TP Matlab 3

Newton et valeurs propres

3.1 Des valeurs propres ?

Les valeurs propres et vecteurs propres sont particulièrement utiles dans nombre de domaines. Un exemple parmi tant d'autres : la RATP a longuement étudié (avec un succès tout relatif), les fréquences de résonances de ses trains pour éviter que des fréquences désagréables pour les utilisateurs ne soient excitées lorsque le train roule. Comment faire pour cela ? Il "suffit" de modéliser le train – c'est-à-dire trouver sa fonction de transfert– puis de chercher les fréquences propres entre 10Hz et 20000Hz excitables, c'est-à-dire qui seraient associées à un module supérieur à l'unité.

Autre exemple particulièrement important : on évite, dans les machines outils, que les fréquences de résonance d'une machine soit un harmonique bas des fréquences de fonctionnement, pour éviter un vieillissement prématuré, voir même une dégradation, de la machine, due aux vibrations.

Enfin, les valeurs propres permettent d'extraire facilement les solutions des systèmes d'équations différentiels linéaires.

En fait, on résout ici des problèmes aux valeurs propres.

3.2 Faire une décomposition aux valeurs propres

Soit M une matrice de $\mathcal{M}_n$. Pour la suite du TP, posons $n = 4$.
On prendra au choix une matrice définie aléatoirement, par exemple faite de nombre entier entre -10 et 10 – utiliser alors la fonction `rand` et la fonction `floor`– ou la matrice M suivante :

$$M = \begin{pmatrix} 2 & 4 & 8 & 1 \\ -5 & -4 & 7 & -10 \\ 6 & 1 & 6 & -2 \\ 9 & -8 & -5 & -4 \end{pmatrix}$$

On pourra voir M comme la matrice d'un système différentiel $\mathcal{S}$:

$$\mathcal{S} : \left\{ \begin{array}{rcrrrr} \dot{x}_1 & = & 2x_1 & +4x_2 & +8x_3 & +x_4 \\ \dot{x}_2 & = & -5x_1 & -4x_2 & +7x_3 & -10x_4 \\ \dot{x}_3 & = & 6x_1 & +x_2 & +6x_3 & -2x_4 \\ \dot{x}_4 & = & 9x_1 & -8x_2 & -5x_3 & -4x_4 \end{array} \right.$$

Divers rappels

Valeur propre Une valeur propre de M est un scalaire λ tel qu'il existe un vecteur $\boldsymbol{x}$ – nommé vecteur propre de M– de $\mathbb{K}^n$ qui vérifie :

$$M\boldsymbol{x} = \lambda\boldsymbol{x}.$$

Intuitivement, un vecteur propre est donc une direction privilégiée de l'espace lorsque $|\lambda| > 1$.

Polynôme caractéristique Le polynôme caractéristique de M est le polynôme ayant pour racines les valeurs propres de M, c'est à dire :

$$\mathbb{P}(X) = \prod (X - \lambda_i)^{m_i}$$

où m_i est la multiplicité de chaque valeur propre (Il y a m_i vecteurs propres indépendants associés à la valeur propre). On a donc $\sum m_i = n$.

Trouver ce polynôme peut se faire grâce au déterminant, en notant $\mathbb{I}d_n$ la matrice unité de $\mathcal{M}_n$:

$$\mathbb{P}(X) = \det(\mathbb{I}d_n X - M)$$

La fonction matlab `poly` rend les coefficients du polynôme caractéristique d'une matrice.

Méthode de Newton Pour trouver les racines (les zéros) d'une fonction f, une méthode itérative instinctive est la méthode de Newton : proche de la racine, la fonction est monotone. Suffisamment proche, on peut même approximer la fonction par sa tangente. L'intersection de la tangente avec l'axe des abscisses doit donc donner une meilleure approximation du zéro. En pratique, partant du point X_0, on calcul X_1 tel que :

$$0 = f(X_0) + f'(X_0)(X_1 - X_0)$$

On itère jusqu'à la précision voulue. Attention, néanmoins, lorsque le zéro est complexe, la méthode ne peut pas converger.

3.3 Au boulot !

3.3.1 Premier job : passer par le polynôme caractéristique

1) Identifier le polynôme caractéristique de M avec la fonction `poly` de matlab.

Calculer ses racines

2) Mettre en place un algorithme pour calculer les racines λ du polynôme caractéristique de M, avec la méthode de Newton. Les fonctions `polyval` et `polyder` seront vos amis. Penser à bien vérifier les possibles multiplicités du zéro.
3) Comparer le résultat avec la fonction `roots` puis avec la fonction `fzero` (qui nécessite de créer une fonction).

Calculer les vecteurs propres de la matrice M

4) Identifier les vecteurs propres $\boldsymbol{x}$ associés à chaque valeur propre identifiée précédemment, en résolvant $M\boldsymbol{x} = \lambda\boldsymbol{x}$. Penser à trouver autant de vecteur propre que la multiplicité de la valeur propre.

3.3.2 Vérification

5) Tester les résultats avec la fonction `eig` de Matlab, qui donne vecteurs propres et valeurs propres d'une matrice.