DEVOIR 1

Analyse numérique pour ingénieur — Hiver 2021
MAT-2910

Consignes

— Deux programmes Matlab, un script (devoir.m) et une fonction (pointfixe.m) accompagnent le présent questionnaire (voir site web).
— Ce devoir doit obligatoirement être réalisé avec le logiciel Matlab.
— Vous ne pouvez pas obtenir d'aide du CDA pour ce devoir.
— Il y a un rapport à fournir. Il doit être produit avec un éditeur tel que Word ou LaTeX. Tout rapport manuscrit se verra attribuer la note 0. L'absence de rapport donnera la note de 0 au devoir. Par ailleurs, l'absence de toute figure demandée dans le rapport entrainera la note 0 pour la question correspondante. Au total il y a 10 figures à mettre dans le rapport, les Figures 3 à 12.
— Il y a au total 4 programmes Matlab à fournir. L'absence des programmes donnera la note de 0 au devoir.
— Il y a au total 2 programmes Matlab (.m) à créer : *newton.m* et *steffenson.m*. Ils sont à remettre avec les programmes *devoir.m* et *pointfixe.m*. Le programme *devoir.m* est à modifier. Le programme *pointfixe.m* ne doit pas être modifié mais il doit quand même être remis avec les autres.
Le fichier script (*devoir .m*) fait appel aux 4 fonctions Matlab. Lorsqu'on lance ce fichier script, il doit produire les 12 figures demandées (Figures 1 à 12). Si les figures n'apparaissent pas, le devoir se verra attribuer la note 0 aux questions correspondantes. De même si le numéro de la figure ne correspond pas à ce qui est demandé pour ce numéro. Si le programme ne fonctionne pas, le devoir se verra attribuer la note 0.
— Des points sont alloués à la qualité du rapport ainsi qu'à la qualité (lisibilité) des figures qui doivent être produites par le script Matlab. Il est donc recommandé de donner des titres (ou légendes) aux figures, ainsi que d'indiquer ce qu'il y a en abscisse ou en ordonnée. L'utilisation de différentes couleurs pour plusieurs courbes sur une même figure est recommandée. Utiliser les commandes *xlabel, ylabel, title, legend* par exemple.
— Le rapport et les fichiers Matlab demandés doivent être compressés en 1 seul fichier qui doit s'appeler equipeXYZ où XYZ désigne le numéro de l'équipe attribué au moment de la création de l'équipe dans MonPortail, et qui doit être déposé dans la **boîte de dépôt** sur le site du cours.
— Le devoir peut être réalisé en équipe d'au plus deux personnes. Les deux personnes peuvent êtres de sections différentes.
— Tout plagiat, même partiel, sera sanctionné (voir plan de cours).

Remise

— Date de remise : voir sur le site du cours (www.portaildescours.ulaval.ca).
— Lieu de remise : boîte de dépôt sur le site du cours (www.portaildescours.ulaval.ca)
— **Tout travail remis après la date et l'heure d'échéance se verra pénalisé de 2% pour chaque 15 minutes de retard pour un maximum 20% pour le premièr jour. Par la suite, une pénalité de 20% par jour de retard sera imposée.**

Support

— 2 séances d'information à distance sont prévues. Les dates et heures vous seront communiquées lorsqu'elles seront fixées.

Soit $g(x) = \frac{1}{5}(x^5 - 3x^3 + 2x + 5)$. On voit bien que $r = 1$ est un point fixe de g.

On peut également facilement voir que la méthode de point fixe $x_{n+1} = g(x_n)$ converge pour ce point fixe puisque $0 < |g'(r)| < 1$.

Dans le script Matlab *devoir.m* accompagnant ce questionnaire on définit la fonction g, ainsi que des valeurs pour des paramètre N, *tol*, et x_0.

On fait ensuite appel à une fonction Matlab *pointfixe.m* qui réalise la méthode de point fixe $x_{n+1} = g(x_n)$ en partant de x_0, avec un nombre maximal d'itérations de N et une tolérance *tol* pour une estimation de l'erreur relative. La suite $(x_n)_{n\geq 0}$ est enregistrée dans le vecteur de sortie x.

Ensuite, toujours dans le script, on calcule les erreurs absolues $E_n = |x_n - r|$ qu'on enregistre dans le vecteur *erreur*, puis on trace E_n en fonction de n sur la Figure 1, en échelle logarithmique pour les ordonnées afin de mieux apprécier la convergence de E_n vers 0 lorsque n augmente.

Enfin, on fait tracer, sur une Figure 2, les rapports d'erreur E_{n+1}/E_n en fonction de n afin de mettre en évidence l'ordre 1 de la convergence de cette méthode de point fixe puisqu'en effet, le rapport semble à peu près constant pour une très majeure partie des valeurs de n.

1. Méthode du point fixe

   a) Choisir une nouvelle fonction g ayant un point fixe r pour lequel la convergence est d'ordre 2. Dans le script, à la suite des commandes précédentes, faire appel à *pointfixe.m* puis faire tracer les erreurs en fonction de n (Figure 3). Tracer une autre courbe (Figure 4) mettant en évidence le caractère quadratique de la convergence. Vous avez la liberté de choisir les valeurs de x_0, N et *tol*. L'important est que les figures traduisent bien les résultats auxquels ont doit s'attendre.
   Indiquer sur les figures les choix de g, x_0 et r.
   Dans le rapport, donner le choix de g et r et démontrer que la convergence est d'ordre 2. Reproduire les Figures 3 et 4 avec un bref commentaire pour chacune d'elles disant ce qu'elles montrent ou illustrent.

   b) Choisir une nouvelle fonction g ayant un point fixe r pour lequel il n'y a pas convergence. Dans le script, faire appel à *pointfixe.m* puis faire tracer les erreurs en fonction de n (Figure 5) jusqu'à une valeur de n pas trop grande (entre 5 et 10) mais montrant que la suite semble diverger. Comme précédemment, vous avez la liberté de choisir les valeurs de x_0, N et *tol*. L'important est que la figure illustre bien les résultats auxquels ont doit s'attendre.
   Indiquer sur la figure le choix de g, x_0 et r.
   Dans le rapport, donner le choix de g et r et démontrer que la méthode de point fixe diverge pour ce point fixe. Reproduire la Figure 5 avec un bref commentaire.

2. Méthode de Steffenson : Copier le programme *pointfixe.m* dans un nouveau programme *steffenson.m* et modifier minimalement ce dernier de manière à ce qu'il applique la méthode de Steffenson à une méthode de point fixe g. Il aura les mêmes entrées et sortie que *pointfixe.m*. Il ne diffèrera de *pointfixe.m* que par quelques lignes (le moins possible).

   a) Toujours dans le script *devoir.m*, faire appel à *steffenson.m* pour l'appliquer à la fonction g donnée au début du questionnaire et ayant 1 pour point fixe. Si possible, partir du même x_0. Faire tracer la courbe de l'erreur dans une nouvelle figure (Figure 6) et tracer également la courbe d'erreur apparaissant dans la Figure 1. il y a maintenant 2 courbes dans cette Figure 6. Indiquer laquelle est laquelle avec la commande Matlab *legend*.
   Dans le rapport, reproduire la Figure 6 avec un bref commentaire.

   b) Toujours dans le script *devoir.m*, faire appel à *steffenson.m* pour l'appliquer à la fonction g de la question 1 a) en partant du même x_0, faire tracer la courbe de l'erreur dans une Figure 7 et ajouter la courbe d'erreur apparaissant dans la Figure 3. il y a maintenant 2 courbes dans cette Figure 7. Indiquer laquelle est laquelle avec la commande Matlab *legend*.
   Dans le rapport, rappeler l'expression de g et la valeur de r, puis reproduire la Figure 7 avec un bref commentaire.

   c) Toujours dans le script *devoir.m*, faire appel à *steffenson.m* pour l'appliquer à la fonction g de la question 1 b) en partant du même x_0, faire tracer la courbe de l'erreur dans une Figure 8 et ajouter la courbe d'erreur apparaissant dans la Figure 5. il y a maintenant 2 courbes dans cette Figure 8. Indiquer laquelle est laquelle avec la commande Matlab *legend*.
   Dans le rapport, rappeler l'expression de g et la valeur de r, puis reproduire la Figure 8 avec un bref commentaire. Noter que la Figure 8 doit montrer le résultat surprenant auquel on doit cependant s'attendre(!).

3. Méthode de Newton : Copier le programme *pointfixe.m* dans un nouveau programme *newton.m* et modifier minimalement ce dernier de manière à ce qu'il applique la méthode de Newton à une fonction f. Il aura les mêmes entrées et sortie que *pointfixe.m*, sauf qu'au lieu de g c'est une fonction f (mais ce ne sont que des notations) et on ajoute une entrée supplémentaire *df* qui sera la dérivée de f. Il ne diffèrera de *pointfixe.m* que par quelques lignes (le moins possible).

   a) Choisir une fonction f ayant une racine r pour laquelle la convergence est d'ordre 2 (ou plus). Dans le script, à la suite des commandes précédentes, faire appel à *newton.m* puis faire tracer les erreurs en fonction de n (Figure 9). Tracer une autre courbe (Figure 10) mettant en évidence le caractère quadratique de la convergence. Vous avez la liberté de choisir les valeurs

de x_0, N et tol. L'important est que les figures traduisent bien les résultats auxquels ont doit s'attendre.
Indiquer sur les figures les choix de f, x_0 et r.
Dans le rapport, donner le choix de f et r et démontrer que la convergence est d'ordre 2 ou plus. Reproduire les Figures 9 et 10 avec un bref commentaire.

b) Choisir une fonction f ayant une racine r pour laquelle la convergence est d'ordre 1. Dans le script, à la suite des commandes précédentes, faire appel à *newton.m* puis faire tracer les erreurs en fonction de n (Figure 11). Tracer une autre courbe (Figure 12) mettant en évidence le caractère linéaire de la convergence. Comme d'habitude, vous avez la liberté de choisir les valeurs de x_0, N et tol. L'important est que les figures traduisent bien les résultats auxquels ont doit s'attendre.
Indiquer sur les figures les choix de f, x_0 et r.
Dans le rapport, donner le choix de f et r et démontrer que la convergence est d'ordre 2 ou plus. Reproduire les Figures 11 et 12 avec un bref commentaire faisant notamment un lien entre la valeur de l'ordre de multiplicité de la racine et la courbe de la Figure 12.