR DE DECALAGE EN FREQUENCE EST :  3.78975E+01
```

Exemple 2 : Décalage du shift

Dans l'algorithme 1 si $|\sigma| \leq f_{corig}$ alors $\sigma = -f_{corig}$. Ce paramètre f_{corig} correspond à une valeur seuil en dessous de laquelle on considère qu'on a une valeur propre numériquement nulle (en mécanique classique, cela correspond à un mode de corps rigide, cf. [§5.5.4]). L'imposition $\sigma = -f_{corig}$ permet ainsi de dissocier ces modes du reste du spectre et d'éviter des instabilités numériques lors du test de Sturm. Ce seuil est paramètrable avec le mot-clé SEUIL_FREQ.

Remarques :

- *les mot-clés précédents sont accessibles à partir du mot-clé facteur CALC_FREQ des deux opérateurs modaux,*
- *Y. Haugazeau [bib7] propose de traiter plus généralement ces problèmes de pivots numériquement "petits" en construisant une matrice, unitairement semblable (c'est-à-dire semblable via une matrice de passage unitaire (orthogonale en réel) à la matrice shiftée, dont la factorisation présenterait moins d'instabilité.*

En prenant en compte ces éléments et sachant que, numériquement, le décompte des pivots strictement négatifs de la matrice diagonale englobe en fait aussi les éléments (théoriquement) nuls de la signature, on peut réécrire le corollaire précédent.

Corollaire 5bis (Sturm étendu numérique)

Suivant les hypothèses du corollaire 5, on a le critère numérique de comptabilité des modes suivant :

Si $\sigma\ \mu > 0$ $\quad \mathrm{card}_\lambda[\sigma,\mu] = \mathrm{pm}(\mu) - \mathrm{pm}(\sigma)$,

si $\sigma\ \mu < 0$ $\quad \mathrm{card}_\lambda[\sigma,\mu] = \mathrm{pm}(\mu) + \mathrm{pm}(\sigma) - 4p$

Preuve (heuristique) :

On applique le fait que numériquement l'opérateur de factorisation fournit la "position modale numérique" $\mathrm{pm}(\sigma) = s + t$ à la propriété 4 et au corollaire 5. D'autre part, l'implantation du critère ne permet ni la nullité du produit, ni l'estimation de $\mathrm{card}_\lambda\{\sigma\}$.

❑

Remarque :

Ce critère est utilisé en pré-traitements dans MODE_ITER_INV (options 'AJUSTE' ou 'SEPARE') et dans MODE_ITER_SIMULT (option 'BANDE'), en post-traitements pour vérifier la validité du nombre de modes (MODE_ITER_SIMULT) et dans des commandes auxiliaires (IMPR_STURM).

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

2.7 Transformation spectrale

Ces techniques permettent de répondre à un triple objectif :

- exhumer un problème modal standard,
- 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 :

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

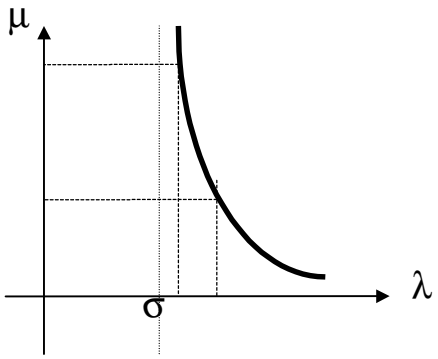


Figure 2.7-a : Effet du "shift and invert" sur la séparation des valeurs propres.

La figure [Figure 2.7-a] 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. [§5.5.1]).*

Pour mémoire, notons qu'avec un shift complexe plusieurs scénarios sont envisageables :

- travailler complètement en arithmétique complexe,
- en arithmétique réelle,
- mixer les deux démarches en isolant les contributions réelles et imaginaires de $\mathbf{A}_\sigma$

$$\tilde{\mathbf{A}}_\sigma = \mathrm{Re}(\mathbf{A}_\sigma) \Rightarrow \tilde{\mu} = \frac{1}{2}\left(\frac{1}{\lambda-\sigma} + \frac{1}{\lambda-\bar{\sigma}}\right),$$

$$\tilde{\tilde{\mathbf{A}}}_\sigma = \mathrm{Im}(\mathbf{A}_\sigma) \Rightarrow \tilde{\tilde{\mu}} = \frac{1}{2i}\left(\frac{1}{\lambda-\sigma} - \frac{1}{\lambda-\bar{\sigma}}\right).$$

Chacune de ses approches a ses avantages et ses inconvénients. Pour les problèmes quadratiques, c'est pour l'instant la deuxième solution qui a été retenue dans le code.

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,*
- *un autre classe de transformation spectrale, avec un double shifts, permet de sélectionner les valeurs propres situées à droite d'un axe vertical. Il s'agit de la transformation rationnelle de Cayley :* $\underbrace{\left(\mathbf{A}-\sigma_1\,\mathbf{B}\right)^{-1}\left(\mathbf{A}-\sigma_2\,\mathbf{B}\right)}_{\mathbf{A}_\sigma}\mathbf{u} = \underbrace{\frac{\lambda-\sigma_2}{\lambda-\sigma_1}}_{\mu}\mathbf{u}$.

Nous allons maintenant énumérer les différentes familles de méthodes permettant de résoudre le problème modal standard.

2.8 Calcul modal

Les méthodes de calcul modal peuvent se regrouper en (au moins) **quatre familles** :

- Les algorithmes de type $\mathbf{QR}$ (cf. [Annexe 1]) qui ont été présentés par H. Rutishauser (1958) et formalisés concurremment par J.C. Francis et V.N. Kublanovskaya (1961). C'est un algorithme fondamental souvent impliqué dans les autres méthodes.

  Périmètre d'utilisation : Calcul de tout le spectre.
  Avantages : Bonne convergence, robustesse, calcul de la forme de Schur (toute matrice complexe $\mathbf{A}$ admet la décomposition de Schur $\mathbf{Q}^*\mathbf{A}\mathbf{Q} = \mathbf{T}$, avec $\mathbf{Q}$ et $\mathbf{T}$ des matrices respectivement, unitaire et triangulaire supérieure. Dans le cas réel, $\mathbf{T}$ n'est que diagonale par bloc 1x1 ou 2x2 (forme de Schur réelle). Les colonnes de la matrice unitaire, dite de Schur, sont appelées vecteurs de Schur).
  Inconvénients : Complexités mémoire et calcul prohibitives, sensibilités au «balancing (les techniques d'équilibrage (balancing en anglais) consistent à transformer réversiblement l'opérateur de travail afin que sa norme matricielle décroisse. Ainsi sa manipulation sera moins sensible aux effets d'arrondi (cf. [§10.3]))».
  Variantes : Avec shift implicite ou explicite, simple ou double ...

- 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. [§3]).

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 ... (ensemble de valeurs propres «très proches»).
Variantes : Puissances inverses, (bi) itération du quotient de Rayleigh (cf. [§3.3.2]) ...

- 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. [§4]).

  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)...

- 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, mais aussi celle de Jacobi, de Laguerre etc...

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 :

- *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,*
- *résolution de systèmes linéaires; La méthode du gradient conjugué est une méthode de sous-espace pour les systèmes symétriques définis positifs,*
- *recherche de racines de polynômes : la méthode des puissances est une méthode de Bernoulli appliquée à la matrice 'compagnon' du polynôme (cf. [bib11] pp502/503).*

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

2.9 Implantation dans le *Code_Aster*

Celle-ci peut se décomposer en quatre phases :

1) 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 paramètré 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 paramètré par NMAX_FREQ.

2) Dans une seconde phase de pré-traitements, on **factorise partiellement la matrice de travail**

$$\mathbf{A}_\sigma = \left(\mathbf{A} - \sigma\,\mathbf{B}\right)^{-1}\mathbf{B}$$

en ne s'intéressant qu'à sa partie inversée. L'action de cet opérateur sur un vecteur quelconque $\mathbf{u}$, notée $\mathbf{u}^2$, se construira ainsi plus efficacement

$$\begin{vmatrix} \mathbf{u}^1 = \mathbf{B}\,\mathbf{u} \\ \left(\mathbf{A} - \sigma\,\mathbf{B}\right)\mathbf{u}^2 = \mathbf{u}^1 \end{vmatrix} \Rightarrow \mathbf{u}^2$$

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 mêmes décalages suivant l'algorithme 1.

Remarque

Dans la V5 lorsque NMAX_ITER_SHIFT est atteint le calcul s'arrête en erreur fatale (uniquement pour ce pré-traitement), ce problème de factorisation pouvant donner lieu à des instabilités numériques.

3) On effectue le **calcul modal** proprement dit: on résout le problème standard, puis on revient au problème initial. Dans MODE_ITER_INV deux variantes sont disponibles: la méthode des itérations inverses (option : **'DIRECT'**) et son accélération par quotient de Rayleigh (**'RAYLEIGH'**). Pour ce qui est de MODE_ITER_SIMULT, il permet l'usage de trois méthodes distinctes: la méthode de Bathe et Wilson (**'JACOBI'**), celle de Lanczos ('TRI_DIAG') et enfin, celle de Sorensen (**'SORENSEN'**).
Chacune de ces méthodes possèdent des tests d'arrêts internes. Sans compter que les méthodes de projection emploient des méthodes modales auxiliaires:Jacobi (cf. [Annexe 3]) pour la première et $\mathbf{QR}$ / $\mathbf{QL}$ (cf. [Annexe 1]) pour les autres. 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.

4) Cette dernière partie regroupe les **tests d'arrêts globaux** qui vérifient le bon déroulement du calcul. Ils sont de deux types :

- **Norme** (on rappelle que $\|\mathbf{u}\|_\infty = \max_i |\mathbf{u}_i|$, $\|\mathbf{A}\|_\infty = \max_i \sum_j |\mathbf{A}_{ij}|$, $\|\mathbf{u}\|_2 = \left(\sum_i |\mathbf{u}_i|^2 \right)^{\frac{1}{2}}$ et $\|\mathbf{A}\|_2 = \max_i |\lambda_i(\mathbf{A}\mathbf{A}^*)|^{\frac{1}{2}}$ $\left(= \max_i |\lambda_i(\mathbf{A})| \text{ si } \mathbf{A} \text{ hermitien} \right)$ **du résidu du problème initial**

$$\mathbf{u} \Leftarrow \frac{\mathbf{u}}{\|\mathbf{u}\|_\infty}$$

Si $|\lambda| > SEUIL\_FREQ$ alors

$$\frac{\|\mathbf{A}\,\mathbf{u} - \lambda\,\mathbf{B}\,\mathbf{u}\|_2}{\|\mathbf{A}\,\mathbf{u}\|_2} \; ? < \; SEUIL,$$

Sinon

$$\|\mathbf{A}\,\mathbf{u} - \lambda\,\mathbf{B}\,\mathbf{u}\|_2 \; ? < \; SEUIL,$$

Fin si.

Algorithme 2 : Test de la norme du résidu

Cette séquence est paramètrée par les mot-clés SEUIL et SEUIL_FREQ, appartenant respectivement au 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 dans MODE_ITER_SIMULT et il permet de vérifier que le nombre de valeurs propres contenues dans une bande test $[\sigma_1, \sigma_2]$ est égal au nombre détecté par l'algorithme. L'inclusion de la bande initiale (en option 'BANDE' il s'agit des valeurs renseignées par l'utilisateur, sinon ce sont les valeurs extrêmes du spectre calculé) $[\sigma_i, \sigma_f]$ dans cette bande test est conduite afin de détecter d'éventuels problèmes de clusters ou de multiplicités aux bornes initiales (cf. TP n°1 [bib25]).

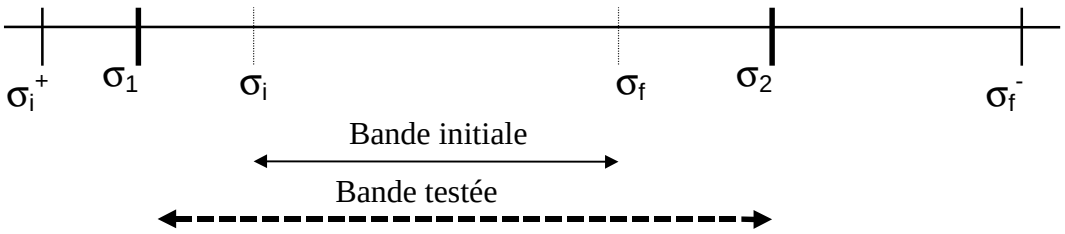


Figure 2.9-a : Comptage des valeurs propres

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 2.9-a]), on a l'algorithme de construction de la bande test suivant :

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

alors

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

Sinon

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

Fin si.

Algorithme 3 : Construction de la bande test

Cette séquence est paramètrée par le mot-clé *PREC_SHIFT* du mot-clé facteur *VERI_MODE*. L'encadré ci-dessous montre 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 : Post-vérifications

L'activation du deuxième post-traitement est subordonnée à l'initialisation de STURM à 'OUI' dans le mot-clé facteur VERI_MODE. Lorsque cette règle est activée et non 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' !

Maintenant que le contexte des problèmes modaux généralisés du *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.

3 Méthode des puissances inverses (MODE_ITER_INV)

3.1 Introduction

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** (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) 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 (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) 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 :

1) 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**),
2) 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. [§3.2.1]),
- 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 (cf. [§3.2.2]).
  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 (cf. [§2.8]), 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,*
- *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 paramètrages) qui sont mis en place dans la première partie du processus.

3.2 Localisation et séparation des valeurs propres

3.2.1 Méthode de bissection

Comme on l'a déjà vu précédemment, le corollaire 5bis de la loi d'Inertie de Sylvester (cf. [§2.5]) permet de déterminer le nombre de valeurs propres contenues dans un intervalle donné en effectuant deux décompositions $\mathbf{LDL}^T$. 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 $\left[\lambda_a, \lambda_b\right]$, on opère donc de la façon suivante, connaissant $\mathrm{pm}\left(\lambda_a\right)$ et $\mathrm{pm}\left(\lambda_b\right)$:

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

Algorithme 4 : 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 $\left(\frac{\left|\lambda_a - \lambda_b\right|}{\lambda^*}\right) \leq PREC\_SEPARE$ (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 couteux 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 $\mathbf{A}, \mathbf{A}^T$ ...). Dans cette optique, la méthode de bissection peut s'avérer plus efficace qu'un $\mathbf{QR}$ en présence de cluster. Sa convergence, bien que linéaire, est en effet majorée par ½ alors que celle de $\mathbf{QR}$ peut tendre vers 1 [bib11].*

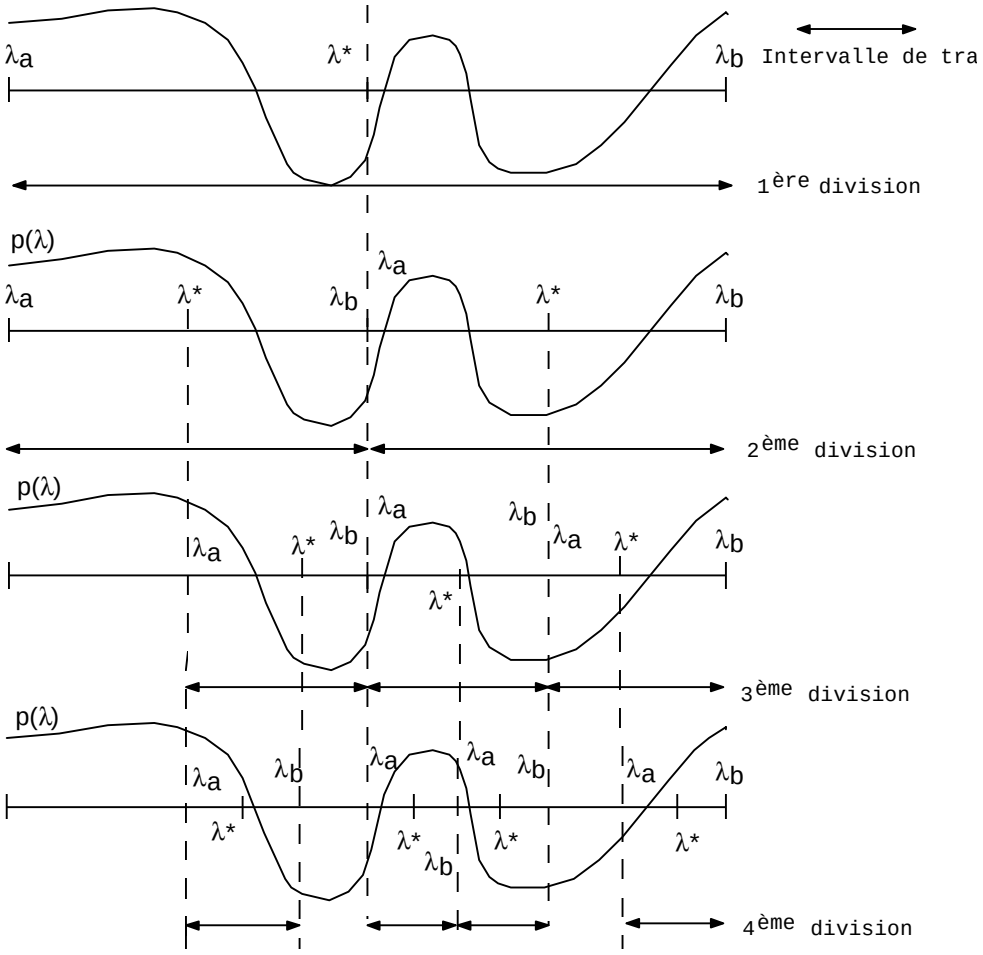


Figure 3.2.1-a : Méthode de bissection

Ces valeurs propres approximées peuvent être améliorées par une minimisation unidimensionnelle cherchant à annuler le polynôme caractéristique $\mathrm{p}(\lambda) = \mathrm{dét}(\mathbf{A} - \lambda\mathbf{B})$. Mais rien que le calcul d'une valeur de ce déterminant requiert N ! opérations.

3.2.2 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 $\mathrm{p}(\lambda)$ 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

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

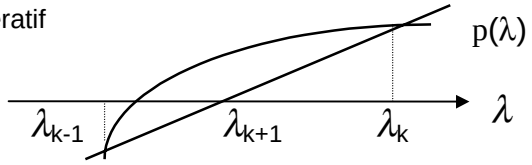


Figure 3.2.2-a : Méthode de la sécante

La tangente étant approximée par une différence finie afin de ne pas avoir à calculer de dérivée de $\mathrm{p}(\lambda)$, seule l'estimation du polynôme est requise.

On considère qu'on a atteint la convergence quand $\frac{|\lambda_{k+1}-\lambda_k|}{\lambda_k} < PREC\_AJUSTE$ et, par ailleurs, on se limite à NMAX_ITER_AJUSTE itérations si ce critère n'est pas atteint.

Remarque :

Cette méthode a une convergence 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.

Nous allons maintenant détailler l'algorithme des puissances inverses (couplé à une accélération de Rayleigh) constituant la seconde partie du processus.

3.3 Méthode des puissances inverses

3.3.1 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** (cette notation ne doit pas être confondue celle symbolisant le «shift and invert » $\mathbf{A}_\sigma = \left(\mathbf{A}-\sigma\,\mathbf{B}\right)^{-1}\mathbf{B}$) $\mathbf{A}^\sigma = \left(\mathbf{A}-\sigma\,\mathbf{B}\right)$ et cela revient à traiter le problème généralisé $\left(\mathbf{A}^\sigma\right)^{-1}\mathbf{B}\mathbf{u} = \underbrace{\frac{1}{\lambda-\sigma}}_{\mu}\ \mathbf{u}$. La méthode des puissances inverses convergeant proritairement 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 approchées $\left(\mathbf{y}_k\right)_k$ 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.

Algorithme 5 : 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 $\frac{\tilde{\mathbf{y}}_k(i)}{\mathbf{y}_{k-1}(i)} \to \frac{1}{\lambda_1-\sigma}\ (i = 1..n)$ (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 [bib6], [bib23],*
- *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. [bib11] 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 de $\mathbf{A}^\sigma$ 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 $k = 1\ldots$ 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 6 : 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 [bib18].

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.

3.3.2 Méthode d'itération du quotient de Rayleigh

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

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

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, $\mathrm{R}(\mathbf{x})$ minimise $\lambda \rightarrow \left\| \left(\mathbf{A} - \lambda\, \mathbf{B} \right) \mathbf{x} \right\|_2$.

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 $\mathrm{R}(\mathbf{x})$ 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 $\mathrm{k} = 1\ldots$ 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 7 : Méthode d'itérations du quotient de Rayleigh avec $\mathbf{B}$-pseudo-norme

Pour le problème modal standard, on peut montrer[bib18] que la convergence de cet algorithme est cubique dans le cas où l'opérateur de travail est normal (une matrice $\mathbf{A}$ est dite normale si $\mathbf{AA^*}$=$\mathbf{A^*A}$. C'est donc le cas des opérateurs hermitiens, antihermitiens ou unitaires) (à 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 $\mathbf{LDL}^{\mathrm{T}}$, ce qui serait très coûteux. D'où l'idée de n'effectuer (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) ce décalage de Rayleigh, que si on est dans un voisinage (notion arbitraire à définir) de la solution.

Remarque

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 $\mathrm{R}_b(\mathbf{x},\mathbf{y}) = \dfrac{\mathbf{y}^T\mathbf{A}\,\mathbf{x}}{\mathbf{y}^T\mathbf{B}\,\mathbf{x}}$ *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 [bib18]).*

3.3.3 Implantation dans le *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 (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 B.N.Parlett [bib18])) (GSM).
- Calcul de $\delta_0 = sign\left(\mathbf{y}_0^T\mathbf{B}\mathbf{y}_0\right)$.
- Pour k=1, NMAX_ITER faire

  Résoudre $\left(\mathbf{A} - \lambda^*\,\mathbf{B}\right)\mathbf{y}_k = \delta_{k-1}\,\mathbf{B}\,\mathbf{y}_{k-1}$.

  $\mathbf{B}$-orthonormalisation (éventuelle) de $\mathbf{y}_k$.

  Calcul de la correction de la valeur propre $c = \dfrac{\mathbf{y}_k^T\mathbf{B}\,\mathbf{y}_{k-1}}{\mathbf{y}_k^T\mathbf{B}\,\mathbf{y}_k}$.

  Si $\left\|\mathbf{y}_k^T\mathbf{B}\,\mathbf{y}_{k-1}\right| - 1\right| \le PREC$ 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 8 : 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.*

3.3.4 Affichage dans le fichier message

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

Position modale	Fréquence	Amortis-sement	Dichotomie	Méthode	Sécante	Méthode	Inverse
			Nombre d'itérations	Nombre d'itérations	Précision	Nombre d'itérations	Précision
*	*	0.	*	*	*	*	*

Tableau 3.3.4-a : Trace de MODE_ITER_INV dans le fichier message

```
------------------------------------------------------------------------
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 : MODE_ITER_INV

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.

Récapitulons maintenant le paramètrage de l'opérateur MODE_ITER_INV.

3.3.5 Récapitulatif du paramètrage

Mot-clé facteur	Mot-clé		Valeur par défaut	Références
	TYPE_RESU	'DYNAMIQUE'	'DYNAMIQUE'	[§2.1]
		'MODE_FLAMB'		[§2.1]
CALC_FREQ	**FREQ**			[§3.1]
	CHAR_CRIT			[§3.1]
	OPTION	'SEPARE'	'AJUSTE'	[§2.1]
		'AJUSTE'		[§3.1]
		'PROCHE'		[§3.1]
	NMAX_FREQ		0	[§3.1]
	NMAX_ITER_SEPARE		30	[§3.2.1]
	PREC_SEPARE		1.E-04	[§3.2.1]
	NMAX_ITER_AJUSTE		15	[§3.2.2]
	PREC_AJUSTE		1.E-04	[§3.2.2]
	NPREC_SOLVEUR		8	[§2.6]
	NMAX_ITER_SHIFT		5	[§2.6]
	PREC_SHIFT		0.05	[§2.6]
	SEUIL_FREQ		1.E-02	[§2.9]
CALC_MODE	**OPTION**	'DIRECT'	'DIRECT'	[§3.3.3]
		'RAYLEIGH'		[§3.3.3]
	PREC		1.E-05	[§3.3.3]
	NMAX_ITER		30	[§3.3.3]
VERI_MODE	STOP_ERREUR	'OUI'	'OUI'	[§2.9]
		'NON'		
	SEUIL		1.E-02	[§2.9]

Tableau 3.3.5-a : Récapitulatif du paramètrage de MODE_ITER_INV

Remarques :

- *dans la V5, 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,*
- *on retrouve toute la "tripaille" de paramètres liée aux pré-traitements du test de Sturm (NPREC_SOLVEUR, 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.***

4 Méthode de sous-espace (MODE_ITER_SIMULT)

4.1 Introduction

Si on souhaite **seulement calculer les p éléments propres d'un problème généralisé d'ordre *n*** où $n << p$ (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 (dans le cas non hermitien, des projections obliques ont été développées par F. Chatelin [bib2],
- 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,
- et, bien sûr, 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 3ème 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.

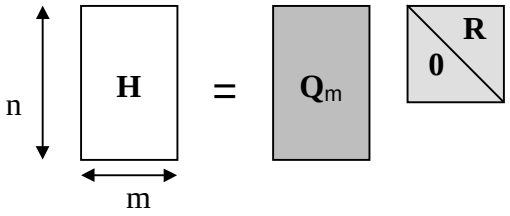
4.2 Analyse de Rayleigh-Ritz

Considérons le problème modal standard d'ordre n, $\mathbf{A}\,\mathbf{u} = \lambda\,\mathbf{u}$, et le sous-espace H de $\Re^n$ 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

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

La recherche des éléments de Ritz $\left(\tilde{\lambda},\tilde{\mathbf{u}}\right)$ 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 :

- 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
- Matrice de Rayleigh



- Eléments propres de $\mathbf{B}_m$: $\left(\tilde{\lambda},\mathbf{x}\right)$
- Eléments de Ritz: $\left(\tilde{\lambda},\tilde{\mathbf{u}} = \mathbf{Q}_m \mathbf{x}\right)$,
- Test d'erreur avec le résidu: $\mathbf{r} = \mathbf{A}\tilde{\mathbf{u}} - \tilde{\lambda}\tilde{\mathbf{u}}$.

Algorithme 9 : Procédure de Rayleigh-Ritz

Remarque :

- *les éléments de Ritz sont en fait les modes propres de la matrice (d'ordre N) de l'approximation de Galerkin $\boldsymbol{C}_m = \boldsymbol{P}_m \boldsymbol{A} \boldsymbol{P}_m$,*
- *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 $\boldsymbol{QR}$ ($O(n^2)$) ou celle d'une itération du quotient de Rayleigh encapsulée dans un processus heuristique (le coût des factorisations est prédominant sur celui des produits matrice-vecteur de l'algorithme proprement dit) tel que celui développé dans* `MODE_ITER_INV` *($>> pn^3$). Mais il est bien clair que ces méthodes sont plus complémentaires que concurrentes.*

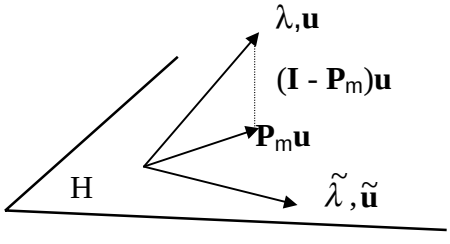
Pour estimer l'erreur commise en utilisant les éléments propres de la matrice $\mathbf{B}_m$ on a le résultat suivant.

Théorème 6

Soit $\left(\lambda,\mathbf{u}\right)$ 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

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

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

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

Preuve :

Cf. [bib11] 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 $\min_{\lambda_i \neq \lambda_j} \left|\lambda_i - \lambda_j\right|$. L'estimation du second membre $\left\|\left(\mathbf{I}-\mathbf{P}_m\right)\mathbf{u}\right\|_2$ dépend du choix du sous-espace.

Remarque :

Le nombre $\min_{\lambda_i \neq \lambda_j} \left|\lambda_i - \lambda_j\right|$ (et ses variantes) est omniprésent dans les analyses d'erreurs, de convergence ou de conditionnement spectrale. 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.

4.3 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. [§7] `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 $\mathbf{QR}$, 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. [§5], `METHODE = 'TRI_DIAG'`) et d'**IRAM** (cf. [§6], `METHODE = 'SORENSEN'`) consiste à partir d'un vecteur initial $\mathbf{y}$, à construire la suite d'espaces de Krylov $(\mathrm{H_i})_{i=1,m}$ où $\mathrm{H_i}$ est l'espace engendré par la famille de vecteurs $\left(\mathbf{y},\mathbf{A}\mathbf{y},...,\mathbf{A}^{i-1}\mathbf{y}\right)$. Ce dernier est appelé l'espace de Krylov de $\mathbf{A}$ et d'ordre i initié par $\mathbf{y}$:

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

  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 (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 exhumés (cf. [§5.4.2]), les techniques d'accélération (technique consistant à remplacer la matrice de travail $\boldsymbol{A}$ par un polynôme matriciel $\boldsymbol{P(A)}$ qui a la particularité d'être de grande amplitude dans les régions spectrales d'intérêt (cf. [§6.4]) 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. [bib11] pp538-45, [bib23] 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 $i^{\text{ème}}$ mode Lorsqu'ils sont rangés classiquement, c'est à dire par ordre décroissant de module (lorsqu'ils sont rangés classiquement, c'est-à-dire par ordre décroissant de module) au bout de k itérations s'écrit :

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

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. [§2.7]).

4.4 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.

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

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).

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

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

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 la commande MODE_ITER_SIMULT, il y a trois 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 $\sigma_0 = \left(2\pi\ f_0\right)^2$, on cherche les fréquences proches de la fréquence FREQ = f_0. (OPTION = 'CENTRE').
- $\sigma = \dfrac{\sigma_0 + \sigma_1}{2}$ avec $\sigma_0 = \left(2\pi\ f_0\right)^2$ et $\sigma_1 = \left(2\pi\ f_1\right)^2$, on cherche toutes les fréquences comprises dans la bande $\left[f_0\ ,\ f_1\right]$ (OPTION = 'BANDE' avec FREQ = {f_0,f_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 5bis (cf. [§2.6]).

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, paramètrables par l'utilisateur, ont été implantés (cf. [§2.6], [§2.9]).

5 Méthode de Lanczos (METHODE = 'TRI_DIAG')

5.1 Introduction

Cet algorithme, mis au point par **Lanczos** [bib10] **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 [bib3] et al. qui lui est entièrement consacré, celui de B.N. Parlett [bib18] et la synthèse actualisée et exhaustive de J.L. Vaudescal [bib23] (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 le *Code_Aster*, nous nous attacherons au cadre réaliste de l'arithmétique finie.

5.2 Algorithme de Lanczos théorique

5.2.1 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 $\mathbf{QR}$ 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 $O(20pm)$ [bib6].
L'algorithme consiste à construire progressivement une famille de vecteurs de Lanczos $\mathbf{q}_1$, $\mathbf{q}_2$,..., $\mathbf{q}_m$ 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.

$\mathbf{q}_0 = \mathbf{0}$, $\beta_0 = 0$.
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 10 : 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 $\mathbf{R}_m = \beta_m \, \mathbf{q}_{m+1} \, \mathbf{e}_m^T$.

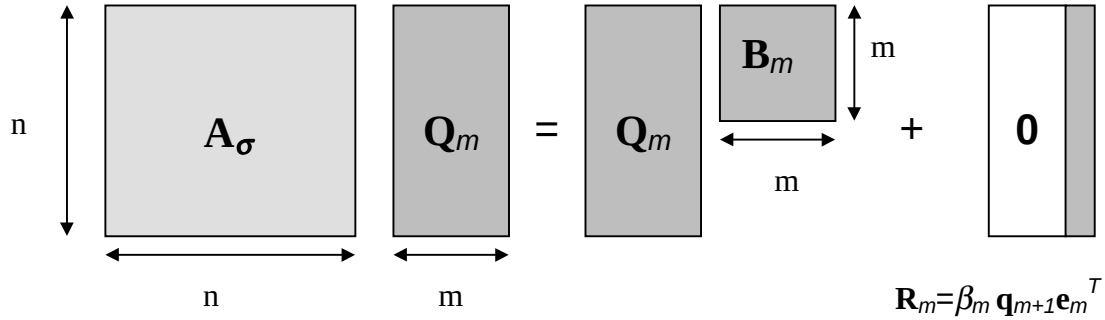


Figure 5.2.1-a : Factorisation de Lanczos

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

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

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 $\mathbf{q}_1$,
  $K_m\left(\mathbf{A}_\sigma, \mathbf{q}_1\right) = Vect\left(\mathbf{q}_1, \mathbf{A}_\sigma \mathbf{q}_1, \ldots, \mathbf{A}_\sigma^{m-1} \mathbf{q}_1\right)$,
- 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 irréductible (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 $\mathbf{A}_{i+1,i} \neq 0 \; \forall i$) (donc à 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 [bib23],*
- *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. [§6.2]).*

5.2.2 Estimations d'erreurs et de convergences

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 7

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

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

Preuve :

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

❑

Remarque :

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

$$\left|\lambda - \tilde{\lambda}\right| \leq \frac{\|\mathbf{r}\|_2^2}{\gamma}$$

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

Donc, plus que la minimisation du résidu, le critère d'arrêt des méthodes de type Lanczos/Arnoldi $\left|\beta_m\right| < \varepsilon$ 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 6 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 8

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

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

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

avec $\gamma_i = 1 + 2\dfrac{\lambda_1 - \lambda_{1+1}}{\lambda_{1+i} - \lambda_n}$, $T_{m\text{-}i}(x)$ le polynôme de Tchebycheff de degré m-i, β la constante du théorème 6 et $\alpha = \dfrac{\beta}{\left\|\mathbf{P}_m \mathbf{u}_i\right\|_2}\tan\left(\mathbf{q}_1, \mathbf{u}_i\right)$.

Preuve :

Cf. [bib11] pp554-557.

❑

Ces majorations indiquent (cf. [bib23] pp59-63) 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 $T_{m\text{-}i}(x)$, 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 [bib20].

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

5.3 Algorithme de Lanczos pratique

5.3.1 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 projeté 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 9 (Analyse de Paige)

En reprenant les notations de l'algorithme 10 et en notant ε la précision machine, à l'itération $k+1$ l'orthogonalité du nouveau vecteur de Lanczos s'exprime sous la forme :

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

Preuve :

Cf. [bib26].

❑

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 7, cela provient de la petitesse du coefficient β_k et donc malgré l'orthogonalité théorique ($\left|\mathbf{v}^T \mathbf{q}_i\right| = 0 \ (i \le k)$), l'orthogonalité effective est loin d'être acquise ($\left|\mathbf{q}_{k+1}^T \mathbf{q}_i\right| >> \varepsilon \ \ (i \le k)$).

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 (cf. [bib23] pp66-78) :

- **Algorithme de Lanczos sans réorthogonalisation**, développé par J.K. Cullum et R.A. Willoughby [bib3] qui consiste à expurger le spectre calculé de ses modes fantômes en étudiant l'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 [bib14] (`METHOD = 'TRI_DIAG'`) du *Code_Aster*, c'est la **stratégie de réorthogonalisation complète** qui a été choisie: le vecteur $\mathbf{q}_{k+1}$ 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 (cf. [§5.4.2]),*
- *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. [§3.1]).*

5.3.2 Capture des multiplicités

On a vu qu'en théorie l'algorithme de Lanczos ne permettait pas, quelque 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 [bib22], 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.

5.3.3 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 ($m >> n$) 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.

5.4 Traitements complémentaires

5.4.1 Détection d'espaces invariants

Dans l'algorithme 10, 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 $\mathbf{A}_\sigma$.
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 :

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

La matrice tridiagonale obtenue a la forme suivante :

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

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 $(\alpha_k,\beta_k)_{k=l+1,m}$ sont obtenues à partir d'un deuxième vecteur aléatoire. Pour détecter la nullité du terme extra-diagonal on utilise le critère numérique

$$\left|\beta_{l-1}\right| \leq PREC\_LANCZOS \left|\alpha_l\right| \text{ alors } \beta_{l-1} = 0,$$

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

Remarques :

- *un critère plus robuste retenu par les grandes librairies mathématiques EISPACK [bib26], LAPACK [bib26] et ARPACK [bib27] utilise le terme diagonal précédent, un paramètre modulable c et la précision machine ε,*

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

  c'est ce critère qui à été retenu pour IRAM (mais l'utilisateur ne peut modifier le paramètre c, celui-ci est fixée 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. [§3.1], [§5.3.1]).*

5.4.2 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)

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

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 (Tchebycheff en réel et Faber en complexe) permettant d'axer la recherche modale dans telle ou telle zone. On parle alors d'accélérations polynomiales explicites [bib22].*

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

5.5 Implantation dans le *Code_Aster*

5.5.1 Variante de Newmann & Pipano

Cette variante développée par M. NEWMAN & A. PIPANO[bib3], [bib14], 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 ($\mathbf{A}$-$\sigma\mathbf{B}$).

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

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

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 cf. [§2.6], [§2.9]) et il permet de chercher des vecteurs de Lanczos ($\mathbf{A}$ - $\sigma\mathbf{B}$)-orthogonaux. La gonalisation totale ne s'effectue que si elle s'avère nécessaire et ce critère est paramètrable.

Remarques :

- *la $\boldsymbol{B}$-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 [§1.7] il existe toute une zoologie de couples opérateur - produit scalaire, celui-ci n'étant qu'une possibilité parmi d'autres. Ainsi, les bibliothèques [bib29] classiques proposent de traiter les problèmes de flambement en «buckling mode» via le même «shift and invert» et le pseudo-produit scalaire introduit par $\boldsymbol{A}$. 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 $\boldsymbol{B}$-produit scalaire.*

Le prix à payer pour ce gain en robustesse est la perte de symétrie possible de la matrice de Rayleigh

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

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 $\mathbf{q}_{init}$.

$\tilde{\mathbf{q}}_1 = \left(\mathbf{A} - \sigma\ \mathbf{B}\right)^{-1}\left(\mathbf{q}_{init}(i).\mathbf{u}_{lagr}(i)\right)_i$.

$\hat{\mathbf{q}}_1 = \mathbf{A}_\sigma\left(\tilde{\mathbf{q}}_1(i).\mathbf{u}_{bloq}(i)\right)_i$.

$\delta_1 = sign\left(\hat{\mathbf{q}}_1^T\left(\mathbf{A} - \sigma\ \mathbf{B}\right)\hat{\mathbf{q}}_1\right)$

$\mathbf{q}_1 = \delta_1\ \hat{\mathbf{q}}_1\left|\hat{\mathbf{q}}_1^T\left(\mathbf{A} - \sigma\ \mathbf{B}\right)\hat{\mathbf{q}}_1\right|^{-\frac{1}{2}}$

$\mathbf{q}_0 = \mathbf{0},\ \beta_0 = 0, \delta_0 = 0.$

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),

$\beta_k = \mathbf{v}^T\left(\mathbf{A} - \sigma\ \mathbf{B}\right)\mathbf{v}$,

$\delta_k = sign\left(\mathbf{v}^T\left(\mathbf{A} - \sigma\ \mathbf{B}\right)\mathbf{v}\right)$,

Si $\beta_k \neq 0$ alors

$\mathbf{q}_{k+1} = \dfrac{\delta_k \mathbf{v}}{\left|\beta_k\right|^{\frac{1}{2}}}$,

Sinon

Déflation;

Fin si.

Fin boucle.

Algorithme 11 : Variante de Newman-Pipano

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
$a = \mathbf{q}_{k+1}^T \left(\mathbf{A} - \sigma\, \mathbf{B}\right)\mathbf{q}_i$,
Si $\left(\left|\left\langle \mathbf{q}_{k+1}^T \left(\mathbf{A} - \sigma\, \mathbf{B}\right)\mathbf{q}_i \right\rangle\right| \geq PREC\_ORTHO\right)$ alors
Pour j = 1, *NMAX_ITER_ORTHO*
$\mathbf{x} = \mathbf{q}_{k+1} - \left(\mathbf{q}_{k+1}^T \left(\mathbf{A} - \sigma\, \mathbf{B}\right)\mathbf{q}_i\right)\mathbf{q}_i$
,
$b = \mathbf{x}^T \left(\mathbf{A} - \sigma\, \mathbf{B}\right)\mathbf{q}_i$
Si $\left(|b| \leq PREC\_ORTHO\right)$ 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
Echec, 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 12 : 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 [bib18] choisi pour l'IRAM (cf. [Annexe 2]).

5.5.2 Paramètrage

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 :

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

où p est le nombre de valeurs propres à calculer,
$n_{ddl\text{-}actifs}$ est le nombre de degrés de liberté actifs du système (cf. [§2.2])

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.

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 (cf. [§5.4.1]), 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 l.

5.5.3 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. [§2.9]) 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.

Remarques :

- *la méthode IRA que nous allons aborder propose un contrôle modulable et dynamique de la précision des résultats, c'est une méthode itérative,*
- *dans la V5, en parallèle de la position modale et de la valeur propre (en fréquence si on est en dynamique), on affiche désormais la norme du résidu du problème initial calculée en post-traitement (après avoir normalisé à l'infini les vecteurs propres (afin d'exhumer des vecteurs propres «neutres» vis-à-vis de la kyrielle de normalisations qu'ils peuvent subir lors des calculs des paramètres modaux de la structure (facteurs de participation...)).*

5.5.4 Périmètre d'utilisation

Attention, cette méthode est la seule utilisable pour traiter un problème quadratique. En cas d'erreur lors de l'initialisation du mot-clé `METHODE`, le calcul s'arrête et une trace est laissée dans les fichiers de sortie.

Dans la V5, 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. [bib25] 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 le *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. 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 (cela peut se produire en présence de clusters au voisinage des bornes ou pour des opérateurs pathologiques (mal conditionnés, non normaux, défectifs...) qui devraient toutefois être assez rares (cela révèle souvent un problème mal posé). On peut aussi envisager de l'encapsuler dans un opérateur auxiliaire (comme `IMPR_STURM`).

Remarque :

Le traitement de problème quadratique est incompatible avec les options `'MODE_FLAMB'` et `'MODE_RIGIDE'`.

5.5.5 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
          5      4.73076E+01       4.26463E-12          4
          6      4.73076E+01       1.43391E-12          4
          7      5.45850E+01       9.52006E-12          4
          8      8.80156E+01       3.45489E-13          4
          9      1.01614E+02       6.13949E-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 : MODE_ITER_SIMULT avec 'TRI_DIAG'

Récapitulons maintenant le paramètrage disponible de l'opérateur MODE_ITER_SIMULT avec cette option METHOD = 'TRI_DIAG'.

5.5.6 Récapitulatif du paramètrage

Mot-clé facteur	Mot-clé		Valeur par défaut	Références
	TYPE_RESU	'DYNAMIQUE'	'DYNAMIQUE'	[§2.1]
		'MODE_FLAMB'		[§2.1]
	METHODE	**'TRI_DIAG'**	'SORENSEN'	[§5.5.3]
	OPTION	'MODE_RIGIDE'	'SANS'	[§5.5.4]
		'SANS'		[§5.5.4]
CALC_FREQ	**FREQ**			[§4.4]
	CHAR_CRIT			[§4. 4]
	OPTION	'PLUS_PETITE'	'PLUS_PETITE''	[§4.4]
		'BANDE'		[§4.4]
		'CENTRE'		[§4.4]
	NMAX_FREQ		10	[§4.4]
	DIM_SOUS_ESPACE		Calculé	[§5.5.2]
	PREC_ORTHO		1.E-12	[§5.5.1], [§5.5.2]
	NMAX_ITER_ORTHO		1.E-04	[§5.5.1], [§5.5.2]
	PREC_LANCZOS		1.E-04	[§5.5.1], [§5.5.2]
	NMAX_ITER_QR		15	[§5.5.1], [§5.5.2]
	NPREC_SOLVEUR		8	[§2.6]
	NMAX_ITER_SHIFT		5	[§2.6]
	PREC_SHIFT		0.05	[§2.6]
	SEUIL_FREQ		1.E-02	[§2.9]
VERI_MODE	STOP_ERREUR	'OUI'	'OUI'	[§2.9]
		'NON'		[§2.9]
	PREC_SHIFT		5.E-03	[§2.9]
	SEUIL		1.E-06	[§2.9]
	STURM	'OUI'	'OUI'	[§2.9]
		'NON'		[§2.9]

Tableau 5.5.6-a : Récapitulatif du paramètrage de MODE_ITER_SIMULT avec 'TRI_DIAG'

Remarques :

- *dans la V5, 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,*
- *on retrouve toute la "tripaille" de paramètres liée aux pré-traitements du test de Sturm (NPREC_SOLVEUR, 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.***

6 Algorithme IRA (*METHODE = 'SORENSEN'*)

6.1 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(cf. [§5.3.1]). Une **généralisation** de cet algorithme **au cas non hermitien** imaginée par **W.E. Arnoldi** [bib31] **en 1951** permet de résoudre partiellement cette problématique. Elle a été remise au goût du jour par Y. Saad [bib32] en 1980 et une abondante littérature couvre le sujet. Mis à part l'article fondateur et les papiers de Y.Saad et M. Sadkane [bib22], on recommande la synthèse actualisée et exhaustive de J.L. Vaudescal [bib23] (pp79-112).

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 paramètrage effectif dans le *Code_Aster* et par un exemple de fichier message.

6.2 Algorithme d'Arnoldi

6.2.1 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 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]), d'où un gain de l'ordre de $O(10m^3/3)$.
L'algorithme est très similaire à celui de Lanczos, il consiste à construire progressivement une famille de vecteur d'Arnoldi $\mathbf{q}_1$, $\mathbf{q}_2$,..., $\mathbf{q}_m$ 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 $m^{\text{ième}}$ vecteur de la base canonique, le **vecteur résidu de la factorisation d'Arnoldi** s'écrit $\mathbf{R}_m = b_{m+1,m}\ \mathbf{q}_{m+1}\ \mathbf{e}_m^T$.

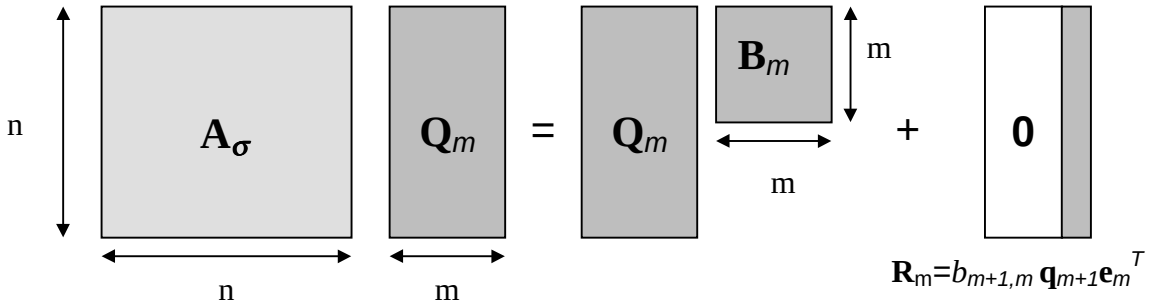


Figure 6.2.1-a : Factorisation d'Arnoldi

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 :

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

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*(cf. [§5.3.1]),
- *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* $\boldsymbol{B}_m$*, 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 [bib32] 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 [bib22], 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.*

6.2.2 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 10

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

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

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 $\left\|\left(\mathbf{I} - \mathbf{P}_m\right)\mathbf{u}\right\|_2$ on peut généraliser le théorème de convergence 8 au cas non hermitien.

Théorème 11

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 :

$$\left|\lambda_1 - \tilde{\lambda}_1\right| \leq \frac{\alpha^2}{\beta}\xi_1\ \delta_1^m$$

avec $\alpha = \dfrac{\beta}{\left\|\mathbf{P}_m \mathbf{u}_i\right\|_2}$, β la constante du théorème 6, $\xi_1 = \sum_{k=1, k\neq 1}^{n}\left|\dfrac{\alpha_k}{\alpha_1}\right|$ et

$\delta_1^m = \left(\sum_{j=2}^{m+1}\prod_{k=2, k\neq j}^{m+1}\left|\dfrac{\lambda_k - \lambda_1}{\lambda_k - \lambda_j}\right|\right)^{-1}$. Ce résultat se décline de la même manière sur les autres modes.

Preuve :

En reprenant la démonstration de Y. Saad ([bib33] pp209-210) et le résultat du théorème 6.

❑

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 δ_i^m),
- la décroissance de l'erreur est proportionnelle à l'augmentation de m (propriété de δ_i^m).

Remarques :

- *lorsqu'une valeur propre est mal conditionnée $\xi_i \to +\infty$ alors il faut augmenter m pour que δ_i^m décroisse*,
- *des résultats analogues ont été exhumés dans le cas d'un opérateur défectif (cf. Zia 94 [bib23]).*

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

6.3 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** $O(4nm(m-p))$ et **mémoires** $O(2np+p^2)$ **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 le 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 échouée 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.

6.4 Algorithme 'Implicit Restarted Arnoldi' (IRA)

Cet algorithme a été initié par D.C. Sorensen [bib30] 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 $\mathbf{QR}$ 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. [§6.5.3]) 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 $\mathbf{QR}$) 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 $\mathbf{QR}$ 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 $\mathbf{QR}$. Après avoir mis à jour les matrices, on les tronque jusqu'à l'ordre p

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

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 $\mathbf{QR}$ 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 (cf. [§5.4.2]) 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: $\mathbf{AQ}_m = \mathbf{Q}_m\mathbf{B}_m + \mathbf{R}_m$.
Pour k = 1, *NMAX_ITER_SOREN* faire
  Calculer $\underbrace{\tilde{\lambda}_1, \tilde{\lambda}_2 \ldots \tilde{\lambda}_p}_{\textit{Conservées pour amélioration}}$, $\underbrace{\tilde{\lambda}_{p+1} \ldots \tilde{\lambda}_{p+q}}_{\textit{Utilisées comme shifts}}$,
  QR avec shifts implicites,
  Mise a jour $\mathbf{Q}_m, \mathbf{B}_m$ et $\mathbf{R}_m$,
  Troncature de ces matrices a l'ordre p,
  $\mathbf{AQ}_p = \mathbf{Q}_p\mathbf{B}_p + \mathbf{R}_p \Rightarrow \mathbf{AQ}_m = \mathbf{Q}_m\mathbf{B}_m + \mathbf{R}_m$,
  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** $\|\mathbf{r}_i\|_2 = \left\|\left(\mathbf{A}_\sigma - \tilde{\lambda}_i\ \mathbf{I}\right)\tilde{\mathbf{u}}_i\right\|_2$ via la propriété 10. On a déjà mentionné que dans le cas non hermitien, ils ne sauraient suffir à cette tâche, notamment en cas de fort défaut de normalité. Pour s'en acquitter, sans avoir recours à d'autres informations (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 parfois difficile à obtenir, même a posteriori !) a priori, on utilise la propriété précédente complétée par un critère dû à Z.Bai et al [bib34]

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

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. [§2.9]) qui est affichée dans les colonnes résultats.

Remarques :

- *la technique d'accélération polynomiale utilisée est plus efficace que celle de Tchebycheff, 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 $\boldsymbol{B}$ en «shift and invert») un filtrage de type Ericsson & Ruhe [bib35] 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),*
- *cet algorithme peut être vu comme une forme tronquée de l'algorithme $\boldsymbol{QR}$ 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.

6.5 Implantation dans le *Code_Aster*

6.5.1 ARPACK

Le package d'origine [bib29] (**ARPACK** pour ARnoldi PACKage) codant la méthode est disponible en freeware sur internet. Ces concepteurs, D.Sorensen, R.Lehoucq et C.Yang de la Rice University de Houston, l'ont voulu à la fois :

- simple d'accès (FORTRAN77, «reverse communication»),
- modulable (il est basé sur les bibliothèques LINPACK, LAPACK [27] et BLAS [bib36] (BLAS est l'acronyme pour Basic Linear Algebra Subprograms)),
- et riche en fonctionnalités (décomposition de Schur, shifts modulables, nombreuses transformations spectrales).

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 ASTER, les produits matrice-vecteur, les factorisations, les résolutions de système...

6.5.2 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-$\mathbf{QL}$ (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 $\mathbf{QR}$), sur **le couple opérateur de travail-produit scalaire** suivant :

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

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

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 (cf. [§5.5.1]) 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.

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 [§2.2]. 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 **égale au nombre de ddls actifs** (cf. [§2.2]), 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, [§2.5]) 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 $p = n_{ddl\text{-}actifs}$, 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 ddls sort du périmètre d'utilisation «idéal» de ce type d'algorithme (cf. [§2.8]). Un bon $\boldsymbol{QR}$ *serait sans nul doute plus efficace, mais c'est un bon moyen de tester l'algorithme.*

6.5.3 Paramètrage

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 :

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

où p est le nombre de valeurs propres à calculer,
$n_{ddl\text{-}actifs}$ est le nombre de degrés de liberté actifs du système (cf. [§2.2])

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.
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 (ε = 2.22.E-16 sur SGI).

6.5.4 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
         4     -7.10527E+05       1.72306E-12
         5     -7.11205E+05       2.41783E-12
         6     -7.11542E+05       7.88981E-13
         7     -7.11703E+05       5.71621E-13
         8     -1.50492E+06       1.17776E-11
         9     -6.02258E+06       2.42221E-11
        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 : MODE_ITER_SIMULT avec '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.

Récapitulons maintenant le paramètrage disponible de l'opérateur MODE_ITER_SIMULT avec cette option METHOD = 'SORENSEN'.

6.5.5 Récapitulatif du paramètrage

Mot-clé facteur	Mot-clé		Valeur par défaut	Références
	TYPE_RESU	*'DYNAMIQUE'*	*'DYNAMIQUE'*	[§2.1]
		'MODE_FLAMB'		[§2.1]
	METHODE	***'SORENSEN'***	*'SORENSEN'*	[§6.4]
CALC_FREQ	**FREQ**			[§4.4]
	CHAR_CRIT			[§4. 4]
	OPTION	'PLUS_PETITE'	'PLUS_PETITE''	[§4.4]
		'BANDE'		[§4.4]
		'CENTRE'		[§4.4]
	NMAX_FREQ		10	[§4.4]
	DIM_SOUS_ESPACE		Calculé	[§6.5.3]
	PREC_SOREN		0.	[§6.4], [§6.5.3]
	NMAX_ITER_SOREN		20	[§6.4], [§6.5.3]
	PARA_ORTHO_SOREN		0.717	[§6.5.3], [Annexe 2]
	NPREC_SOLVEUR		8	[§2.6]
	NMAX_ITER_SHIFT		5	[§2.6]
	PREC_SHIFT		0.05	[§2.6]
	SEUIL_FREQ		1.E-02	[§2.9]
VERI_MODE	STOP_ERREUR	'OUI'	'OUI'	[§2.9]
		'NON'		[§2.9]
	PREC_SHIFT		5.E-03	[§2.9]
	SEUIL		1.E-06	[§2.9]
	STURM	'OUI'	'OUI'	[§2.9]
		'NON'		[§2.9]

Tableau 6.5.5-a : Récapitulatif du paramètrage de MODE_ITER_SIMULT avec 'SORENSEN'

Remarques :

- *dans la V5, 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,*
- *on retrouve toute la "tripaille" de paramètres liée aux pré-traitements du test de Sturm (NPREC_SOLVEUR, 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.***

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

7.1 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 $\mathbf{A}^{\sigma}\ \mathbf{x} = (\lambda - \sigma)\,\mathbf{B}\,\mathbf{x}$. On distingue quatre parties dans l'algorithme[bib1], [bib4] :

- 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 dans le sous-espace de Ritz en résolvant $\left(\overline{\mathbf{A}} - \overline{\lambda}_i\ \overline{\mathbf{B}}\right)\mathbf{u}_i = 0$ où $\overline{\mathbf{A}} = \mathbf{Q}^T\mathbf{A}^{\sigma}\mathbf{Q}$ et $\overline{\mathbf{B}} = \mathbf{Q}^T\mathbf{B}\mathbf{Q}$. On revient ensuite à l'espace initial (pour les vecteurs propres) via la transformation $\mathbf{X} = \mathbf{Q}\mathbf{U}$ où $\mathbf{U} = \left[\left\{\mathbf{u}_i\right\}\right]$,
- tester la convergence des modes propres λ_i.

7.2 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 $\overline{\mathbf{A}}$ et $\overline{\mathbf{B}}$ 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

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

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.

7.3 Implantation dans le *Code_Aster*

7.3.1 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). $r = \left(p + q\right)/2$ semble un bon choix. Pour q on prend habituellement [bib1]

$$q = \min\left(p + 8\ ,\ 2p\right).$$

7.3.2 Choix des vecteurs initiaux

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

- premier vecteur tel que $x_{1_i} = \frac{\mathbf{A}_{ii}^{\sigma}}{\mathbf{B}_{ii}}$,
- pour les autres vecteurs de 2 à $(q-1)$

$$\mathbf{x}_2 = \begin{Bmatrix} 0 \\ 1 \\ 0 \end{Bmatrix} \text{--ligne } i_1 \qquad \mathbf{x}_{q-1} = \begin{Bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{Bmatrix} \text{--ligne } i_2 \quad ,...$$

  où 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.

7.3.3 Paramètres dans le *Code_Aster*

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'	[§2.1]
		'MODE_FLAMB'		[§2.1]
	METHODE	**'JACOBI'**	'SORENSEN'	[Annexe 3]
CALC_FREQ	**FREQ**			[§4.4]
	CHAR_CRIT			[§4. 4]
	OPTION	'PLUS_PETITE'	'PLUS_PETITE''	[§4.4]
		'BANDE'		[§4.4]
		'CENTRE'		[§4.4]
	NMAX_FREQ		10	[§4.4]
	DIM_SOUS_ESPACE		Calculé	[§7.3.1]
	PREC_BATHE		1.E-10	[§7.2]
	NMAX_ITER_BATHE		40	[§7.2]
	PREC_JACOBI		1.E-12	[Annexe 3]
	NMAX_ITER_JACOBI		12	[Annexe 3]
	NPREC_SOLVEUR		8	[§2.6]
	NMAX_ITER_SHIFT		5	[§2.6]
	PREC_SHIFT		0.05	[§2.6]
	SEUIL_FREQ		1.E-02	[§2.9]
VERI_MODE	STOP_ERREUR	'OUI'	'OUI'	[§2.9]
		'NON'		[§2.9]
	PREC_SHIFT		5.E-03	[§2.9]
	SEUIL		1.E-06	[§2.9]
	STURM	'OUI'	'OUI'	[§2.9]
		'NON'		[§2.9]

Tableau 7.3.3-a : Récapitulatif du paramètrage de MODE_ITER_SIMULT avec 'JACOBI'

Remarques :

- *dans la V5, 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,*
- *on retrouve toute la "tripaille" de paramètres liée aux pré-traitements du test de Sturm (NPREC_SOLVEUR, 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.***

8 Conclusion - Synthèse

Les périmètres d'utilisation optimaux des opérateurs modaux du *Code_Aster* 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 a fait indiqué. Il regroupe des algorithmes heuristiques et ceux de type puissances (cf. [§3]), qui ont été historiquement développés les premiers pour résoudre des problèmes modaux génériques. Il couple une phase de localisation des valeurs propres (via une technique de bissection et une méthode de la sécante) avec une amélioration de ces estimations et un calcul des vecteurs propres associés par une méthode d'itérations inverses (mâtinée, ou non, de quotient de Rayleigh).

Par contre, pour **capturer une partie significative du spectre,** on a recourt à **MODE_ITER_SIMULT**. Ce dernier fédère les méthodes dites de «sous-espace» (Lanczos [§4], [§5], IRAM [§6], Bathe & Wilson [§7]) qui projettent l'opérateur de travail afin d'obtenir un spectre approximé de taille plus réduite (traité alors par une méthode globale de type **QR** ou Jacobi, cf. [Annexe 1] et [Annexe 3]). Outre leurs qualités numériques (complexités calcul et mémoire réduites, couplage facilité de techniques de déflation, de redémarrage et d'accélération...) et mathématiques (bonne convergence, contrôle de la qualité de la spectre de l'opérateur projeté...), il faut noter qu'elles ne requièrent pas nécessairement la connaissance de l'opérateur de travail mais celle de son action sur un vecteur (cette particularité est très utile pour traiter de gros systèmes mais elle n'est pas utilisée dans le *Code_Aster* où on assemble toutes les matrices avant de les traiter).

Jusqu'à présent, ces algorithmes achoppaient régulièrement sur les mêmes écueils: la détection correcte de modes multiples, de modes de corps rigide et de manière générale, le traitement de spectre tassé. Tout ceci conduisait à l'apparition de **modes «fantômes»** parfois mal aisés à détecter (modes correspondants à des multiplicités ratées et pouvant générer des résidus corrects au sens d'Aster, et ce, d'autant plus que les critères des vérifications post-modales étaient parfois permissifs (résidu en 10^{-2} au lieu du 10^{-6} actuel), désactivés voire insuffisants (test de Sturm limité aux valeurs propres positives)) qui suscitent des distorsions de résultats en aval du calcul, lors des projections sur base modale. Pour s'affranchir des problèmes récurrents à ce type d'approche, on a donc proposé d'enrichir MODE_ITER_SIMULT (à partir de la V5) de **l'algorithme IRA** ('Implicit Restarted Arnoldi' [§6]).
Cette variante d'Arnoldi, initiée par D.C. Sorensen en 1992, 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 itérations **QR** implicites.
L'algorithme IRA tente d'apporter un remède élégant aux problèmes récurrents soulevés par les autres approches. On n'a 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...

En bref, **numériquement,** IRAM procure une meilleure robustesse globale tout en améliorant les complexités calcul et mémoire (surtout par rapport à un Lanczos simple tel que celui de Newman & Pipano) pour une précision fixée.
D'un **point de vue fonctionnel**, l'implantation de cette méthode a permis un gain en complexité calcul (observé) à minima d'ordre 2. Désormais, l'utilisateur a un réel contrôle sur la qualité des modes via un paramétrage idoine. On n'a, de plus, renforcé la sévérité des vérifications post-modales et étendu leur domaine d'application.

L'utilisation d'IRAM (**par défaut dans MODE_ITER_SIMULT**) **est donc à conseiller dans tous les cas de figures**, y compris pour la recherche de quelques valeurs propres (on peut même tirer partie des excellentes propriétés de la méthode des puissances inverses concernant le calcul de vecteurs propres et, affiner le résultat, en redémarrant MODE_ITER_INV avec pour estimation les valeurs propres exhumées par MODE_ITER_SIMULT). Au delà de leurs spécificités numériques et fonctionnelles qui sont reprises dans ce document, on peut synthétiser les méthodes modales du *Code_Aster* sous la forme du tableau ci-dessous (**les valeurs par défaut sont matérialisées en gras**).

Opérateur/ Périmètre d'application	Algorithme	Mot-clé	Avantages	Inconvénients
MODE_ITER_INV				
1ère phase (heuristique)				
Calcul de quelques modes	Bissection	'SEPARE'		
Calcul de quelques modes	Bissection + Sécante	**'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
2ième phase (méthode des puissances proprement dite)				
Méthode de base	Puissances inverses	**'DIRECT'**	Très bonne construction de vecteurs propres	Peu robuste
Option d'accélération	Quotient de Rayleigh	'RAYLEIGH'	Améliore la convergence	Coût calcul
MODE_ITER_SIMULT				
Calcul d'une partie du spectre	Bathe & Wilson	'JACOBI'		Peu robuste
	Lanczos (Newman- Pipano)	'TRI_DIAG'		Peu robuste
	IRAM (Sorensen)	**'SORENSEN'**	Robustesse accrue. Meilleures complexités calcul et mémoire. Contrôle de la qualité des modes.	

Tableau 8-a : Récapitulatif des méthodes modales du *Code_Aster*

IRAM a permis de solder toutes les anomalies logicielles liées aux problèmes modaux généralisés, mais il semble que seule sa déclinaison par blocs ou une version incorporant du «purge and lock»[bib23] peuvent nous garantir une détection «quasi-infaillible» du spectre d'un opérateur standard (i.e. pas trop mal conditionné).
Une classe d'algorithme dit de «Jacobi-Davidson» [bib37] semble encore plus prometteuse pour traiter des cas pathogènes. Elle utilise un algorithme de type Lanczos qu'elle préconditionne via une méthode de Rayleigh.
Concluons en notant que l'algorithme IRA n'a pas encore été étendu au cas quadratique dans les opérateurs modaux du code [R5.01.02].

9 Bibliographie

[1] K.J. BATHE : Solution methods for large generalized eigenvalue problems in structural engineering. California university Berkeley, 1971.

[2] F. CHATELIN : Valeurs propres de matrices. Masson, 1988.

[3] J.K. CULLUM & R.A. WILLOUGHBY : Lanczos algorithms for large symetric eigenvalue computations. Vol 1 Theory, Birkhäuser 1985.

[4] DHATT & TOUZOT : Une présentation de la méthode des éléments finis. Maloine S.A. Edition,1984.

[5] A. DUBRULLE, R.S. MARTIN & J.H. WILKINSON : The Implicit **QL** Algorithm pp241-248, dans Handbook for automatic computation. Vol 2, Linear algebra, Springer-Verlag, 1971.

[6] G.H.GOLUB & C.F. Van LOAN : Matrix computations, The Johns Hopkins university press, 1989.

[7] Y. HAUGAZEAU : Application du théorème de Sylvester à la localisation des valeurs propres de $\mathbf{Ax} = \lambda\mathbf{Bx}$ dans le cas symétrique. RAIRO Analyse numérique, vol.14, n°1 pp25-41 ,1980.

[8] D. HO : Tchebychev acceleration technique for large scale nonsymmetric matrices. Numerische mathematik. Vol 56, pp731-734.

[9] J.F. IMBERT : Analyse des structures par éléments finis. Sup Aéro, CEPADUES Editions.

[10] 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, oct. 1950.

[11] P. LASCAUX & R. THEODOR : Analyse numérique matricielle appliquée à l'Art de l'ingénieur. Masson, 1986.

[12] R.S. MARTIN, G. PETERS & J.H. WILKINSON : The **QR** Algorithm for Real Hessenberg Matrices pp358-371, dans Handbook for automatic computation,. Vol. 2, Linear algebra, Springer-Verlag, 1971.

[13] R.S. MARTIN & J.H. WILKINSON : Similarity Reduction of a General Matrix to Hessenberg From pp339-358, dans Handbook for automatic computation. Vol. 2 Linear algebra, Springer-Verlag, 1971.

[14] M. NEWMANN & A. PIPANO : Fast modal extraction in NASTRAN via FEER computer programs, NASTRAN, user manuel, NASA Langley Research Center pp485-506, 1977.

[15] B. NOUR-OMID, B.N. PARLETT & R.L. TAYLOR : Lanczos versus subspace iteration for solution of eigenvalue problems, Int. J. for numerical methods in engineering. Vol. 19, pp859-871, 1983.

[16] I.V. OJALVO & M. NEWMANN : Vibrations modes of large structures by an automatic matrix-reduction method. AIAA, vol.8 n°7, 1970, pp1234-1239.

[17] E.E. OSBORNE : On pre-conditioning of matrices. J. Assoc. Comput. Mach., vol 7, pp338-345, 1960.

[18] B.N. PARLETT : The symetric eigenvalue problem. Prentice hall, Englewoods Cliffs, 1980.

[19] B.N. PARLETT & C. REINSCH : Balancing a matrix for calculation of eigenvalues and eigenvectors. pp315-326, dans Handbook for automatic computation. Vol. 2, Linear vlgebra, Springer-Verlag, 1971.

[20] 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, october 1980.

[21] H.D. SIMON : The Lanczos algorithm with partial reorthogonalisation, Mathematics of computation. Vol 42, n° 165, pp115-142, 1984.

[22] M. SADKANE : A block Arnoldi-Tchebychev method for computing the leading ergenpairs of large sparse unsymmetric matrices. Numerische mathematik, vol 64, pp181-193, 1993.

[23] J.L. VAUDESCAL : Introduction aux méthodes de résolution de problèmes aux valeurs propres de grande taille. Note HI-72/00/01A, 2000.

[24] J.H. WILKINSON & C. REINSCH : Handbook for automatic computation, vol 2, Linear Algebra, Springer-Verlag, New-York, 1971.

[25] O. BOITEAU & B. QUINNEZ : Méthode d'analyse spectrale dans le *Code_Aster*. Fascicule du Cours de dynamique, mars 2000.

[26] C.C. PAIGE : Accuracy and effectivness of the Lanczos algorithm for the symmetric eigenproblem. Linear algebra and its applications, vol 34, 1980.

[27] J.J. DONGARRA & al : LAPACK's users guide. SIAM, 1992.

[28] J.J. DONGARRA & al : EISPACK's users guide. Springer verlag, vol 6, 1976.

[29] R.B. LEHOUCQ, D.C. SORENSEN & C. YANG : ARPACK's users guide. 1996.

[30] D.C. SORENSEN : Implicit applications of polynomial filters in a k-step Arnoldi method. SIAM J. Matrix Anal. Appl., vol 13, pp357-385, 1992.

[31] W.E. ARNOLDI : The principle of minimized iterations in the solution of the matrix eigenvalue problem. Quart. Appl. Math, vol 9, pp17-19, 1951.

[32] Y. SAAD : Variation on Arnoldi's method for computing eigenelements of large unsymetric matrices. Linear algebra and its applications, vol 34, pp269-2395, 1980.

[33] Y. SAAD : Numerical meyhods for large eigenvalue problems. Manchester university, press series in algorithms and architectures for advanced scientific computing, 1991.

[34] 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.

[35] 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.

[36] J.J. DONGARRA & al : An extended set of Fortran basic linear algebra subprograms. ACM transactions on mathematical software, vol 14, pp1-17, 1998.

[37] R.B. MORGAN & D.S. SCOTT : Preconditionning the Lanczos algorithm for sparse symmetric eigenvalue problems. SIAM J. Sci. Comp, vol 14, 1993.

Annexe 1 Généralités sur l'algorithme *QR*

A1.1 Principe

Les algorithmes de type **QR**(cf [§2.8]) 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 **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

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

Ceci peut s'effectuer via diverses transformations orthogonales (Householder, Givens, Gram-Schmidt...) et leur coût (de l'ordre de $O(10n^3/3)$) est négligeable comparé au gain qu'elles permettent de réaliser à chaque itération du processus global: $O(n^2)$ (avec Householder ou Fast-Givens on a plus précisément $O(2n^2)$ contre $O(4n^2)$ avec un Givens simple) 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) : $O(20n)$.

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 $i^{\text{ème}}$ mode (rangés classiquement par ordre décroissant de module) s'effectue alors en :

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

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 (cf. [§5.5.1]) est effectuée via un* ***QR*** *(ou un* ***QL*** *en symétrique) simple de ce type (avec, au préalable, l'équilibrage). Le seul paramètre accessible par l'utilisateur est le nombre maximum d'itérations admissibles* `NMAX_ITER_QR`,
- *pour IRAM(cf. [§6.4]), 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 le 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* $\boldsymbol{QR}$ *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.

A1.2 La stratégie du shift

Cette stratégie consiste à **susciter artificiellement un phénomène de déflation** (cf. [§3.1], [§5.4.1]) 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 : $\mathbf{QR}$ 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** [bib24] 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 réfreiner 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 $\mathbf{QR}$, au pire, cela fausse complètement son fonctionnement.
Afin de palier à ces sournois inconvénients numériques, J.C. Francis a mis au point une version avec **double shift implicite**. Elle est très bien explicitée dans [bib6] (pp377-81) et on se contentera juste ici d'en résumer la philosophie.

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

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

avant de la factoriser sous forme $\mathbf{QR}$ 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 rien que le 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 :

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

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.

A1.3 Equilibrage

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. A ce sujet, E.E. Osborne [bib17] (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

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

telle que :

$$\left\|\tilde{\mathbf{A}}\right\|_2 << \|\mathbf{A}\|_2 .$$

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 :

$$\mathbf{A}_k \xrightarrow[k\to\infty]{} \mathbf{A}_f$$

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

Remarque :

- *cette technique a été généralisée à toute norme matricielle induite par la norme discrète de l^q avec l^q l'ensemble des suites complexes $(u_n)_n$ telles que $\sum_n |u_n|^q < \infty$,*
- *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.
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 $i-1 \equiv k-1 \quad [n]$). 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 $\overline{\mathbf{D}}_k$ telle que :

$$\overline{\mathbf{D}}_k = \begin{cases} \mathbf{Id} + (f-1)\,\mathbf{e}_i\,\mathbf{e}_i^T & \text{si} \quad \left(\mathbf{C}_k\ f\right)^p + \left(\mathbf{R}_k\ /\ f\right)^p < \gamma\left(\mathbf{C}_k^p + \mathbf{R}_k^p\right) \\ \mathbf{Id} & \text{sinon} \end{cases}$$

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

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

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

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 $\boldsymbol{X}$ car les termes diagonaux des deux matrices triangulaires supérieures sont des valeurs propres de la matrice de travail.
- *dans le Code_Aster, on a choisi $p = 1$ et $\gamma = 0.95$ (cf. [bib19]),*
- *avant d'appliquer la méthode $\boldsymbol{QR}$, on commence par équilibrer la matrice de travail et ensuite, on la transforme sous forme de Hessenberg supérieure. Une fois le calcul $\boldsymbol{QR}$ 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.*

A1.4 La méthode QL

La factorisation $\mathbf{QL}$ d'une matrice $\mathbf{A}$ consiste à orthonormaliser ses vecteurs colonnes en commençant par le dernier (contrairement à l'algorithme $\mathbf{QR}$ qui débute par le premier) donnant ainsi une matrice $\mathbf{L}$ triangulaire inférieure régulière. On montre d'ailleurs que l'algorithme $\mathbf{QL}$ appliqué à l'opérateur $\mathbf{A}$ inversible est équivalent à l'algorithme $\mathbf{QR}$ appliqué à $\mathbf{A}^{-*}$ (matrice transposée conjuguée de l'inverse de $\mathbf{A}$).
La méthode mise en place dans le *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 2x2 le plus haut.

Remarque :

Contrairement à $\boldsymbol{QR}$ 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.

Annexe 2 Orthogonalisation de Gram-Schmidt

A2.1 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 le *Code_Aster*, nous avons retenu une **version itérative du processus de Gram-Schmidt** (l'IGSM de Kahan-Parlett [bib18]) 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 [bib23] (pp33-36) à partir d'un exemple très simple.

A2.2 Algorithme de Gram-Schmidt (GS)

Etant donné k **vecteurs indépendants** de $\Re^n$, $\left(\mathbf{x}_i\right)_{i=1,k}$ on désire obtenir k **vecteurs orthonormaux** (par rapport à un produit scalaire quelconque) $\left(\mathbf{y}_i\right)_{i=1,k}$ 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 $r_{ji} = \left(\mathbf{y}_j, \mathbf{x}_i\right)$;
  Fin boucle.
  Calcul de $\hat{\mathbf{y}}_i = \mathbf{x}_i - \sum_{j=1,i-1} r_{ji}\ \mathbf{y}_j$,
  Calcul de $r_{ii} = \left\|\hat{\mathbf{y}}_i\right\|$,
  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.

Remarques :

- *le problème est tout à fait similaire à celui rencontré par les inversions de systèmes mises en place dans le code (cf. [§2.6]),*
- *en notant $\boldsymbol{Q}$ la matrice engendrée par les $\left(\mathbf{y}_i\right)_{i=1,k}$, on a donc construit explicitement une factorisation $\boldsymbol{QR}$ de la matrice initiale $\boldsymbol{X}$ liée aux $\left(\mathbf{x}_i\right)_{i=1,k}$. C'est en fait le but de tout procédé d'orthogonalisation.*

A2.3 Algorithme de Gram-Schmidt Modifié (GSM)

Afin d'évincer 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 $r_{ji} = \left(\mathbf{y}_j, \hat{\mathbf{y}}_i\right)$,
    Calcul de $\hat{\mathbf{y}}_i = \hat{\mathbf{y}}_i - r_{ji}\,\mathbf{y}_j$;
  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 $\boldsymbol{QR}$ de la matrice initiale $\boldsymbol{X}$. Il requiert seulement $O(2nk^2)$ opérations (avec n le nombre de lignes de la matrice).

A2.4 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 [bib18] (cf. pp105-110).

Pour $\mathrm{i} = 1, \mathrm{k}$ faire
  $\hat{\mathbf{y}}_i = \mathbf{x}_i$,
  Pour $\mathrm{j} = 1, \mathrm{i} - 1$ faire
    Calcul de $\mathrm{r_{ji}} = \left(\mathbf{y}_j, \hat{\mathbf{y}}_i\right)$,
    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 $\tilde{\mathrm{r}}_{\mathrm{ji}} = \left(\mathbf{y}_j, \tilde{\mathbf{y}}_i\right)$,
      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
        $\hat{\mathbf{y}}_{\mathrm{i}} \Leftarrow \mathbf{0}$,
        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 le *Code_Aster*, le critère α est paramètré par le mot-clé PARA_ORTHO_SOREN (cf. [§6.5]). On montre que son intervalle de validité est $[1.2e,0.83\text{-}\varepsilon]$ avec ε la précision machine et on lui attribue généralement la valeur 0.717 (par défaut).

Remarques :

- *plus la valeur du paramètre α est grande, moins la réorthogonalisation se déclenche, mais cela affecte la qualité du processus,*
- *contrairement à la version «maison» développée pour Lanczos (cf. [§5.5.1]) 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 et al. (papier de 1976 soumis à Mathematics of Computation, vol.30, pp772-795).*

Annexe 3 Méthode de Jacobi

A3.1 Principe

La méthode de Jacobi [bib4], [bib6], [bib11] 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. [§7]). Elle consiste à transformer les matrices $\mathbf{A}$ et $\mathbf{B}$ du problème $\mathbf{A}\,\mathbf{u} = \lambda\,\mathbf{B}\,\mathbf{u}$ 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 :

$$\mathbf{A}^1 = \mathbf{A} \qquad \mathbf{B}^1 = \mathbf{B}$$

$$\mathbf{A}^2 = \mathbf{Q}_1^T \mathbf{A}^1 \mathbf{Q}_1 \qquad \mathbf{B}^2 = \mathbf{Q}_1^T \mathbf{B}^1 \mathbf{Q}_1$$

...

$$\mathbf{A}^k = \mathbf{Q}_{k-1}^T \mathbf{A}^{k-1} \mathbf{Q}_{k-1} \qquad \mathbf{B}^k = \mathbf{Q}_{k-1}^T \mathbf{B}^{k-1} \mathbf{Q}_{k-1}$$

$$\mathbf{A}^k \xrightarrow[k\to\infty]{} \mathbf{A}^d \text{ matrice diagonale} \qquad \mathbf{B}^k \xrightarrow[k\to\infty]{} \mathbf{B}^d \text{ matrice diagonale}$$

Algorithme 3.1 : Processus de Jacobi

Les valeurs propres sont données par $\lambda = \mathbf{A}^d \left(\mathbf{B}^d\right)^{-1}$ 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) non diagonal et non nul de $\mathbf{A}^k$ ou de $\mathbf{B}^k$ soit nul après la transformation (pour le choix de ce terme, cf. [§0]).

A3.2 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 ?

A3.2.1 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 (cf. [§0]). 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 $\mathbf{A}_{13}^k,...,\mathbf{A}_{1n}^k,\mathbf{A}_{23}^k,...$. 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 le *Code_Aster*.

A3.2.2 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 :

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

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

$$f_\lambda = \underset{i}{Max} \frac{\left|\lambda_i^k - \lambda_i^{k-1}\right|}{\lambda_i^{k-1}}$$

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

A3.2.3 Algorithme dans le *Code_Aster*

L'algorithme mis en oeuvre dans le *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 global ε_{glob}.

Pour chaque cycle k=1, $n\_\max_{jaco}$

Définir la tolérance dynamique : $\varepsilon_k = \left(\varepsilon_{dyn}\right)^k$,

$l = 0$,
Pour chaque ligne i= 1,n
Pour chaque colonne j= 1,n

Calcul des facteurs de couplage $f_{\mathbf{A}_{ij}^{k,l}} = \frac{\left|\mathbf{A}_{ij}^{k,l}\right|}{\sqrt{\left|\mathbf{A}_{ii}^{k,l}\mathbf{A}_{jj}^{k,l}\right|}}$ $f_{\mathbf{B}_{ij}^{k,l}} = \frac{\left|\mathbf{B}_{ij}^{k,l}\right|}{\sqrt{\left|\mathbf{B}_{ii}^{k,l}\mathbf{B}_{jj}^{k,l}\right|}}$ $i \neq j$,

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 $f_\lambda = \underset{i}{Max} \frac{\left|\lambda_i^k - \lambda_i^{k-1}\right|}{\lambda_i^{k-1}}$.

Calcul des facteurs de couplage globaux $f_{\mathbf{A}} = \underset{\substack{i,j \\ i\neq j}}{Max} \frac{\left|\mathbf{A}_{ij}^{k,l}\right|}{\sqrt{\left|\mathbf{A}_{ii}^{k,l}\mathbf{A}_{jj}^{k,l}\right|}}$ $f_{\mathbf{B}} = \underset{\substack{i,j \\ i\neq j}}{Max} \frac{\left|\mathbf{B}_{ij}^{k,l}\right|}{\sqrt{\left|\mathbf{B}_{ii}^{k,l}\mathbf{B}_{jj}^{k,l}\right|}}$

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 3.2 : Méthode de Jacobi implantée dans le Code_Aster

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

Remarque :

*Les matrices **A** et **B** étant symétriques, seulement la moitié des termes est stockée sous forme d'un vecteur.*

A3.2.4 Matrice de rotation

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 :

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

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 :

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

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 :

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

sinon :

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

avec :

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

$$c_3 = \mathbf{A}_{ii}^{k,l}\,\mathbf{B}_{jj}^{k,l} - \mathbf{B}_{ii}^{k,l}\,\mathbf{A}_{jj}^{k,l} \quad d = \frac{c_3}{2} + sign(c_3)\sqrt{\left(\frac{c_3}{2}\right)^2 + c_1\,c_2}$$

Remarques :

- *si $d = 0$ alors on est dans le cas proportionnel,*
- *si la matrice $\boldsymbol{B}$ est définie positive alors $\left(\frac{c_3}{2}\right)^2 + c_1\,c_2$ est positif [bib11] ce qui donne bien un sens au paramètre d .*

A3.2.5 Récapitulatif du paramètrage

Pour accéder aux paramètres qui interviennent dans la méthode de Jacobi, l'utilisateur doit sélectionner les mots-clés suivants :

- ε_{dyn} mot-clé `PREC_JACOBI` sous le mot-clé facteur `CALC_FREQ`,
- $n\_\max_{jaco}$ mot-clé `NMAX_ITER_JACOBI` sous le mot-clé facteur `CALC_FREQ`.

La précision globale ε_{glob} est égale à la précision de convergence demandée dans la méthode de Bathe et Wilson. On a donc $\varepsilon_{glob} = \varepsilon_{bath}$ (cf. [§7]).