



Calcul de fonctions de matrices

Date de publication :
10 octobre 2014

Cet article est issu de : **Sciences fondamentales | Mathématiques**

par **Gérard MEURANT**

Mots-clés
fonctions de matrices | logiciel mathématique | calcul scientifique | ingénierie mathématique

Résumé On rappelle les définitions d'une fonction $f(A)$ d'une matrice carrée à coefficients réels ou complexes. On présente ensuite des méthodes numériques permettant de calculer les éléments de $f(A)$, l'action sur un vecteur donné $f(A)v$ ou des formes bilinéaires $u^T f(A)v$ où interviennent deux vecteurs.

Keywords
matrix functions | mathematical software | scientific computing | engineering mathematics

Abstract We first recall what is a matrix function $f(A)$ of a square matrix with real or complex entries. Then we describe numerical methods to compute all the entries of $f(A)$, the action on a given vector $f(A)v$ or bilinear forms $u^T f(A)v$ with two given vectors.

Pour toute question :
Service Relation clientèle
Techniques de l'Ingénieur
Immeuble Pleyad 1
39, boulevard Ornano
93288 Saint-Denis Cedex

Document téléchargé le : **06/04/2019**
Pour le compte : **7200043660 - centralesupelec // 195.221.160.3**

Par mail :
infos.clients@teching.com
Par téléphone :
00 33 (0)1 53 35 20 20

Calcul de fonctions de matrices

par Gérard MEURANT

Ancien directeur de recherche au CEA

1. Définitions et propriétés de $f(A)$	AF 486 – 2
1.1 Exemples de fonctions de matrices.....	— 2
1.2 Définitions de $f(A)$	— 2
1.2.1 Forme canonique de Jordan	— 3
1.2.2 Interpolation polynomiale	— 3
1.2.3 Intégrale de Cauchy	— 3
1.3 Propriétés de $f(A)$	— 3
2. Méthodes de calcul de $f(A)$.....	4
2.1 Méthodes d'approximation.....	— 4
2.2 Méthodes de factorisation	— 4
2.3 Exponentielle	— 5
2.4 Logarithme	— 5
2.5 Racine carrée	— 6
2.6 Fonction signe	— 6
2.7 Prétraitements	— 6
2.8 Logiciels	— 7
2.9 Exemples numériques.....	— 7
3. Méthodes pour $f(A)v$	8
3.1 Méthodes directes	— 8
3.2 Méthodes de Krylov	— 8
3.3 Logiciels	— 8
3.4 Exemples numériques.....	— 8
4. Méthodes pour $u^T f(A)v$	9
4.1 Logiciels	— 10
4.2 Exemples numériques.....	— 10
5. Conclusion.....	— 10
Pour en savoir plus.....	Doc. AF 486

Cet article est consacré au calcul de fonctions de matrices. Avant de définir ce qu'elles sont, expliquons brièvement ce qu'elles ne sont pas. Supposons que l'on ait une fonction f suffisamment régulière et une matrice carrée A d'ordre n à coefficients $a_{i,j}$, réels ou complexes. La matrice $f(A)$ d'ordre n n'est pas la matrice dont les éléments sont $f(a_{i,j})$, auquel cas le calcul serait trivial. Les définitions de $f(A)$ rappelées ci-dessous visent à reproduire, pour une matrice, la plupart des propriétés des fonctions scalaires. Dans une première partie, on présentera les définitions et les principales méthodes de calcul de tous les éléments de $f(A)$. Cette partie est inspirée du livre [13] qui contient l'état de l'art concernant le calcul de $f(A)$, encore que certaines des méthodes décrites aient été légèrement améliorées depuis la parution de ce livre. On pourra également consulter [9] avec profit.

Les algorithmes pour $f(A)$ visent à calculer les n^2 éléments de la matrice. On utilise souvent des méthodes basées sur des factorisations de la matrice A à l'aide de transformations orthogonales et/ou des approximations de la fonction f permettant un calcul plus facile, par exemple des polynômes ou des fractions rationnelles. Les algorithmes correspondants ont donc un coût proportionnel à n^3 . Il n'est donc pas faisable, même avec les ordinateurs puissants dont on dispose aujourd'hui, de calculer $f(A)$ pour des matrices de très grande taille. Il se trouve que de nombreuses applications n'ont besoin que de calculer $f(A)v$ où v est un vecteur donné. Ceci peut être fait, sans calculer

explicitelement tous les éléments de $f(A)$, à l'aide de méthodes itératives de Krylov qui peuvent s'appliquer à de très grandes matrices creuses et que nous décrirons dans une deuxième partie.

Enfin, il existe d'autres applications pour lesquelles on n'a besoin que de calculer des scalaires $u^T f(A)v$, u et v étant des vecteurs donnés. Les méthodes pour calculer efficacement des bornes ou des approximations de ces quantités seront présentées dans une troisième et dernière partie.

1. Définitions et propriétés de $f(A)$

1.1 Exemples de fonctions de matrices

Il existe de nombreuses applications dans lesquelles interviennent des fonctions de matrices. On manipule souvent des fonctions de matrices sans le savoir. Par exemple, lorsqu'il existe, l'inverse A^{-1} de A correspond à la fonction telle que $f(x) = 1/x$. Résoudre le système linéaire $Ax = b$ est donc appliquer implicitement $f(A) = A^{-1}$ au vecteur b .

Un autre exemple simple est la résolution de systèmes d'équations différentielles linéaires qui fait intervenir la fonction exponentielle. On veut calculer y , solution de l'équation

$$\frac{dy}{dt} = Ay, \quad y(0) = c,$$

où A est une matrice et c un vecteur qui sont donnés. La solution est évidemment $y(t) = e^{tA}c$. Ici, on n'a besoin que de l'application de la fonction e^{tA} à un vecteur c , mais, dans certaines applications, il faut résoudre des équations différentielles dont l'inconnue est une matrice,

$$\frac{dA}{dt} = BA, \quad y(0) = I,$$

où B est une matrice connue et I la matrice identité. Les fonctions de matrices interviennent aussi dans la résolution de certaines équations matricielles. Par exemple, si avec A et B données, l'on veut calculer X solution de $XAX = B$, la solution est $X = B(AB)^{-1/2}$. Il faut donc calculer l'inverse de la racine carrée de AB (lorsqu'elle existe). De nombreuses autres applications sont décrites dans [13].

Les formes bilinéaires $u^T f(A)v$ (ou quadratiques lorsque $u = v$) sont elles aussi présentes dans de nombreuses applications. Par exemple, considérons la résolution d'un système linéaire $Ax = b$ et supposons que l'on ait une solution approchée \hat{x} . Le résidu est $r = b - A\hat{x}$ et l'erreur $e = x - \hat{x}$. Il est facile de voir que $Ae = r$. Si l'on veut calculer une valeur approchée de la norme de l'erreur, sachant que celle-ci n'est évidemment pas connue, on a :

$$\|e\|^2 = e^T e = r^T A^{-2} r.$$

On peut obtenir des approximations de cette quantité sans avoir à calculer le carré de l'inverse de A . Une autre application, plus actuelle, a trait à la modélisation et l'étude des propriétés des réseaux.

Un réseau, par exemple les réseaux sociaux, Internet ou d'autres, peut se modéliser par un graphe comprenant des nœuds et des arêtes qui les relient. On veut souvent savoir quelle est l'importance d'un nœud et quels sont les nœuds les plus importants. Cela ne se mesure pas uniquement au nombre de voisins. On veut

compter approximativement le nombre de chemins dans le graphe qui commencent et finissent au nœud considéré. La centralité d'un nœud i est définie par $[e^A]_{i,i}$ où A est la matrice d'adjacence du graphe, c'est-à-dire telle que $a_{ij} = 1$ si i et j sont reliés par une arête et 0 sinon. La communicabilité entre deux nœuds i et j est $[e^A]_{i,j}$ (voir [8]). On peut également utiliser d'autres fonctions que l'exponentielle, par exemple $(A - \alpha I)^{-1}$. Dans les exemples pratiques, A est une très grande matrice creuse et l'on ne peut donc pas calculer tous les éléments de son exponentielle, mais l'on a $[e^A]_{i,j} = e_i^T e^A e_j$, où e_i est la i -ème colonne de la matrice identité et, donc, le problème se ramène au calcul d'une approximation d'une forme bilinéaire ou quadratique.

Les fonctions qui se rencontrent le plus souvent dans les applications sont l'exponentielle, le logarithme, la racine carrée, les racines p -èmes et les fonctions trigonométriques. Une fonction utile dans certains problèmes de physique (en particulier la chromodynamique quantique) est la fonction signe.

1.2 Définitions de $f(A)$

Dans la suite de cet article on suppose que A est une matrice carree d'ordre n à coefficients réels ou complexes, c'est-à-dire $A \in \mathbb{C}^{n \times n}$. La fonction f est généralement supposée suffisamment continûment différentiable. Il est facile de définir $f(A)$ si f est un polynôme $p(x) = \sum_{j=0}^q \alpha_j x^j$. On a alors de façon naturelle

$$f(A) = p(A) = \sum_{j=0}^q \alpha_j A^j.$$

Certaines fonctions de matrice peuvent être définies facilement à l'aide de leur série. Par exemple, la série correspondant à l'exponentielle ayant un rayon de convergence infini, on peut définir l'exponentielle d'une matrice quelconque A par :

$$e^A = I + A + \frac{1}{2!} A^2 + \frac{1}{3!} A^3 + \dots$$

où I est la matrice identité d'ordre n . Mais ce type de définition nécessite parfois de restreindre l'ensemble de définition de A pour que la série soit convergente. Par exemple,

$$\log(I + A) = A - \frac{1}{2} A^2 + \frac{1}{3} A^3 - \frac{1}{4} A^4 + \dots, \quad \rho(A) < 1$$

où $\rho(A)$ est le rayon spectral de A , c'est-à-dire le maximum des modules des valeurs propres. Pour utiliser une série pour définir et calculer $f(A)$, il faut que le rayon de convergence de la série soit supérieur au rayon spectral de la matrice.

Des définitions plus générales sont listées ci-après. Pour un historique des définitions d'une fonction de matrice, voir [13] ou [16]. On pourra consulter également [14] et [15]. Pour la suite il est utile d'introduire la forme canonique de Jordan de la matrice A . Il existe

une matrice non singulière Z telle que $Z^{-1}AZ = J = \text{diag}(J_1, \dots, J_p)$ où J est une matrice diagonale par blocs et le k -ième bloc diagonal a la forme suivante :

$$J_k = \begin{pmatrix} \lambda_k & & 1 & & \\ & \lambda_k & & \ddots & \\ & & \ddots & & 1 \\ & & & & \lambda_k \end{pmatrix} \in \mathbb{C}^{m_k \times m_k},$$

où $\lambda_1, \dots, \lambda_s$ sont les valeurs propres distinctes de A (avec $s \leq p$) et $m_1 + \dots + m_p = n$. On note n_i la dimension du plus grand bloc de Jordan où apparaît λ_i . On dit que f est définie sur le spectre $\sigma(A)$ de A si les dérivées

$$f^{(j)}(\lambda_i), \quad j = 0, \dots, n_i - 1, \quad i = 1, \dots, s$$

existent. Notons que cette définition ne dit rien des valeurs ou des propriétés de f ou de ses dérivées en dehors du spectre de A .

1.2.1 Forme canonique de Jordan

Si f est définie sur le spectre de A au sens vu précédemment, la fonction de matrice $f(A)$ est égale à $Zf(J)Z^{-1}$ où $f(J)$ est une matrice bloc diagonale avec des blocs $f(J_k)$, faisant intervenir les dérivées de f , qui sont définis par :

$$f(J_k) = \begin{pmatrix} f(\lambda_k) & f'(\lambda_k) & \cdots & \frac{1}{(m_k - 1)!} f^{(m_k - 1)}(\lambda_k) \\ & f(\lambda_k) & \ddots & \vdots \\ & & \ddots & f'(\lambda_k) \\ & & & f(\lambda_k) \end{pmatrix}.$$

Dans la forme canonique de Jordan la matrice Z n'est pas nécessairement unique, mais on peut montrer que la définition précédente est indépendante du choix qui est fait. Cette définition peut être justifiée en considérant le développement de Taylor de f .

La définition de $f(A)$ à partir de la forme de Jordan présente surtout un intérêt théorique en permettant de démontrer certaines propriétés de $f(A)$ car il est bien connu que cette forme est très sensible aux perturbations de A . De petites variations de la matrice peuvent changer la taille des blocs. Pour ces raisons, il est quasiment impossible de calculer la forme de Jordan d'une matrice en arithmétique flottante à précision finie.

Il y a néanmoins un cas pour lequel cette définition est utile. Si la matrice A est diagonalisable, tous les blocs de Jordan sont d'ordre 1 et l'on a $A = ZDZ^{-1}$ où D est la matrice diagonale des valeurs propres de A . On a alors $f(A) = Zf(D)Z^{-1}$ et $f(D)$ est une matrice diagonale dont les éléments diagonaux sont les valeurs $f(\lambda_i)$, qui sont donc les valeurs propres de $f(A)$. Le cas le plus intéressant numériquement, pour des raisons de stabilité, est d'avoir A normale, c'est-à-dire telle que $A^*A = AA^*$ où A^* est la conjuguée transposée de A . Dans ce cas $A = ZDZ^*$ où Z est une matrice unitaire ($Z^*Z = I$, $\|Z\| = 1$) et l'on a $f(A) = Zf(D)Z^*$. C'est, par exemple, le cas si A est hermitienne ($A = A^*$) ou réelle et symétrique ($A = A^T$).

Il convient d'être prudent avec ce qu'on peut appeler des fonctions multivaluées. Par exemple la racine carrée et le logarithme. Si l'on a $x^2 = a$, on a évidemment $x = \pm \sqrt{a}$. Dans toutes les définitions de $f(A)$ il faut donc utiliser la même branche de la fonction. Par exemple dans le cas de la racine carrée, on utilisera le signe + pour tous les blocs de Jordan contenant la même valeur propre. Dans ce cas, on parle de **fonctions primaires**. Si l'on voulait calculer les racines carrées de la matrice identité d'ordre 2, c'est-à-dire X telle que $X^2 = I$, avec la définition précédente on trouverait $X = \pm I$. Mais, les deux matrices

$$\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

sont des racines carrées **non primaires**. En fait, il existe une infinité de racines carrées ; les matrices de rotations

$$\begin{pmatrix} \cos(\theta) & \sin(\theta) \\ \sin(\theta) & -\cos(\theta) \end{pmatrix}$$

pour $\theta \in [0, 2\pi]$ sont toutes des racines carrées de la matrice identité d'ordre 2.

1.2.2 Interpolation polynomiale

Si f est définie sur le spectre de A , on peut définir $f(A)$ à l'aide du polynôme d'interpolation de Hermite, c'est-à-dire un polynôme interpolant la fonction et ses dérivées. On a $f(A) = p(A)$ où p est un polynôme de degré inférieur ou égal à $\sum_{i=1}^s n_i$ (qui est le degré du polynôme minimal de A) satisfaisant les conditions :

$$p^{(j)}(\lambda_i) = f^{(j)}(\lambda_i), \quad j = 0, \dots, n_i - 1, \quad i = 1, \dots, s$$

On a vu précédemment qu'il est facile de définir $p(A)$. Cependant, de même que pour la définition à partir de la forme de Jordan, l'interpolation d'Hermite ne fournit pas toujours un algorithme fiable pour calculer une fonction de matrice. Notons qu'il faut connaître le spectre de A , calculer les coefficients du polynôme de façon stable et ensuite évaluer correctement le polynôme, ce qui n'est pas toujours facile en arithmétique flottante.

1.2.3 Intégrale de Cauchy

La définition la plus concise et élégante d'une fonction de matrice utilise une intégrale de Cauchy. Si f est une fonction analytique sur un ouvert Ω et $\Gamma \subset \Omega$ est une courbe fermée qui entoure le spectre de A dans le plan complexe, on peut définir $f(A)$ par :

$$f(A) = \frac{1}{2\pi i} \int_{\Gamma} f(z)(zI - A)^{-1} dz$$

avec $i^2 = -1$. Rappelons qu'une fonction est analytique si elle est développable en série entière au voisinage de chaque point de son ensemble ouvert de définition. Cette définition a été utilisée dans [12] pour calculer certaines fonctions de matrice à l'aide de transformations conformes et de formules de quadrature.

Les définitions de $f(A)$ par la forme canonique de Jordan et par l'interpolation d'Hermite sont équivalentes. Si f est analytique, la définition par l'intégrale de Cauchy est équivalente aux deux autres.

1.3 Propriétés de $f(A)$

Les définitions précédentes de $f(A)$ ont été introduites afin qu'une fonction de matrice possède la plupart des propriétés d'une fonction définie sur les nombres réels ou complexes. Il y a cependant certaines propriétés qui pourraient sembler naturelles et qui ne sont pas vraies pour toute matrice. Il convient donc d'être prudent.

Les propriétés qui sont toujours vraies pour des fonctions définies sur le spectre de A sont les suivantes :

- $(f + g)(A) = f(A) + g(A)$;
- $(f \cdot g)(A) = f(A)g(A)$;
- si f est constante, $f(x) = \alpha \in \mathbb{C}$, alors $f(A) = \alpha I$;
- $f(A)$ commute avec A ;
- si B commute avec A , B commute avec $f(A)$;
- $f(A^T) = f(A)^T$;
- les valeurs propres de $f(A)$ sont $f(\lambda_i)$;
- si X est non singulière, $f(XAX^{-1}) = Xf(A)X^{-1}$;
- si A est diagonale par bloc avec des blocs diagonaux $A_{i,i}$, $f(A)$ est bloc diagonale avec des blocs diagonaux $f(A_{i,i})$;
- si A est triangulaire par bloc avec des blocs diagonaux $A_{i,i}$, $f(A)$ est triangulaire par bloc avec la même structure et $[f(A)]_{i,i} = f(A_{i,i})$;

- si f est une fraction rationnelle

$$f(x) = c \frac{(x - \alpha_1) \cdots (x - \alpha_n)}{(x - \beta_1) \cdots (x - \beta_k)}$$

dont les pôles β_i n'appartiennent pas au spectre de A , on a

$$f(A) = c(A - \alpha_1 I) \cdots (A - \alpha_n I)(A - \beta_1 I)^{-1} \cdots (A - \beta_k I)^{-1}$$

- les relations entre fonctions se généralisent. Si $P(f_1, \dots, f_m) = 0$ où P est un polynôme, alors $P(f_1(A), \dots, f_m(A)) = 0$. Par exemple on a $\sin^2(A) + \cos^2(A) = I$ et $e^{iA} = \cos(A) + i \sin(A)$.

Pour définir la composition de fonctions $(f \circ g)(A)$ il faut prendre quelques précautions. Soit g définie sur le spectre de A ; on suppose que les valeurs $f^{(j)}(g(\lambda_j))$ existent pour $j = 0, \dots, n_i$ et $i = 1, \dots, s$. Alors on a $(f \circ g)(A) = f(g(A))$.

Voyons à présent quelques propriétés qui ne sont pas « naturelles » :

- $f(A^*) \neq f(A)^*$;
- on a $e^{\log(A)} = A$, mais pas nécessairement $\log(e^A) = A$;
- $(AB)^{1/2} \neq A^{1/2}B^{1/2}$;
- $e^{A+B} \neq e^A e^B$.

2. Méthodes de calcul de $f(A)$

Il existe de nombreuses méthodes de calcul pour les fonctions de matrices. Certaines sont générales et d'autres destinées à des fonctions particulières. Les méthodes les plus générales utilisent des approximations de la fonction f et/ou des factorisations de la matrice A .

Notons qu'une fonction de matrice d'une matrice réelle n'est pas toujours réelle. Cependant, si faire se peut, on veut éviter d'utiliser l'arithmétique complexe si A est réelle.

La plupart des algorithmes décrits ci-dessous visent à obtenir une bonne précision du résultat avec un coût le plus faible possible.

2.1 Méthodes d'approximation

Si on connaît un développement de Taylor de f au voisinage de a ,

$$f(x) = \sum_{j=0}^{\infty} \alpha_j (x - a)^j, \quad \alpha_j = \frac{f^{(j)}(a)}{j!}$$

on a :

$$f(A) = \sum_{j=0}^{\infty} \alpha_j (A - aI)^j.$$

Il faut, bien entendu, que l'on ait $|\lambda_i - a| < R$ où R est le rayon de convergence de la série de f . Pour calculer $f(A)$ on peut tronquer le développement en série. On connaît des bornes du reste (voir [13]),

$$\left\| f(A) - \sum_{i=0}^{s-1} \alpha_j (A - aI)^j \right\| \leq \frac{1}{s!} \max_{0 \leq t \leq 1} \left\| (A - aI)^s f^{(s)}(aI + t(A - aI)) \right\|.$$

Il reste qu'il faut calculer les puissances de $A - aI$ et le polynôme le plus efficacement et rapidement possible. Il existe malheureusement des possibilités de « cancellation » et de croissance des erreurs d'arrondi dans l'évaluation du polynôme. Néanmoins, si l'on n'a pas besoin d'une très grande précision, on peut utiliser cette méthode pour des fonctions comme l'exponentielle ou pour des fonctions trigonométriques sin, cos, etc.

On préfère généralement utiliser des approximations par fraction rationnelle, en particulier des approximants de Padé, voir [5] [AF 1 390]. Pour un approximant de Padé au voisinage de 0, on a $f(x) - r_{k,m}(x) = O(x^{k+m+1})$ avec $r_{k,m}(x) = p_{k,m}(x) / q_{k,m}(x)$ où $p_{k,m}$ et $q_{k,m}$ sont deux polynômes.

2.2 Méthodes de factorisation

Le cas le plus facile à traiter est celui d'une matrice normale (telle que $A^*A = AA^*$), en particulier une matrice hermitienne ou symétrique. On a $f(A) = Zf(D)Z^*$ et Z est une matrice unitaire, $Z^*Z = I$. La diagonale de D contient les valeurs propres de A et il suffit d'évaluer la fonction f pour ces valeurs. Dans le cas où la matrice n'est que diagonalisable, c'est-à-dire $A = ZDZ^{-1}$, il est parfois plus délicat de calculer $f(A)$ de cette façon car la matrice Z peut avoir un nombre de conditionnement grand, ce qui peut donner un résultat avec une précision insuffisante.

Si la matrice n'est pas diagonalisable, comme nous l'avons déjà dit, il n'est pas possible d'utiliser de façon fiable la forme canonique de Jordan. La forme la plus simple que l'on puisse obtenir à l'aide de transformations orthogonales (ou unitaires) est la forme triangulaire. Pour toute matrice A il existe une matrice unitaire Z telle que $Z^*AZ = T$ est une matrice triangulaire supérieure. La matrice ZT^*Z s'appelle la **forme de Schur** de la matrice A . Les éléments diagonaux de T sont les valeurs propres de A , la matrice T peut donc avoir des éléments complexes même si A est réelle.

Si l'on veut exploiter ce résultat, il faut pouvoir calculer $f(T)$. On sait déjà que cette matrice est triangulaire supérieure et que les éléments diagonaux sont $f(t_{i,i})$ si l'on note $t_{i,j}$ les éléments de T . Il existe une formule exacte donnant les éléments non diagonaux (voir [13] page 84). Cependant cette formule conduit à un nombre d'opérations trop important. B.N. Parlett a remarqué en 1976 que, puisque l'on connaît les éléments diagonaux, les autres pouvaient être calculés en utilisant le fait que T commute avec $f(T)$. On écrit l'élément (i, j) de la relation $Tf(T) = f(T)T$ en notant $f_{i,j}$ les éléments de $f(T)$ et on obtient :

$$f_{i,j} = t_{i,i} \frac{f_{i,i} - f_{j,j}}{t_{i,i} - t_{j,j}} + \sum_{k=i+1}^{j-1} \frac{f_{i,k} t_{k,j} - f_{k,j} t_{i,k}}{t_{i,i} - t_{j,j}}, \quad i < j.$$

Cette formule permet de calculer $f(T)$ colonne par colonne ou diagonale par diagonale. Mais l'on voit qu'il faut que $t_{i,i} \neq t_{j,j}$. Puisque les éléments diagonaux de T sont les valeurs propres de A , il faut donc que A n'ait pas de valeur propre multiple si l'on veut calculer tous les éléments de $f(A)$ de cette façon. Il est également clair que si l'on a des valeurs propres très proches, on risque d'avoir des problèmes numériques puisque l'on divisera par une valeur petite. Cette méthode n'est donc pas vraiment utilisable de façon générale. Un moyen d'améliorer les choses est de considérer une forme de Schur par blocs. La matrice T est alors triangulaire supérieure par bloc avec des blocs $T_{i,j}$. La relation de commutativité s'écrit alors :

$$T_{i,i}F_{i,j} - F_{i,j}T_{j,j} = F_{i,i}T_{i,j} - T_{i,j}F_{j,j} + \sum_{k=i+1}^{j-1} (F_{i,k}T_{k,j} - T_{i,k}F_{k,j}), \quad i < j$$

De plus la matrice T peut alors être réelle si A est réelle. Il faut donc d'abord calculer les blocs diagonaux $F_{i,i}$, ce qui peut être fait avec une méthode d'approximation, série de Taylor ou approximant de Padé. Ensuite, pour les blocs non diagonaux, il faut résoudre des équations de Sylvester qui sont de la forme $S_1X - XS_2 = S_3$ où les S_i , $i = 1, 2, 3$ sont connues. Pour qu'il existe une solution, il faut que les matrices $T_{i,i}$ et $T_{j,j}$ n'aient pas de valeur propre en commun. Les blocs sont donc choisis après permutation tels que les valeurs propres correspondant à un bloc soient proches et que celles de deux blocs distincts soient suffisamment séparées. Pour un algorithme heuristique de partitionnement de T en blocs (voir [13] chapitre 9). Cette méthode de calcul de $f(A)$ est dénommée **algorithme de Schur-Parlett**.

Les équations de Sylvester se résolvent en calculant la forme de Schur complexe de S_1 et S_2 , en transformant S_3 à l'aide des transformations unitaires calculées pour la forme de Schur, en calculant une solution de l'équation transformée par substitution et enfin en appliquant les inverses des transformations unitaires.

2.3 Exponentielle

On a vu que e^A est définie par sa série entière. Une des méthodes les plus utilisées pour calculer l'exponentielle d'une matrice A est basée sur l'identité $e^A = (e^{A/\alpha})^\alpha$ pour $\alpha = 2^s$ (cette égalité n'étant pas vraie pour tout $\alpha \in \mathbb{C}$) et sur le fait qu'un approximant de Padé de la fonction exponentielle à l'origine donne de bons résultats si la norme de la matrice est petite. On veut choisir s de telle sorte que A/α ait une petite norme et que l'on ait une erreur inverse relative petite, proche de la précision machine. La méthode comporte donc d'abord une mise à l'échelle, le calcul d'une approximation et enfin un calcul de la puissance 2^s du résultat. Donnons quelques détails d'après [13].

Les approximants de Padé de l'exponentielle sont connus explicitement. On a :

$$e^x - r_{k,m}(x) = \mathcal{O}(x^{k+m+1}) \text{ avec } r_{k,m}(x) = p_{k,m}(x)/q_{k,m}(x)$$

et

$$\begin{aligned} p_{k,m}(x) &= \sum_{j=0}^k \frac{(k+m-j)! k!}{(k+m)! (k-j)!} \frac{x^j}{j!}, \\ q_{k,m}(x) &= \sum_{j=0}^m \frac{(k+m-j)! m!}{(k+m)! (k-j)!} \frac{(-x)^j}{j!}. \end{aligned}$$

Le reste peut s'écrire :

$$e^x - r_{k,m}(x) = (-1)^m \frac{k! m!}{(k+m)! (k+m+1)!} x^{k+m+1} + \mathcal{O}(x^{k+m+2}).$$

On choisit $k = m$ et l'on note $r_m = r_{m,m}$. On a $e^{-2^{-s}A} r_m(2^{-s}A) = I + G$ avec $\|G\| < 1$. Il vient alors $r_m(2^{-s}A)^{2^s} = e^{A+E}$, ce qui montre que l'approximation de la solution est la solution exacte d'un problème perturbé. Ensuite on borne l'erreur inverse :

$$\frac{\|E\|}{\|A\|} \leq -\frac{\log(1-\|G\|)}{\|2^{-s}A\|}.$$

Il faut ensuite borner la norme de G . D'après les propriétés des approximants de Padé on a :

$$e^{-x} r_m(x) - 1 = \sum_{j=2m+1}^{\infty} c_j x^j.$$

On calcule en précision étendue un certain nombre de coefficients c_j . Ensuite, on borne la norme de G :

$$\|G\| \leq \sum_{j=2m+1}^{\infty} |c_j| \theta^j, \quad \theta = \|2^{-s}A\|.$$

Pour déterminer m on termine en obtenant, pour des petites valeurs de m , la plus grande valeur θ_m telle que la borne de l'erreur relative ne soit pas plus grande que la précision machine $u = 1,1 \cdot 10^{-16}$. Les valeurs possibles de m sont $m = 1, 2, 3, 5, 7, 9, 13, 17, 21$. Le choix se fait ensuite en considérant le nombre d'opérations nécessaires pour calculer $r_m(A)$ de la façon la plus efficace possible. Cela conduit à choisir $m = 13$ avec la valeur $\theta_m = 5,4$ (qui ne dépend pas de A). On a $\lceil \log_2(\|A/\theta_m\|) \rceil$ si $\|A\| \geq \theta_m$ et $s = 0$ sinon. L'algorithme proposé dans [13] commence par tester si la 1-norme

de A est inférieure à θ_m pour l'un des indices $m = 3, 5, 7, 9$, auquel cas on calcule $r_m(A)$ sans faire de mise à l'échelle. Sinon, on choisit le plus petit entier s tel que $\|2^{-s}A\|_1 \leq \theta_{13}$, on calcule l'approximant de Padé $r_{13}(2^{-s}A)$ et enfin on obtient l'approximation de e^A qui est $r_{13}(2^{-s}A)^{2^s}$. L'approximant de Padé est évalué en remarquant que $q_{m,m}(x) = p_{m,m}(-x)$ et en essayant d'utiliser le moins possible de multiplications par A . Cette méthode a été utilisée dans la fonction Matlab expm. Toutefois, il a été remarqué que, dans certains cas, elle conduisait à choisir des valeurs de s grandes et donnait des résultats peu précis, alors que des valeurs de s plus petites donnaient de bien meilleurs résultats. Ce problème a été étudié et un remède proposé dans [2]. Pour borner la norme de G on utilise :

$$\|G\| \leq \sum_{j=2m+1}^{\infty} |c_j| \left(\|2^{-s}A\|^t \right)^{1/t},$$

où t est l'indice qui donne le maximum de $\|2^{-s}A\|^t$ pour $j \geq 2m+1$. Ces bornes plus précises donnent des valeurs de s plus petites et de meilleurs résultats (voir [2]).

On peut également utiliser la méthode de Schur-Parlett par blocs avec des blocs diagonaux d'ordre 1 ou 2. Pour les blocs d'ordre 2, on connaît explicitement la solution :

$$\exp\begin{pmatrix} \lambda_1 & \alpha \\ 0 & \lambda_2 \end{pmatrix} = \begin{pmatrix} e^{\lambda_1} & \alpha e^{(\lambda_1+\lambda_2)/2} \sinh(\lambda_1-\lambda_2)/2 \\ 0 & e^{\lambda_2} \end{pmatrix}.$$

mais il faut faire attention à l'évaluation de $\sinh(y)/y$ lorsque y est proche de 0.

Pour l'intérêt historique de ces articles, on pourra aussi consulter [18] et [19].

2.4 Logarithme

Un logarithme de A est une matrice X telle que $e^X = A$. Une matrice non singulière A a une infinité de logarithmes. En général, on suppose que A n'a pas de valeurs propres appartenant au demi-axe des réels négatifs \mathbb{R}^- et on note par \log le logarithme principal, c'est-à-dire tel que les valeurs propres de $\log(A)$ soient dans l'ensemble $\{z \mid -\pi < \operatorname{Im}(z) < \pi\}$. Concernant les propriétés du logarithme, on sait que si A vérifie l'hypothèse précédente, on a $\log(A^\alpha) = \alpha \log(A)$ pour $\alpha \in [-1, 1]$ et donc, $\log(A^{1/2}) = (1/2) \log(A)$.

De même que pour l'exponentielle, on utilise une mise à l'échelle basée sur la relation $\log(A) = 2^s \log(A^{1/2^s})$ et un approximant de Padé de la fonction $\log(1+x)$ qui s'écrit :

$$r_m(x) = \sum_{j=1}^m \frac{w_j^{(m)} x}{1 + t_j^{(m)} x}$$

où $w_j^{(m)}$ et $t_j^{(m)}$ sont les poids et les noeuds de la formule de quadrature de Gauss-Legendre sur $[0, 1]$ avec m noeuds.

Dans [13] on préfère appliquer la méthode à la matrice triangulaire T obtenue par factorisation de Schur $A = ZTZ^*$ car elle permet de calculer les racines carrées nécessaires avec l'algorithme de Schur décrit dans la section suivante. Les valeurs de m et s sont obtenues en bornant les normes de $I - A^{1/2^s}$. Les bornes utilisées dans [13] ont été améliorées dans [4]. Ces bornes conduisent à choisir $m \leq 7$ et à des valeurs de s parfois grandes, ce qui peut donner une moindre précision. Une fois que l'on a déterminé m et s , on calcule $U = r_m(T^{1/2^s} - I)$ à l'aide de la formule ci-dessus avec $T^{1/2^s}$ obtenue par application répétitive de la fonction racine carrée

et finalement $X = 2^s ZUZ^*$. La version décrite en détail dans [4] donne de meilleurs résultats que celle de [13].

Si l'on ne souhaite pas utiliser la factorisation de Schur, on doit calculer de façon répétitive des racines carrées de A , ce qui peut se faire avec les méthodes itératives décrites dans la section suivante.

2.5 Racine carrée

Une racine carrée de A est une matrice X telle que $X^2 = A$. Lorsque l'ordre de A est égal à 1, on sait qu'un nombre réel positif à deux racines carrées, l'une positive et l'autre négative. De même, la racine carrée d'une matrice n'est pas nécessairement unique et elle peut ne pas exister si A est singulière. Par exemple, la matrice

$$A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

n'a pas de racine carrée. Par contre, il existe des matrices ayant une infinité de racines carrées.

Cependant, on sait que si A n'a pas de valeurs propres appartenant au demi-axe des réels négatifs \mathbb{R}^- , alors il existe une racine carrée unique X dont les valeurs propres sont dans le demi-plan ouvert $\text{Re}(\lambda) > 0$. On la note $X = A^{1/2}$. De plus, si A est réelle, X est réelle et si A est hermitienne et définie positive, X l'est aussi. Si A est réelle sans valeurs propres réelles négatives, A a exactement 2^{rc} racines carrées primaires où r est le nombre de valeurs propres réelles distinctes et c le nombre de paires de valeurs propres complexes conjuguées.

Pour calculer $A^{1/2}$ de façon simple, on peut utiliser l'algorithme de Schur-Parlett ou une méthode itérative. En supposant A non singulière et en utilisant la décomposition de Schur $A = ZTZ^*$ avec Z triangulaire supérieure, il faut calculer Y triangulaire supérieure telle que $Y^2 = T$. Ceci conduit aux équations :

$$\begin{aligned} y_{i,i}^2 &= t_{i,i}, \\ (y_{i,i} + y_{j,j})y_{i,j} &= t_{i,j} - \sum_{k=i+1}^{j-1} y_{i,k}y_{k,j}, \end{aligned}$$

pour $i = 1, \dots, n$ et $j = 1, \dots, n$. On choisit toujours le même signe pour la racine carrée de $t_{i,i}$ et le facteur $y_{i,i} + y_{j,j}$ ne peut pas être nul car les $t_{i,i}$ et donc les $y_{i,i}$ ne sont pas nuls puisque A est non singulière. Cette méthode utilise de l'arithmétique complexe. Si A est réelle il peut être utile de travailler en arithmétique réelle, ce qui peut être fait avec une décomposition de Schur par bloc avec des blocs diagonaux 1×1 et 2×2 . On doit donc calculer la racine carrée de matrices réelles R d'ordre 2. Il existe deux racines carrées réelles :

$$R = \begin{pmatrix} r_{1,1} & r_{1,2} \\ r_{2,1} & r_{2,2} \end{pmatrix}, \quad R^{1/2} = \pm \begin{pmatrix} a + \frac{1}{4a}(r_{1,1} - r_{2,2}) & \frac{1}{2a}r_{1,2} \\ \frac{1}{2a}r_{2,1} & a - \frac{1}{4a}(r_{1,1} - r_{2,2}) \end{pmatrix},$$

où a est la partie réelle positive d'une racine carrée des valeurs propres de R qui sont une paire de nombres complexes conjugués. Ensuite, comme on l'a vu plus haut, on résout des équations de Sylvester pour calculer les blocs non diagonaux.

Il existe aussi de nombreuses méthodes itératives pour calculer la racine carrée. La méthode de Newton peut s'écrire après quelques manipulations :

$$X_{k+1} = \frac{1}{2}(X_k + X_k^{-1}A), \quad X_0 = A.$$

Le terme $X_k^{-1}A$ se calcule en résolvant n systèmes linéaires avec pour second membre les colonnes de A . On peut, en fait, partir de n'importe quelle matrice qui commute avec A . Cependant, la convergence n'est pas assurée si X_0 n'est pas assez proche de la solution.

Il existe des variantes de la méthode précédente. Si l'on pose $Y_k = A^{-1}X_k$, on obtient les itérations couplées :

$$\begin{aligned} X_{k+1} &= \frac{1}{2}(X_k + Y_k^{-1}), \quad X_0 = A, \\ Y_{k+1} &= \frac{1}{2}(Y_k + X_k^{-1}), \quad Y_0 = I. \end{aligned}$$

Cette méthode a été proposée par Denman et Beavers. Si elle converge, on a $\lim_{k \rightarrow \infty} X_k = A^{1/2}$ et $\lim_{k \rightarrow \infty} Y_k = A^{-1/2}$. Il existe encore d'autres variantes de la méthode de Newton ; par exemple :

$$\begin{aligned} X_{k+1} &= X_k + E_k, \quad X_0 = A, \\ E_{k+1} &= -\frac{1}{2}E_k X_{k+1}^{-1} E_k, \quad E_0 = \frac{1}{2}(I - A). \end{aligned}$$

Notons que toutes ces méthodes nécessitent le calcul de matrices inverses ou la résolution de systèmes linéaires avec des seconds membres multiples.

2.6 Fonction signe

La fonction signe scalaire est définie pour $z \in \mathbb{C}$ en dehors de l'axe imaginaire par :

$$S = \text{signe}(z) = \begin{cases} 1, & \text{Re}(z) > 0, \\ -1, & \text{Re}(z) < 0. \end{cases}$$

Pour une matrice, on suppose qu'il n'existe pas de valeurs propres imaginaires pures. Si la forme de Jordan de la matrice est $A = ZJZ^{-1}$, en regroupant en deux groupes les valeurs propres de parties réelles négatives et positives, on a :

$$\text{signe}(A) = Z \begin{pmatrix} -I_s & 0 \\ 0 & I_t \end{pmatrix} Z^{-1},$$

où I_s et I_t sont des matrices identités d'ordre s et t . La matrice S est telle que :

$$S^2 = I \text{ et } SA = AS.$$

Pour calculer la fonction signe on utilise le même type de méthodes que pour la racine carrée. Si on peut calculer la décomposition de Schur $A = ZTZ^*$ et si $S = \text{signe}(T)$, on a $\text{signe}(A) = ZSZ^*$. Le calcul de S se fait en utilisant $S^2 = I$ et $ST = TS$. Les éléments diagonaux de S sont égaux à ± 1 .

On peut également utiliser la méthode de Newton et ses variantes. La méthode de base s'écrit :

$$X_{k+1} = \frac{1}{2}(X_k + X_k^{-1}), \quad X_0 = A.$$

On élimine la nécessité de calculer des inverses avec la méthode de Newton-Schulz :

$$X_{k+1} = \frac{1}{2}X_k(3I - X_k^2), \quad X_0 = A.$$

2.7 Prétraitements

Dans certains cas il peut être utile d'appliquer des prétraitements à la matrice A avant de calculer la fonction. Par exemple, pour l'exponentielle on peut utiliser une translation $e^{A-\mu I} = e^{-\mu}e^A$. On peut également calculer une matrice diagonale D qui minimise $\|D^{-1}AD\|_F$ où l'on utilise la norme de Frobenius.

2.8 Logiciels

Il existe un certain nombre de logiciels permettant de calculer tous les éléments de fonctions de matrices. Matlab contient des fonctions calculant l'exponentielle (`expm`), le logarithme (`logm`) et la racine carrée (`sqrtm`) ainsi qu'une fonction permettant de calculer $f(A)$ pour une fonction définie par l'utilisateur (`funm`). Les méthodes utilisées ont varié au cours des versions du logiciel et de l'évolution de l'état de l'art, comme on va le voir ci-dessous. D'autre part, on peut se procurer gratuitement deux boîtes à outils permettant le calcul de certaines fonctions. L'une `mft_toolbox` est accessible sur le site Internet de Nick Higham (voir rubrique Sites Internet du *Pour en savoir plus*) et correspond aux méthodes décrites dans [13]. Les fonctions correspondant aux améliorations décrites dans [2] et [4] (`expm_new` et `logm_new`) sont aussi disponibles sur ce site. L'adresse Internet de l'autre boîte à outils, `expokit`, est donnée dans le *Pour en savoir plus* et concerne principalement la fonction exponentielle dans l'optique de la résolution de systèmes d'équations différentielles voir [22]. À cette même adresse on trouve des routines Fortran calculant les mêmes fonctions.

Des fonctions Matlab contenant des intégrateurs exponentiels pour les équations différentielles (logiciel `Expint`) sont aussi données.

Des fonctions calculant l'exponentielle, le logarithme et la racine carrée de matrices sont également présentes dans Octave et Scilab.

Mathematica contient des fonctions pour calculer des exponentielles et des logarithmes de matrice, éventuellement définies de façon symbolique. D'après la documentation, on utilise la série correspondante. Si la matrice est symbolique, on utilise la forme de Jordan.

Maple propose des fonctions calculant par interpolation polynomiale l'exponentielle et des puissances de matrices définies de façon symbolique.

On peut également utiliser des librairies commerciales pour d'autres langages de programmation. Par exemple, la bibliothèque NAG contient des routines Fortran et C pour calculer des fonctions de matrices basées sur les mêmes méthodes que celles utilisés dans Matlab.

2.9 Exemples numériques

Considérons deux exemples simples. La matrice

$$A = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 1 & 1 \end{pmatrix}$$

a pour exponentielle :

$$X = \begin{pmatrix} e^2 & 0 & 0 \\ 0 & e \cos(1) & -e \sin(1) \\ 0 & e \sin(1) & e \cos(1) \end{pmatrix}.$$

Utilisons tout d'abord une vieille version de Matlab, la version 7.1 (R14) datant de 2005. La matrice A a pour valeurs propres 2 et $1 \pm i$ et des vecteurs propres orthogonaux. On peut donc utiliser la définition $f(A) = Zf(D)Z^*$. L'erreur obtenue est :

$$Zf(D)Z^* - X = \begin{pmatrix} -1,77636 \cdot 10^{-15} & 0 & 0 \\ 0 & -2,22045 \cdot 10^{-16} & 4,44089 \cdot 10^{-16} \\ 0 & -4,44089 \cdot 10^{-16} & -2,22045 \cdot 10^{-16} \end{pmatrix}.$$

Si on utilise la fonction `expm` de cette version, on a :

$$X_{\text{expm}} - X = \begin{pmatrix} 1,77636 \cdot 10^{-15} & 0 & 0 \\ 0 & 1,11022 \cdot 10^{-15} & -1,33027 \cdot 10^{-15} \\ 0 & 1,33027 \cdot 10^{-15} & 1,11022 \cdot 10^{-15} \end{pmatrix}$$

ce qui donne un résultat moins précis. La fonction `expm_new` de [2] donne :

$$X_{\text{expm_new}} - X = \begin{pmatrix} 4,44089 \cdot 10^{-16} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & -1,11022 \cdot 10^{-16} & 0 \end{pmatrix}.$$

ce qui est sensiblement meilleur. Ce dernier résultat est aussi ce qui est obtenu avec la fonction standard `expm` de la version 7.11 plus récente de Matlab (2010).

On peut calculer le logarithme de X et, pour cette matrice, l'on doit retrouver A . On considère $\log(X) - A$. Avec Matlab 7.11, `logm` et `logm_new` donnent les valeurs suivantes :

$$\begin{pmatrix} 4,44089 \cdot 10^{-16} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & -1,11022 \cdot 10^{-16} & 0 \end{pmatrix}.$$

Prenons la racine carrée de X et calculons $\left[(X)^{1/2}\right]^2 - X$. Avec Matlab 7.11 et la fonction `sqrtm` on trouve :

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & -2,2204 \cdot 10^{-16} & 0 \\ 0 & -8,8818 \cdot 10^{-16} & -2,2204 \cdot 10^{-16} \end{pmatrix}.$$

La méthode de Newton telle que décrite plus haut (sans mise à l'échelle) converge en 7 itérations et donne :

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & -4,4409 \cdot 10^{-16} & 0 \\ 0 & 0 & -4,4409 \cdot 10^{-16} \end{pmatrix}.$$

Avec mise à l'échelle, on converge en 6 itérations (avec un résidu relatif plus petit que $3u$) mais l'erreur est légèrement plus grande. La méthode de Denman-Beavers converge en 6 itérations et donne la même erreur que ci-dessus.

Comme autre exemple considérons une matrice symétrique A d'ordre n dont tous les éléments sont nuls exceptés $a_{i,n-i+1} = 1$, $i = 1, \dots, n$. Les éléments non nuls sont situés sur l'antidiagonale principale. Supposons n pair, l'exponentielle de A est connue explicitement. Les éléments non nuls sont situés sur la diagonale, $[e^A]_{i,i} = \cosh(1)$, $i = 1, \dots, n$ et sur l'antidiagonale principale $[e^A]_{i,n-i+1} = \sinh(1)$, $i = 1, \dots, n$. On peut donc calculer la norme de l'exponentielle de A moins la solution exacte. Choisissons $n = 100$. Avec Matlab 7.1 et `expm` on trouve $1,4120 \cdot 10^{-15}$, `expm_new` donne $5,6878 \cdot 10^{-16}$. Avec Matlab 7.11, on obtient respectivement $7,0601 \cdot 10^{-16}$ et $5,6878 \cdot 10^{-16}$.

On calcule la norme de $\log(e^A) - A$ (qui n'est pas nulle $\forall A$). Avec Matlab 7.1 on trouve $4,4409 \cdot 10^{-16}$.

Pour la racine carrée, on a $\|A^{1/2}\|^2 - A\| = 6,6613 \cdot 10^{-16}$ lorsqu'on calcule avec `sqrtm` de Matlab 7.1. Les méthodes de Newton et de Denman-Beavers convergent en 6 et 5 itérations et donnent respectivement $6,6613 \cdot 10^{-16}$ et $1,5543 \cdot 10^{-15}$.

On voit donc que toutes ces méthodes donnent des résultats satisfaisants sur ces petits exemples.

3. Méthodes pour $f(A)v$

Dans cette section, on décrit des méthodes pour calculer l'application de $f(A)$ à un vecteur v sans calculer tous les éléments de $f(A)$.

3.1 Méthodes directes

Une méthode pour calculer $e^A v$ est décrite dans [3]. Cet article s'intéresse également à la résolution d'équations différentielles non linéaires. La méthode utilisée peut être qualifiée de directe car elle est basée sur les résultats de [2]. La différence est qu'au lieu d'utiliser un approximant de Padé de l'exponentielle, on se sert de la série de Taylor après avoir fait une mise à l'échelle. On a donc $e^A v \approx ((T_m(s^{-1}A))^s v)$ où T_m représente la troncature de la série de Taylor. Les valeurs de m et s sont calculées comme dans [2] en obtenant des bornes du reste et en minimisant l'erreur inverse et le coût.

On pourra également consulter [22] et [21] pour l'exponentielle.

3.2 Méthodes de Krylov

Pour une présentation succincte des méthodes de Krylov, voir [17] et [20] pour l'exponentielle. L'espace de Krylov d'ordre k construit sur A et v et noté $\mathcal{K}_k(A, v)$ est défini comme l'espace engendré par les vecteurs :

$$v, Av, A^2v, \dots, A^{k-1}v.$$

Pour développer des méthodes de Krylov, la première question qui se pose est de savoir comment choisir la base de l'espace $\mathcal{K}_k(A, v)$ pour un vecteur v donné. On choisit, pour des raisons de stabilité, de construire de façon incrémentale une base orthogonale de l'espace de Krylov. Ceci est fait par le procédé dit d'Arnoldi qui n'est pas autre chose que la méthode de Gram-Schmidt (voir [10]) appliquée à l'espace de Krylov. Lorsque la matrice est symétrique, on obtient l'algorithme de Lanczos.

On construit les vecteurs de base orthonormaux v_j de $\mathcal{K}_k(A, v)$ et une matrice H_k d'éléments $h_{i,j}$ de la façon suivante :

$$v_1 = v / \|v\|.$$

L'algorithme pour calculer le vecteur v_{j+1} est

$$h_{i,j} = (Av_j, v_i), \quad i = 1, \dots, j,$$

$$\bar{v}_j = Av_j - \sum_{i=1}^j h_{i,j}v_i,$$

$$h_{j+1,j} = \|\bar{v}_j\|, \quad v_j = \frac{\bar{v}_j}{h_{j+1,j}}.$$

Généralement, pour des raisons de stabilité, on préfère utiliser l'algorithme de Gram-Schmidt modifié. Ceci revient à calculer $h_{i,j}$ et v_{j+1} comme

$$w_j = Av_j,$$

et pour $i = 1, \dots, j$

$$h_{i,j} = (w_j, v_i), \quad w_j = w_j - h_{i,j}v_i.$$

Les relations précédentes jusqu'à l'étape $k + 1$ peuvent s'écrire de façon matricielle :

$$AV_k = V_k H_k + h_{k+1,k} v_{k+1}(e_k)^T$$

où V_k est la matrice $n \times k$ dont les colonnes sont v_1, v_2, \dots et H_k est une matrice de Hessenberg supérieure (ce qui signifie que les éléments de la partie triangulaire supérieure sont non nuls ainsi que la sous-diagonale adjacente à la diagonale principale), enfin e_k est la k -ième colonne de la matrice identité.

Remarquons que lorsque A est une matrice symétrique, la matrice est aussi symétrique et donc H_k est tridiagonale (seules les trois diagonales principales sont non nulles). Ceci entraîne que les vecteurs de base v_j peuvent être calculés par une récurrence à trois termes peu coûteuse en termes d'opérations et de stockage.

Pour approcher $f(A)v$ on choisit $v_1 = v / \|v\|$ et ensuite à l'étape k :

$$f(A)v \approx \|v\|V_k f(H_k)e_1.$$

Notons que le membre de droite est aussi égal à $V_k f(H_k) V_k^* v$. L'approximation est obtenue en calculant la fonction f appliquée à H_k . Généralement on a $k \ll n$ et l'on peut utiliser les méthodes que l'on a vu précédemment pour calculer $f(H_k)$. Il est intéressant de noter que l'approximation de $f(A)v$ est un polynôme en la matrice A appliquée à v . Plus précisément

$$\|v\|V_k f(H_k)e_1 = p_{k-1}(A)v,$$

où p_{k-1} est l'unique polynôme de degré au plus $k - 1$ qui interpole f sur le spectre de H_k .

L'inconvénient de la méthode d'Arnoldi est que la taille mémoire nécessaire et le coût par itération croissent au cours des itérations puisque l'on orthogonalise par rapport à tous les vecteurs de base précédents, sauf dans le cas où A est hermitienne. Pour résoudre cette difficulté, on redémarre le procédé d'Arnoldi après un nombre fixé d'itérations (qui est fonction de la place mémoire disponible). Ce redémarrage qui est classique lorsqu'on résout des systèmes linéaires est plus délicat dans le cas d'une fonction f quelconque (voir [1] et [7]).

3.3 Logiciels

On peut trouver sur Internet des fonctions Matlab correspondant aux méthodes décrites ci-dessus. La méthode directe pour l'exponentielle de [3] est disponible sur le site de N.J. Higham cité précédemment (fonction `expmv`). On peut également utiliser `expokit`. Des fonctions utilisant la méthode d'Arnoldi avec redémarrage pour une fonction quelconque sont disponibles à l'adresse donnée à cet effet dans le *Pour en savoir plus*.

3.4 Exemples numériques

Considérons la fonction exponentielle pour f et le second exemple décrit ci-dessus. On choisit $n = 10\,000$. La matrice A est très creuse puisqu'il n'y a qu'un seul élément non nul par ligne. Le vecteur v est choisi tel que $v_i = i$, $i = 1, \dots, n$. On connaît les composantes de $x = e^A v$ qui sont $x_i = \cosh(1)v_i + \sinh(1)v_{n-i+1}$. On utilise Matlab 7.1. La fonction `expmv` de [3] donne une norme de l'erreur de $2,3493 \cdot 10^{-10}$ pour un temps de calcul de 0,20 s. La fonction `expv` de [22] donne $3,3262 \cdot 10^{-10}$ et 0,011 s. `funm_kryl` avec 4 itérations et sans redémarrage donne $2,8849 \cdot 10^{-10}$ et 0,070 s. Ces temps de calcul doivent être considérés dans un sens relatif puisqu'ils dépendent évidemment de l'ordinateur utilisé ainsi que de la version de Matlab. D'autre part, cet exemple ne reflète peut-être pas les mérites respectifs des différentes méthodes pour une matrice

comportant beaucoup plus d'éléments non nuls par ligne. Il a été choisi parce que l'on connaît facilement la solution exacte.

4. Méthodes pour $u^T f(A)v$

Dans cette section on décrit comment utiliser des méthodes de quadrature pour calculer des approximations de $u^T f(A)v$ (voir [11]). On suppose que la matrice A est symétrique réelle et on utilise la décomposition spectrale de A qui s'écrit $A = ZDZ^T$, où Z est une matrice orthonormale dont les colonnes sont les vecteurs propres normalisés de A et D est une matrice diagonale dont les éléments diagonaux sont les valeur propres λ_i de A , que l'on ordonne

$$\lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n.$$

Comme on l'a vu, la définition de $f(A)$ est :

$$f(A) = Zf(D)Z^T.$$

La forme bilinéaire $u^T f(A)v$ peut s'écrire :

$$\begin{aligned} u^T f(A)v &= u^T Zf(D)Z^T v, \\ &= \gamma^T f(D)\beta, \\ &= \sum_{i=1}^n f(\lambda_i) \gamma_i \beta_i. \end{aligned}$$

La dernière somme peut être considérée comme une intégrale de Riemann-Stieltjes :

$$I[f] = u^T f(A)v = \int_a^b f(\lambda) d\alpha(\lambda),$$

où la mesure α est constante par morceaux et définie par :

$$\alpha(\lambda) = \begin{cases} 0, & \text{si } \lambda < a = \lambda_1, \\ \sum_{j=1}^i \gamma_j \beta_j, & \text{si } \lambda_i \leq \lambda < \lambda_{i+1}, \\ \sum_{j=1}^n \gamma_j \beta_j, & \text{si } b = \lambda_n \leq \lambda. \end{cases}$$

Lorsque $u = v$ on a $\gamma = \beta$ et α est une fonction positive croissante ainsi que lorsque $\gamma_j \beta_j > 0$. Remarquons que la mesure α dépend de u et de A à travers ses valeurs et vecteurs propres.

Considérons le cas $u = v$. L'idée pour obtenir une approximation de $I[f]$ est d'utiliser des formules de quadrature pour l'intégrale de Riemann-Stieltjes. On choisit des quadratures de Gauss. La formule générale est :

$$I[f] = \int_a^b f(\lambda) d\alpha(\lambda) = \sum_{j=1}^N w_j f(t_j) + \sum_{k=1}^M v_k f(z_k) + R[f],$$

où les poids $[w_j]_{j=1}^N$ et les noeuds $[t_j]_{j=1}^N$ sont inconnus et les noeuds $[z_k]_{k=1}^M$ sont prescrits. Si $M = 0$, on obtient la quadrature de Gauss sans noeuds prescrits. Si $M = 1$ et $z_1 = a$ ou $z_1 = b$, on a la quadrature de Gauss-Radau. Si $M = 2$ et $z_1 = a$, $z_2 = b$, on a la quadrature de Gauss-Lobatto. La formule de quadrature de Gauss intègre exactement les polynômes jusqu'au degré $2N - 1$ en évaluant le polynôme en N points.

Le reste $R[f]$ est donné par :

$$R[f] = \frac{f^{(2N+M)}(\eta)}{(2N+M)!} \int_a^b \prod_{k=1}^M (\lambda - z_k) \left[\prod_{j=1}^N (\lambda - t_j) \right]^2 d\alpha(\lambda), \quad a < \eta < b.$$

Dans certains cas on connaît le signe du reste. Par exemple, si la fonction f est telle que la dérivée d'ordre $2N$ est positive sur $[a, b]$, le reste de la quadrature de Gauss ($M = 0$) est positif et on obtient une borne inférieure de $I[f]$. C'est le cas de la fonction exponentielle. Pour cette fonction les formules de Gauss-Radau donnent une borne inférieure avec $z_1 = a$ et une borne supérieure avec $z_2 = b$.

Comment calcule-t-on les noeuds et les poids de ces formules de quadrature ? Il est bien connu qu'il existe une famille de polynômes orthogonaux associés à la mesure α . Plus précisément, il existe $p_0(\lambda), p_1(\lambda), \dots$ tels que :

$$\int_a^b p_i(\lambda) p_j(\lambda) d\alpha(\lambda) = \begin{cases} 1, & \text{si } i = j, \\ 0, & \text{autrement} \end{cases}$$

et le polynôme p_k est de degré k exactement. De plus les racines de p_k sont distinctes, réelles et dans l'intervalle $[a, b]$. Comme pour tous les polynômes orthogonaux, cet ensemble vérifie une relation à trois termes :

$$\begin{aligned} \gamma_j p_j(\lambda) &= (\lambda - \omega_j) p_{j-1}(\lambda) - \gamma_{j-1} p_{j-2}(\lambda), \quad j = 1, 2, \dots, N \\ p_{-1}(\lambda) &\equiv 0, \quad p_0(\lambda) \equiv 1, \end{aligned}$$

si l'on suppose $\int_a^b d\alpha = 1$. La récurrence peut s'écrire sous forme matricielle :

$$\lambda P(\lambda) = J_N P(\lambda) + \gamma_N P_N(\lambda) e_N,$$

où

$$\begin{aligned} P(\lambda) &= [p_0(\lambda), p_1(\lambda), \dots, p_{N-1}(\lambda)]^T, \\ e_N &= (0, 0, \dots, 0, 1)^T, \end{aligned}$$

et

$$J_N = \begin{pmatrix} \omega_1 & \gamma_1 & & & & \\ \gamma_1 & \omega_2 & \gamma_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & & & \ddots & \\ & & & & & \gamma_{N-2} & \omega_{N-1} & \gamma_{N-1} \\ & & & & & & \gamma_{N-1} & \omega_N \end{pmatrix}$$

est une matrice tridiagonale symétrique. Les valeurs propres de J_N sont simples et réelles car $\gamma_i \neq 0$, $i = 1, \dots, N - 1$. On voit facilement que les zéros du polynôme P_N sont les valeurs propres de J_N . Il est bien connu que les noeuds de la formule de quadrature de Gauss sont donnés par les zéros de P_N et donc par les valeurs propres de J_N (voir [11]). Les poids w_j sont donnés par les carrés des premiers éléments des vecteurs propres normalisés de J_N . Dans le cas où la mesure n'est pas d'intégrale égale à 1, il faut multiplier les carrés par $\int_a^b d\alpha$. On voit donc qu'il n'est pas nécessaire de connaître les valeurs et les vecteurs propres de A pour calculer une approximation de la forme bilinéaire.

Pour les quadratures de Gauss-Radau et de Gauss-Lobatto, il faut étendre la matrice J_N avec une ligne et une colonne supplémentaires de telle sorte que cette matrice ait une ou deux valeurs propres imposées (voir [11]).

Il reste à savoir comment l'on peut calculer la matrice J_N (et/ou ses extensions). Pour ce faire on peut utiliser la méthode de Lanczos. Cette dernière, bien qu'elle ait été introduite auparavant, peut être vue comme un cas particulier de la méthode d'Arnoldi, exposée plus haut, lorsque la matrice A est symétrique. La propriété de symétrie de la matrice fait qu'il n'est pas nécessaire d'orthogonaliser à chaque itération par rapport à tous les vecteurs précédents, mais simplement par rapport aux deux précédents. Les matrices

H_k que l'on génère sont donc tridiagonales et symétriques, on a donc $J_N = H_N$. Les vecteurs de Lanczos v_j sont orthogonaux et donnés par $v_k = p_k(A)v_1$ et l'on a :

$$(v_k, v_l) = \int_a^b p_k(\lambda) p_l(\lambda) d\alpha(\lambda)$$

où $a \leq \lambda_{min}(A)$, $b \geq \lambda_{max}(A)$ et α est la mesure définie précédemment.

Pour calculer $v^T f(A)v$ il suffit donc d'utiliser la méthode de Lanczos avec v (éventuellement normalisé) pour vecteur de départ. On génère des matrices tridiagonales dont il faut calculer les valeurs propres et les premières composantes des vecteurs propres pour obtenir les nœuds et les poids de la formule de quadrature de Gauss. En fait, il n'est quelquefois même pas nécessaire de calculer les nœuds et les poids car l'on a :

$$\sum_{j=1}^N w_j f(t_j) = (e_1)^T f(J_N) e_1.$$

Il suffit donc, si faire se peut, de calculer l'élément $(1, 1)$ de $f(J_N)$, J_N étant une matrice tridiagonale symétrique de « petite » taille. Cet aspect de la méthode est à relier avec ce que l'on a exposé dans la section précédente, cependant, ici, on peut obtenir des bornes inférieure et supérieure si la fonction f possède de bonnes propriétés.

Lorsque $u \neq v$ on peut utiliser la relation :

$$u^T f(A)v = [(u+v)^T f(A)(u+v) - (u-v)^T f(A)(u-v)]/4$$

Il suffit donc de calculer des approximations de $(u+v)^T f(A)(u+v)$ et $(u-v)^T f(A)(u-v)$, ce qui se fait comme on l'a vu dans le cas $u=v$.

Les techniques décrites ci-dessus peuvent être étendues au cas où l'on s'intéresse à $W^T f(A)W$ avec W une matrice $n \times m$. On utilise alors la méthode de Lanczos par blocs. Notons que les méthodes par blocs donnent aussi des approximations de $u^T f(A)v$ dans le cas $u \neq v$ car il suffit de choisir $W = [u \ v]$.

Avec les méthodes de quadrature on peut également obtenir des quantités qui ne peuvent pas s'exprimer comme $v^T f(A)v$. Par exemple, si l'on veut calculer la trace de puissances de A (éventuellement négatives), on a :

$$\text{trace}(A^r) = \sum_{j=1}^n \lambda_j^r = \int_a^b \lambda^r d\alpha(\lambda).$$

L'intégrale peut être calculée de façon approchée avec la quadrature de Gauss. Une autre application est le calcul du déterminant de A . Pour cela on utilise, lorsque la matrice A est symétrique et définie positive, la formule

$$\log(\det(A)) = \text{trace}(\log(A)).$$

Ces méthodes ont été, dans une certaine mesure, étendues au cas où A est non symétrique. On peut dans ce cas utiliser l'algorithme d'Arnoldi (voir [6]). Pour la fonction $f(x) = 1/x$, voir aussi [23] où l'on

utilise la méthode du gradient bi-conjugué. Cependant, ces méthodes ne sont pas basées sur des formules de quadrature et l'on n'obtient pas des bornes mais simplement des approximations.

4.1 Logiciels

Des fonctions Matlab (`mmq_toolbox`), correspondant aux méthodes décrites dans cette section et à la référence [11], sont disponibles à l'adresse donnée dans le *Pour en savoir plus*.

4.2 Exemples numériques

Choisissons la matrice symétrique définie positive venant de la discréttisation par différences finies de l'équation aux dérivées partielles $-\Delta u = g$ sur le carré $[0, 1]^2$ avec des conditions aux limites de Dirichlet homogènes $u = 0$. Prenons 32 points de discréttisation dans chaque direction. On obtient une matrice A d'ordre $n = 900$. On veut calculer un élément de l'exponentielle de A , par exemple l'élément diagonal $(50, 50)$ dont la valeur α calculée par `expm_new` est 277,40605059. La quadrature de Gauss donne une borne inférieure et on obtient une différence de $-6,8212 \cdot 10^{-13}$ entre la borne inférieure et α après 12 itérations de la méthode de Lanczos. Avec la quadrature de Gauss-Radau on peut obtenir une borne supérieure avec une différence $6,8212 \cdot 10^{-13}$. Le temps de calcul avec Matlab 7.1 est de 0,024 s pour calculer quatre approximations (Gauss, Gauss-Radau et Gauss-Lobatto). Notons qu'on obtient déjà de bonnes approximations avec seulement 5 ou 6 itérations et que, de plus, avec ce logiciel on peut considérer n'importe quelle fonction suffisamment régulière. Pour d'autres expériences numériques, voir [11].

Si on utilise `expmv` pour calculer $f = e^A v$ et ensuite $v^T f$ avec v la 50-ième colonne de la matrice identité, on obtient une différence de $-5,1159 \cdot 10^{-13}$ et un temps de calcul de 0,014 s. La fonction `funm_kryl` utilisée de la même façon avec redémarrage toutes les 5 itérations et après 9 cycles (soit 45 itérations au total) donne une différence de $2,8422 \cdot 10^{-13}$ et un temps de calcul de 0,11 s, mais on pourrait peut-être régler les paramètres pour diminuer ce temps de calcul. L'utilisation de `expv` venant d'`expokit` donne une différence de $-6,2528 \cdot 10^{-13}$, soit sensiblement la même chose que la quadrature de Gauss avec un temps de calcul de 0,030 s.

5. Conclusion

Les méthodes de calcul des éléments d'une fonction de matrice décrites dans cet article ont atteint un degré de maturité satisfaisant, tant sur le plan de la précision des calculs que sur le plan des temps d'exécution. On peut espérer faire encore quelques progrès pour certaines fonctions particulières. Pour le calcul d'une fonction de matrice appliquée à un vecteur, il est possible que de nouvelles méthodes itératives puissent amener des améliorations. En ce qui concerne les formes quadratiques et bilinéaires, on peut souhaiter des développements de nouveaux algorithmes lorsque la matrice n'est pas symétrique.

Calcul de fonctions de matrices

par **Gérard MEURANT**
Ancien directeur de recherche au CEA

Sources bibliographiques

- [1] AFANASJEW (M.), EIERMANN (M.), ERNST (O.G.) et GÜTTEL (S.). – *Implementation of a restarted Krylov subspace method for the evaluation of matrix functions*. Linear Algebra Appl., vol. 429, p. 2293-2314 (2008).
- [2] AL-MOHY (A.H.) et HIGHAM (N.J.). – *A new scaling and squaring algorithm for the matrix exponential*. SIAM J. Matrix Anal. Appl., vol. 31 n° 3, p. 970-989 (2009).
- [3] AL-MOHY (A.H.) et HIGHAM (N.J.). – *Computing the action of the matrix exponential with an application to exponential integrators*. SIAM J. Sci. Comput., vol. 33 n° 2, p. 488-511 (2011).
- [4] AL-MOHY (A.H.) et HIGHAM (N.J.). – *Improved inverse scaling and squaring algorithms for the matrix logarithm*. SIAM J. Sci. Comput., vol. 34 n° 4, p. C153-C169 (2012).
- [5] BREZINSKI (C.) et VAN ISEGHEM (J.). – *Padé approximations*. in « Handbook of Numerical Analysis », vol. III, P.G. Ciarlet and J.L. Lions eds., North-Holland, Amsterdam, p. 47-222 (1994).
- [6] CALVETTI (D.) et SUN-MI KIM ET REICHEL (L.). – *Quadrature rules based on the Arnoldi process*. SIAM J. Matrix Anal. Appl., vol. 26 n° 3, p. 765-781 (2005).
- [7] EIERMANN (M.), ERNST (O.G.) et GÜTTEL (S.). – *Deflated restarting for matrix functions*. SIAM J. Matrix Anal. Appl., vol. 32 n° 2, p. 621-641 (2011).
- [8] ESTRADA (E.), NAOMICHI (H.) et BENZI (M.). – *The physics of communicability in complex networks*. Physics Reports, vol. 514.3, p. 89-119 (2012).
- [9] FROMMER (A.) et SIMONCINI (V.). – *Matrix functions in « Model Order Reduction : Theory, Research Aspects and Applications »*, Mathematics in Industry, W.H.A. Schilders, et H.A. van der Vorst eds, Springer, p. 275-304 (2008).
- [10] GOLUB (G.H.) et VAN LOAN (C.). – *Matrix computations*, third edition, Johns Hopkins University Press (1989).
- [11] GOLUB (G.H.) et MEURANT (G.). – *Matrices, moments and quadrature with applications*. Princeton University Press (2010).
- [12] HALE (N.), HIGHAM (N.J.) et TREFETHEN (L.N.). – *Computing A^α , $\log(A)$ and related matrix functions by contour integrals*, SIAM J. Numer. Anal., vol. 46 n° 5, p. 2505-2523 (2008).
- [13] HIGHAM (N.J.). – *Functions of matrices, theory and computation*. SIAM (2008).
- [14] HIGHAM (N.J.). – *Functions of matrices*. MIMS EPrint 2013.27, University of Manchester (UK) (2013).
- [15] HIGHAM (N.J.) et AL-MOHY (A.H.). – *Computing matrix functions*. Acta Numerica, vol. 19, p. 159-208 (2010).
- [16] HIGHAM (N.J.) et LIN LIJING. – *Matrix functions : A short course*. MIMS EPrint 2013.73, University of Manchester (UK) (2013).
- [17] MEURANT (G.). – *Méthodes de Krylov pour la résolution des systèmes linéaires*. Techniques de l'Ingénieur, [AF 488] (2007).
- [18] MOLER (C.) et VAN LOAN (C.). – *Nineteen dubious ways to compute the exponential of a matrix*. SIAM review, vol. 20 n° 4, p. 801-836 (1978).
- [19] MOLER (C.) et VAN LOAN (C.). – *Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later*. SIAM review, vol. 45 n° 1, p. 3-49 (2003).
- [20] SAAD (Y.). – *Analysis of some Krylov subspace approximations to the matrix exponential operator*. SIAM J. Numer. Anal., vol. 29, p. 209-228 (1992).
- [21] SHEEHAN (B.N.), SAAD (Y.) et SIDJE (R.B.). – *Computing $\exp(-\tau A)b$ with Laguerre polynomials*, Electron. Trans. Numer. Anal., vol. 37, p. 147-165 (2010).
- [22] SIDJE (R.B.). – *Expokit : A software package for computing matrix exponentials*. ACM Transactions on Mathematical Software, vol. 24 n° 1, pp. 130-156 (1998).
- [23] STRAKOŠ (Z.) et TICHÝ (P.). – *On efficient numerical approximation of the bilinear form $c^* A^{-1} b$* . SIAM J. Sci. Comput., vol. 33, p. 565-587 (2011).

À lire également dans nos bases

BREZINSKI (C.) et REDIVO-ZAGLIA (M.). – *Interpolation, approximation et extrapolation rationnelles*. [AF 1 390] Mathématiques pour l'ingénieur (2013).

Outils logiciels

CISIA juin 2000 *Le Bayésien* (version pour Windows Vista)
 CISIA 1 avenue Herbillon, 94160 Saint-Mandé, France

Les outils logiciels disponibles sont décrits dans le texte

Sites Internet

Boîte à outils mft_toolbox pour Matlab :
<http://www.maths.manchester.ac.uk/~higham>
 Boîte à outils expokit pour Matlab :
<http://www.maths.uq.edu.au/expokit/>
 Fonctions Matlab contenant des intégrateurs exponentiels pour les équations différentielles :
<http://www.math.ntnu.no/num/expint/matlab.php>

Fonctions Matlab utilisant la méthode d'Arnoldi avec redémarrage pour une fonction quelconque :
<http://www.guettel.com>
 Boîte à outils mmq_toolbox pour Matlab :
<http://gerard.meurant.pagesperso-orange.fr/>