TP 1 : Calcul approché et méthode du point fixe

NB : Ne sont corrigés ici que les questions n'ayant pas été traités et corrigés en TP par tous le monde (pour ces questions, cf. vos notes).

3 Méthode du point fixe

Dans cette section section, nous allons étudier de manière thérique et pratique différentes méthodes permettant de résoudre

$$x - \cos(x) = 0 \qquad x \in [0, 1]$$

Durant le TP, veuillez passer rapidement les questions théoriques pour vous concentrer sur Scilab. Un corrigé sera distribué plus tard pour les questions théoriques.

Question 3 *Montrer que la fonction $f(x) = x - \cos(x)$ n'admet qu'un seul et unique zéro sur $[0, 1]$. (Pour aide, $0 < \sin(1) < 1$ et $0 < \cos(1) < 1$). On note ℓ ce zéro.*

Remarquons tout d'abord que $f(0) = -1 < 0$ et $f(1) = 1 - \cos(1) > 0$, alors en utilisant le **théorème des valeurs intermédiaires** on a que f admet au moins un zéro de f dans $[0, 1]$. D'autre part, $f'(x) = 1 - \sin(x)$, f admet donc le tableau des variations suivant sur $[0, 1]$

x	0	ℓ	1
$f'(x)$	0	$+$	
$f(x)$	-1	0 ↗	$f(1) > 0$

et f admet un unique zéro sur l'intervalle $[0, 1]$.

3.1 Méthode naïve

On a vu dans le cours qu'une méthode permettant de résoudre de manière numérique était de définir la suite suivante

$$\begin{cases} x_0 \text{ fixé dans } [0, 1] \\ x_{n+1} = g(x_n) \end{cases} \qquad g(x) = \cos(x) \text{ pour } x \in [0, 1]$$

Question 4 *Montrer que cette suite converge bien vers ℓ.*

On a montré prédemment que f admettait un unique zéro sur $[0, 1]$. Or pour tout n, $x_n \in [0, 1]$, donc pour montrer que la suite converge bien vers ℓ, il suffit de montrer que la suite (x_n) converge.

g est une fonction continue et dérivable sur $[0,1]$ et $g'(x) = \sin(x)$ est une fonction continue sur $[0,1]$. g' admet donc un maximum sur $[0,1]$ et comme elle est strictement croissante sur $[0,1]$

$$\text{Pour tout } x \text{ de } [0,1] \qquad 0 \leq g'(x) \leq g'(1) < 1$$

donc par passage au maximum

$$\max_{x\in[0,1]} |g'(x)| \leq g'(1) < 1$$

Le maximum de $|g'|$ sur $[0,1]$ est strictement inférieur à 1, donc d'après le cours, la suite (x_n) converge vers ℓ unique solution de l'équation $\ell = g(\ell)$ sur $[0,1]$ (et solution de $f(\ell) = 0$).

Question 5 *Montrer que cette méthode est d'ordre 1 dans ce cas.*

On a $g(\ell) = 0$ (par définition) et $g'(\ell) = \sin(\ell) \neq 0$ (car $\ell \neq 0$ sur $[0,1]$) donc la méthode est d'ordre 1.

Question 6 *Récupérer sur l'intranet le fichier* ***pointfixe.sci*** *définissant une certaine fonction.*

```
function [ y ] = pointfixe(x0,n)
y = x0 ;
for i = 1:n
   y = ..................... ;
end
endfunction
```

Cette fonction prend en entrée `x0` *une valeur initiale de la suite et* `n` *le nombre d'itérations. La fonction doit retourner* `y` *valant* x_n*. Compléter le fichier. Le cosinus se calcule par* `cos()` *en Scilab. Charger la fonction dans Scilab et vérifier pour différentes valeurs initiales prises dans* $[0,1]$ *que la suite converge vers le même* $\ell = \cos(\ell)$*. Que vaut* ℓ *? (Normalement vous devez trouver* $\ell \approx 0,739085133215160672293 1$*)*

Voici le fichier complété

```
function [ y ] = pointfixe(x0,n)
y = x0 ;
for i = 1:n
   y = cos(y) ;
end
endfunction
```

Question 7 *Récupérer sur l'intranet le fichier* ***pointfixeErreur.sci*** *définissant une certaine fonction.*

```
function [ err ] = pointfixeErreur(x0)
err = 0 ;
for i = 1:50
   y = pointfixe(x0,i) ;
   err(i) = ................. ;
end
endfunction
```

Cette fonction prend en entrée `x0` *une valeur initiale de la suite. Compléter le fichier tel que* `err(i)` *vaut* $|e_i|$ *l'erreur à l'étape* i *en valeur absolue ici. La valeur absolue se calcule par* `abs()` *en Scilab. Charger la fonction dans Scilab. Afficher alors l'évolution de l'erreur en tapant directement en ligne les commandes suivantes*

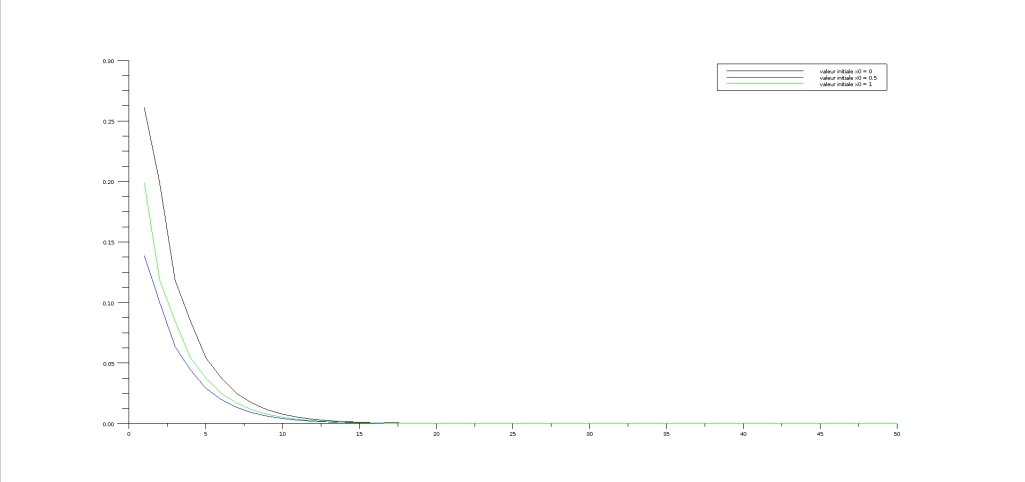
```
--> erreur = pointfixeErreur(0.1)
--> plot2d(erreur)
```

`plot2d` *est une fonction permettant d'afficher graphiquement les résultats. Essayer pour différentes valeurs initiales prises dans* $[0, 1]$.

On complète le fichier **pointfixeErreur.sci** de la manière suivante

```
function [ err ] = pointfixeErreur(x0)
err = 0 ;
l = 0.7390851332151606722931 ; // limite de la suite attendue
for i = 1:50
   y = pointfixe(x0,i) ;
   err(i) = abs(y - l) ; // ecart en valeur absolue entre x_i et la limite
end
endfunction
```

Puis on trace pour quelques valeurs ($x_0 = 0$ en noir, $x_0 = 0.5$ en bleu et $x_0 = 1$ en vert) l'évolution de l'écart entre x_i et la limite attendue.



On vérifie bien sur ces exemples la convergence vers la valeur attendue.

Question 8 *Récupérer sur l'intranet le fichier* ***pointfixeVitesseConvergence.sci*** *définissant une certaine fonction*

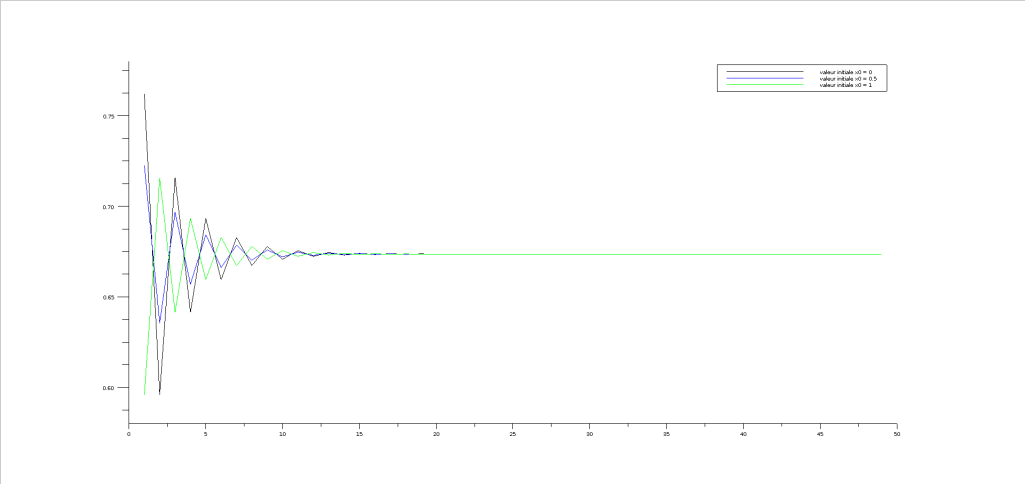
```
function [ ratio ] = pointfixeVitesseConvergence(x0)
err = pointfixeErreur(x0) ;
for i = 1:49
   ratio(i) = ..... ;
end
endfunction
```

Cette fonction prend en entrée `x0` *une valeur initiale de la suite. Compléter le fichier tel que* `ratio(i)` *vaut* $\frac{|e_{i+1}|}{|e_i|}$ *le coefficient de réduction asymptotique de l'erreur. Charger la fonction dans Scilab. Afficher graphiquement les résultats à l'aide de la commande* `plot2d`*. Vérifie t-on que la méthode est d'ordre* 1 *?*

On complète le fichier **pointfixeVitesseConvergence.sci** de la manière suivante

```
function [ ratio ] = pointfixeVitesseConvergence(x0)
err = pointfixeErreur(x0) ;
for i = 1:49
   ratio(i) = abs(err(i+1)./err(i)) ; // err(i) correspond a abs(l - x_i)
end
endfunction
```

Puis on trace pour quelques valeurs ($x_0 = 0$ en noir, $x_0 = 0.5$ en bleu et $x_0 = 1$ en vert) l'évolution du ratio calculé en fonction du nombre d'itérations de la suite.



Les trois courbes tendent vers le même réel strictement positif donc $\frac{|e_{i+1}|}{|e_i|}$ admet une limite réelle strictement positive, cela confirme bien que la méthode est d'ordre 1.

3.2 Méthode de Newton

On a vu rapidement dans le cours la méthode de Newton

$$\begin{cases} x_0 \text{ fixé dans } [0,1] \\ x_{n+1} = x_n - \dfrac{f(x_n)}{f'(x_n)} \end{cases}$$

Question 9 *Que vaut* g *? Montrer que cette méthode est d'ordre* 2 *dans ce cas. (Pour info,* $0 < \cos(\ell) < 1$ *et* $0 < \sin(\ell) < 1$*).*

La fonction g vaut ici sur $[0,1]$

$$g(x) = x - \frac{f(x)}{f'(x)} = x - \frac{x - \cos(x)}{1 + \sin(x)}$$

et sa dérivée g' sur $[0,1]$ vaut

$$g'(x) = 1 - \frac{(1+\sin x)^2 - (x-\cos x)\cos x}{(1+\sin x)^2} = \frac{\cos x(x-\cos x)}{(1+\sin x)^2}$$

On vérifie que $g'(\ell) = 0$ (car $l = \cos(\ell)$), la méthode de Newton est au moins d'ordre 2. Étudions maintenant g'' sur $[0,1]$

$$g''(x) = \frac{1}{(1+\sin x)^4}\left[(2\cos x \sin x - x\sin x)(1+\sin x)^2 - 2\cos^2 x(x-\cos x)(1+\sin x)\right]$$

et

$$g''(\ell) = \frac{(2\cos(\ell)-\ell)\sin(\ell)(1+\sin(\ell))^2}{(1+\sin(\ell))^4} = \frac{(2\cos(\ell)-\ell)\sin(\ell)}{(1+\sin(\ell))^2} = \frac{\ell\sin(\ell)}{(1+\sin(\ell))^2} \neq 0$$

La méthode de Newton est donc bien d'ordre 2.

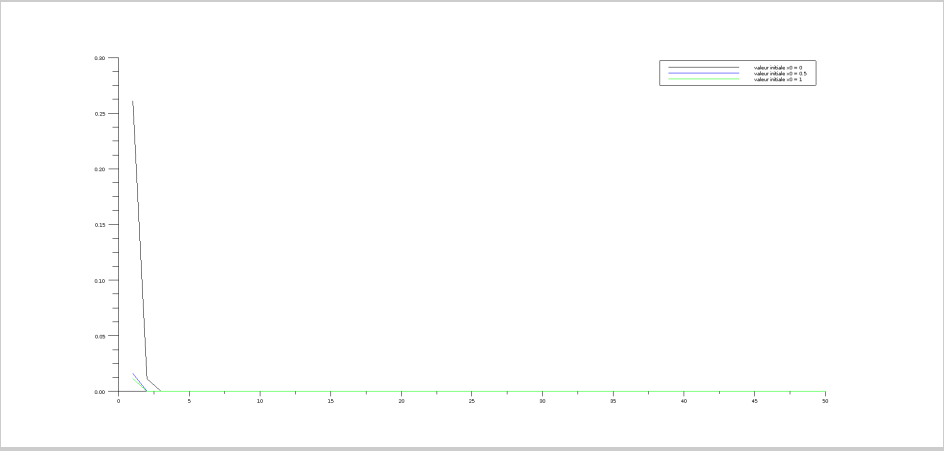
Question 10 *Modifier le fichier* ***pointfixe.sci*** *afin que la suite utilise la méthode de Newton. Charger la fonction dans Scilab et vérifier pour différentes valeurs initiales prises dans* $[0,1]$ *que la suite converge vers le même* $\ell = \cos(\ell)$*. Que vaut* ℓ *? (Normalement vous devez trouver* $\ell \approx 0,739085133215160672293 1$*)*

On modifie le fichier **pointfixe.sci** de la manière suivante afin d'utiliser la méthode de Newton.

```
function [ y ] = pointfixe(x0,n)
y = x0 ;
for i = 1:n
   y = y - (y - cos(y))./(1 + sin(y)) ;
end
endfunction
```

Question 11 *Regarder l'évolution de l'erreur en utilisant* ***pointfixeErreur.sci*** *et comparer avec la méthode naïve.*

On trace pour quelques valeurs ($x_0 = 0$ en noir, $x_0 = 0.5$ en bleu et $x_0 = 1$ en vert) l'évolution de l'écart entre x_i et la limite attendue.



On vérifie bien sur ces exemples la convergence vers la valeur attendue. Notons aussi que la méthode de Newton converge plus rapidement que la méthode naïve.

Question 12 *Modifier le fichier* ***pointfixeVitesseConvergence.sci*** *afin de vérifier si la méthode de Newton est d'ordre 2 ou non. Les résultats obtenus sont ils conforme aux attentes ?*

Pour vérifier si la méthode de Newton est d'ordre 2 ou non. Il faudrait tout d'abord vérifier que le ratio $\frac{|e_{i+1}|}{|e_i|}$ tende vers 0 indépendamment de la valeur initiale choisie puis érifier que le ratio $\frac{|e_{i+1}|}{|e_i|^2}$ tende vers un réel strictement positif indépendamment de la valeur initiale. Dans ce second cas, il faudrait modifier le fichier **pointfixeVitesseConvergence.sci** de la manière suivante

```
function [ ratio ] = pointfixeVitesseConvergence(x0)
err = pointfixeErreur(x0) ;
for i = 1:49
   ratio(i) = abs(err(i+1))./(abs(err(i)).^2) ; // err(i) correspond a abs(l - x_i)
end
endfunction
```

Cependant, le programme ne va pas afficher de résultats conforme à ce qu'on attend car très rapidement $e_i = 0$ (au bout de 2 ou 3 itérations) d'où une division par 0 pour le calcul du ratio. On ne peut donc pas vérifier ici que la méthode de Newton est d'ordre 2. Néanmoins, on s'en convainquera du fait de la plus grande rapidité dans la convergence (comparé à celle de la méthode naïve).