Assignment 1 - Equations non-linéaires

March 4, 2025

1 Assignment 1: Equations non-linéaires

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt

from NonLinearEquationsLib import plotPhi, plotPhiIterations
```

1.1 Partie 1

Ècrire une fonction pour l'algorithme d'itérations de point fixe avec la structure suivante:

```
def  FixedPoint( phi, x0, tol, nmax ) :
    '''
    Find the fixed point of a function by iterative iterations
    FixedPoint( PHI,X0,TOL,NMAX) tries to find the fixedpoint X of the a
    continuous function PHI nearest to X0 using
    the fixed point iterations method.
    If the search fails, an error message is displayed.

    Inputs : [phi, x0, tol, nmax]
    phi   : function for which a fixed point is searched
    x0    : initial guess on the fixed point of phi
    tol   : tolerance on the increment, below which the iterations are stopped
    nmax  : maximum number of iterations, above which the iterations are stopped and
            an error message is displayed

    Outputs : [x, r, n, x_sequence]
    x     : the approximated fixed point of the function
    r     : the absolute value of the residual in X : |phi(x) - x|
    n     : the number of iterations required for computing the solution
    inc   : the increments (in absolute value) computed by FixedPoint
    x_seq : the sequence computed by Newton
    '''
```

[ ]:
```
def  FixedPoint( phi, x0, tol, nmax ) :
    '''
    Find the fixed point of a function by iterative iterations
    FixedPoint( PHI,X0,a,b, TOL,NMAX) tries to find the fixedpoint X of the a
```

```
   continuous function PHI nearest to X0 using
   the fixed point iterations method.
   If the search fails, an error message is displayed.

   Inputs : [phi, x0, tol, nmax]
   phi   : function for which a fixed point is searched
   x0    : initial guess on the fixed point of phi
   tol   : tolerance on the increment, below which the iterations are stopped
   nmax  : maximum number of iterations, above which the iterations are␣
↪stopped and
           an error message is displayed

   Outputs : [x, r, n,  x_sequence]
   x     : the approximated fixed point of the function
   r     : the absolute value of the residual in X : |phi(x) - x|
   n     : the number of iterations required for computing the solution
   inc   : the increments (in absolute value) computed by FixedPoint
   x_seq : the sequence computed by Newton
   '''

   # initialisation of loop components
   # increments (in abs value) at each iteration
   inc = [tol + 1]  # set initial increment larger than tolerance
   # sequence of iterates
   x_seq = [x0]
   # iteration counter
   n = 0

   ## !! COMPLETE HERE !! --> FIXED POINT ITERATIONS

   # Final solution and final residual
   x = x_seq[-1]
   r = np.abs(x - phi(x))

   x_seq = np.array(x_seq)
   inc = np.array(inc)[1:]

   # Error if not converged
   if n > nmax :
       raise ValueError(f'FixedPoint stopped without converging to the desired␣
↪tolerance '
                        f'because the maximum number of iterations ({nmax})␣
↪was reached!')
   else:
       print(f'FixedPoint converged in {n} iterations.\n'
             f'Identified fixed point: {x:.4e}\n'
             f'Final residual: {r:.4e}\n')
```

```
    return ## !! COMPLETE HERE !!
```

1.2 Partie 2

On considère les méthodes de point fixe $x^{(n+1)} = \varphi_i(x^{(n)})$ $(i = 1, 2)$ avec:

$$\varphi_1(x) = \frac{1}{4}\tan(2x^2)\,\mathrm{sign}(x) = \begin{cases} \frac{1}{4}\tan(2x^2) & \text{si } x \geq 0\ , \\ -\frac{1}{4}\tan(2x^2) & \text{si } x < 0\ . \end{cases} \qquad \varphi_2(x) = \frac{1}{2}\sqrt{2\arctan(4x^2)}\,\mathrm{sign}(x) = \begin{cases} \frac{1}{2}\sqrt{2\arctan(4 \\ -\frac{1}{2}\sqrt{2\arctan(\end{cases}$$

Vous pouvez visualiser les fonctions d'itération $\varphi_i(x)$ en exécutant la prochaine cellule.

Pour chaque point fixe $\bar{x}_k^i$ de chaque fonction d'itération φ_i $(i = 1, 2)$, on suppose choisir une valeur initiale $x^{(0)}$ qui soit proche de $\bar{x}_k^i$. Avec Python, étudier si la méthode converge vers $\bar{x}_k^i$, en exécutant les cellules ci-dessous.

Remarque: on va utiliser une tolerance $\tau = 10^{-4}$ et un nombre maximale d'itérations $N = 50$.

Donner un commentaire sur les résultats obtenus. En particulier, pour les deux fonctions: * discuter la **convergence locale** de la méthode, pour chaque point fixe; * discuter la convergence de la methode si on choisi x_0 très proche de 0.

[ ]:
```
[a,b] = [-1.1, 1.1]

## !! COMPLETE HERE !!
#phi1 = lambda x: ??
#phi2 = lambda x: ??

plt.rcParams['figure.figsize'] = [20, 5]

plt.subplot(1,2,1)
plotPhi(a,b,phi1,r'$\varphi_1$')

plt.subplot(1,2,2)
plotPhi(a,b,phi2,r'$\varphi_2$')

plt.show()
```

1.2.1 Test avec fonction φ_1

[ ]:
```
tol  = 1e-4
nmax = 50

# Initial Point
x0 = 0.25  # you can change this value to see when convergence occurs and when␣
 ↪it does not
```

```
# !! COMPLETE HERE !!
# zero1, residual1, niter1, inc1, x1 = ??

plt.subplot(131)
plotPhi (a,b,phi1,r'$\varphi_1$')
plt.plot(x1,phi1(x1), 'rx')

# plot the graphical interpretation of the Fixed Point method
plotPhiIterations(x1)
```

1.2.2 Test avec fonction φ_2

[ ]:
```
tol  = 1e-4
nmax = 50

# Initial Point
x0 = 0.25  # you can change this value to see when convergence occurs and when it does not

# !! COMPLETE HERE !!
# zero2, residual2, niter2, inc2, x2 = ??

plt.subplot(131)
plotPhi (a,b,phi2,r'$\varphi_2$')
plt.plot(x2,phi2(x2), 'rx')

# plot the graphical interpretation of the Fixed Point method
plotPhiIterations(x2)
```

1.3 Commentaire

Ècrivez ici votre commentaire

1.4 Partie 3

On defini la fonction f comme suit:

$$f(x) = \arctan(4x^2) - 2x^2 \quad \text{avec} \quad x \in [-1.1, 1.1] \ .$$

Pouvez-vous déduire le lien entre f et les fonctions $\{\varphi_i\}_{i=1}^2$ précédemment définies? Completer et exécuter la cellule suivante pour avoir un indice; donner un petit commentaire.

[ ]:
```
[a, b] = [-1.1, 1.1]
z = np.linspace(a,b,201)
f = lambda x: np.arctan(4*x**2) - 2*x**2

plt.figure()
```

```
plt.plot(z,f(z),'k-', linewidth=2)

plt.xlabel('x', fontsize=16)
plt.ylabel('f(x)', fontsize=16);
plt.plot([a,b], [0,0], 'k-',linewidth=0.5)
plt.plot([0,0], [np.min(f(z)),np.max(f(z))], 'k-',linewidth=0.5)
plt.legend(['f(x)'], fontsize=16)
plt.title(r'Graph of f(x)', fontsize=18, fontweight='bold')

# TODO: which values should be put in place of the zeros to visualize the␣
 ↪correct answer?
alpha = np.array([0.0, 0.0, 0.0])

plt.plot(alpha,f(alpha),'ro')
plt.annotate(r"$\alpha_1$", (alpha[0], -.2), fontsize=16)
plt.annotate(r"$\alpha_2$", (alpha[1], -.2), fontsize=16)
plt.annotate(r"$\alpha_3$", (alpha[2], -.2), fontsize=16)

plt.show()
```

1.4.1 Commentaire

Èrivez ici votre commentaire

1.5 Partie 4

D'après les résultats vus en classe, est-il possible de définir une autre fonction φ_3 qui a la même propriété que les $\{\varphi_i\}_{i=1}^2$ par rapport à f? Si oui, écrivez son expression dans la cellule suivante et exécutez-la pour vérifier le résultat des itérations de point fixe. Puis éxpliquez comment vous avez derivé son expression et discutez la **convergence locale** de la méthode vers les differents points fixes.

[ ]:
```
# WRITE HERE THE RIGHT EXPRESSION OF THE FUNCTION!
#phi3 = lambda x: ??

tol  = 1e-4
nmax = 50

# Initial Point
x0 = 0.25  # you can change this value to see when convergence occurs and when␣
 ↪it does not

# !! COMPLETE HERE !!
# zero3, residual3, niter3, inc3, x3 = ??

plt.subplot(131)
plotPhi (a,b,phi3,r'$\varphi_3$')
plt.ylim([2*a,2*b])
```

```
plt.plot(x3, phi3(x3), 'rx')

# plot the graphical interpretation of the Fixed Point method
plotPhiIterations(x3)

plt.show()
```

1.5.1 Commentaire

Èrivez ici votre commentaire