Calcul Numérique

TP : Résolution d'équations non-linéaires.

On s'intéresse dans ce TP aux diverses méthodes vues en cours et/ou en TD pour la résolution d'équations non-linéaires.

1. Implémenter une fonction `dichotomie.m`, prenant en entrée une fonction f, ainsi que trois scalaires a, b et ε et qui renvoie x, une valeur approchée du zéro de la fonction f à ε près obtenue à l'aide de la méthode de dichotomie sur l'intervalle $[a, b]$.
2. Implémenter une fonction `point_fixe.m`, prenant en entrée une fonction ψ, ainsi que deux scalaires x_0 et ε et qui renvoie deux scalaire x et k. La valeur x est une valeur approchée du point fixe de la fonction ψ à ε près (obtenu à l'aide de la méthode de point fixe initialisé à x_0 et k est le nombre d'itération qu'il a fallut pour converger.
3. Calculer (à la main) les zéros de la fonction $f\colon x \mapsto x^2 - x - 2$. Vérifier le résultat avec la méthode de dichotomie.
4. Analyser numériquement la convergence de la méthode de point fixe $x^{k+1} = \phi(x^k)$ pour le calcul des zéros de la fonction quand on utilise les fonctions d'itération suivantes

$$\begin{aligned}
&\phi_1\colon x \mapsto x^2 - 2,\\
&\phi_2\colon x \mapsto \sqrt{2+x},\\
&\phi_3\colon x \mapsto -\sqrt{2+x},\\
&\phi_4\colon x \mapsto 1 + \frac{2}{x}, \quad x \neq 0.
\end{aligned}$$

   On vérifiera notamment que la méthode ne converge pas avec ϕ_1, elle converge seulement vers 2 avec ϕ_2 et ϕ_4 et elle converge seulement vers -1 avec ϕ_3.

Dans la suite on désigne par $f : \mathbb{R} \mapsto \mathbb{R}$ la fonction donnée par

$$f(x) = 5e^{-x^2} - 6e^{-2x^2} + \frac{x}{2} + \frac{1}{2}\,.$$

5. Dans un script Matlab, construire une variable f de type fonction prenant en entrée une matrice x et qui renvoie une matrice de même type que x dont chaque composante $(f(x))_{ij}$ vaut $u(x_{ij})$.
6. Dans la figure 1, tracer la fonction f dans l'intervalle $I = [-5, 5]$ à l'aide d'une discrétisation de l'intervalle I formée de $M = 200$ points.
7. Implémenter une fonction Matlab appelée *Newton.m* prenant en entrée :
   - ▷ Une variable de type fonction f,
   - ▷ Une variable de type fonction fp,
   - ▷ un scalaire x_0,
   - ▷ un scalaire positif ε.

   et qui renvoie :
   - ▷ un scalaire x vérifiant $|f(x)| < \varepsilon$ (obtenu à l'aide de la méthode de Newton sur la fonction f, sa dérivée fp, un point de dépard x_0 et une précision ε)
   - ▷ le nombre d'itérations nécessaire
8. On fixe dans la suite $\varepsilon = 10^{-6}$. Appliquer[1] la fonction *Newton* sur la fonction u avec quelques points de départs x_0 différents. Indiquer dans chaque cas si la méthode converge. Arrive-t-on à retrouver tous les zéros de u ?

1. prendre garde à ne pas faire de boucle infinie. Si cela survient par malgré tout, stopper le calcul en faisant $ctrl + c$ dans la fenêtre de commande.

9. Implémenter une fonction Matlab appelée *Secante.m* prenant en entré :
   - ▷ Une variable de type fonction f,
   - ▷ un scalaire x_0,
   - ▷ un scalaire positif ε,
   - ▷ éventuellement d'autres choses

   et qui renvoie :
   - ▷ un scalaire x vérifiant $|f(x)| < \varepsilon$ (obtenu à l'aide de la méthode de la sécante).
   - ▷ le nombre d'itérations nécessaire
10. Cette nouvelle méthode converge-t-elle pour les mêmes valeurs initiales que précédemment ?
11. On fixe $x_0 = -\frac{5}{2}$ et on note x_k les différents itérés de la méthode de Newton.

    Afficher sur la figure 3 :
    - ▷ La fonction u, en bleu
    - ▷ l'axe de abscisses en noir
    - ▷ le point de coordonné $(x_0, 0)$ indiqué par une croix rouge
    - ▷ le point de coordonné $(x_0, u(x_0))$ indiqué par une croix rouge
    - ▷ la tangente à la courbe représentative de u en x_0 en vert
    - ▷ le point de coordonné $(x_1, 0)$ indiqué par une croix rouge
    - ▷ le point de coordonné $(x_1, u(x_1))$ indiqué par une croix rouge
    - ▷ la tangente à la courbe représentative de u en x_1 en cyan (syntaxe plot(·,·,'c') où les points sont à remplcer par ce que l'on veux afficher)
    - ▷ le point de coordonné $(x_2, 0)$ indiqué par une croix rouge

    Enfin, restreindre la zone d'affichage à l'aide de la commande : $ylim([-1, 2.5])$