TP 2

# Révisions. Les fonctions en Scilab.



**Compétences attendues.**

- ✓ Définir une fonction en Scilab.
- ✓ Tracer la courbe représentative d'une fonction.


Mathieu Mansuy - Professeur de Mathématiques en ECS2 au Lycée Louis Pergaud (Besançon)
`mansuy.mathieu@hotmail.fr`

# 1 Les fonctions

## Définition.

- **Fonctions usuelles :** `log` (logarithme népérien ln), `exp`, `cos`, `sin`, `tan`, `sqrt` (racine carrée), `floor` (partie entière) et `abs`.

- **Syntaxe pour définir une fonction réelle d'une variable réelle $f$ :**

```
function y=f(x)
      y=...
endfunction
```

- **Plus généralement, syntaxe pour définir une fonction SciLab :**

```
function [y1,y2,...,yp]=nom_fonction(x1,x2,...,xn)
      instructions
endfunction
```

**Remarque.** Le plus souvent, on définit une fonction dans un fichier se terminant par .sci[1]. Le nom du fichier doit obligatoirement être le même que le nom de la fonction éditée dans le script.

**Exemple.** Pour définir la fonction $x \mapsto (1+3x+2x^2-x^4)e^{-x}$, taper dans un fichier `P.sci` :

```
function y=P(x)
      y=(1+3*x+2*X^2-x^4)*exp(-x)
endfunction
```

Pour charger la fonction dans `Scilab`, taper la commande :

```
-->exec("P.sci")
```

On peut alors utiliser cette fonction dans la console comme s'il s'agissait d'une autre fonction `Scilab`. Taper par exemple :

```
-->P(1),P(-1),P(sqrt(3))
```

### Remarques

- Une fois qu'une fonction a été chargée (le script qui la contient a été sauvé et exécuté), elle est disponible jusqu'à la fin de la session `Scilab`.
- Les variables utilisées dans le corps d'une fonction sont des variables locales : elles n'existent pas à l'extérieur de la fonction.

[1] Ce n'est pas obligatoire mais cela permet d'identifier les scripts de fonctions des autres scripts.

**Mise en garde.**

Inutile d'ajouter les fonctions d'entrée et de sortie `input` et `disp`. L'utilisateur est sensé connaitre l'argument à rentrer pour la fonction. Scilab lui, renverra le contenu des variables de sorties `y` ou `[y1,y2,...,yp]`.

Dans l'exemple, l'utilisateur rentre donc un réel à la fonction `P`, et Scilab renvoie le contenu de la variable `y` en sortie.

**Exemple.** Entrer la fonction suivante.

```
// Plus grand et plus petit de deux nombres
// m: le plus petit, M: le plus grand
function  [m,M] = min_max(x,y)
if x<=y then
      m=x;
      M=y;
      else m=y;
           M=x;
end
endfunction
```

Enregistrer et charger cette fonction. Exécuter cette fonction sur l'exemple suivant :

```
-->min_max(2.3,-4.7)
```

**Remarque.** Si la fonction possède plusieurs paramètres de sortie, la syntaxe `nom_fonction(valeurx1,...,valeurxn)` ne rend que la valeur de y1. Pour obtenir toutes les valeurs de sortie, il faut utiliser la syntaxe `[nomvar1,...,nomvarp]=nom_fonction(valeurx1,...,valeurxn)`.

**Exemple.** Taper l'instruction suivante :

```
-->[m,M]=min_max(2.3,-4.7)
```

**Remarques.**

- Les paramètres d'entrée `x1,x2,...,xn` et les paramètres de sortie `y1,y2,...,yp` peuvent être de tout type (réel, complexe, vecteur, matrice, etc...).
- Si la fonction ne nécessite pas de paramètre de sortie (la fonction ne fait que des actions), écrire `function []=nom_fonction(x1,x2,...,xn)`.
  Si la fonction ne nécessite pas de paramètre d'entrée, écrire `function y=nom_fonction()`.

# 2 Les représentations graphiques

## 2.1 Fonction `plot2d`

Pour représenter une courbe, `Scilab` trace des points et les relie par des lignes droites.

**Définition.**

Soit `x` et `y` deux vecteurs-lignes (ou deux vecteurs-colonnes) de même taille.
`plot2d(x,y)` trace une ligne brisée entre les points dont les abscisses sont données par le vecteur `x` et les ordonnées sont données par le vecteur `y`.

**Exemple.** Taper les instructions suivantes.

```
-->x=[-1, 0, 1, 2]; y=[2, -1, 0, 3]; plot2d(x,y)
```

**Remarques.**

- On peut aussi utiliser l'instruction `plot` qui diffère de `plot2d` uniquement par les options possibles. Consulter l'aide pour cela.

  Dans la pratique, plutôt que d'utiliser des options, on effectuera les modifications désirées directement dans la fenêtre graphique.

- Si on trace successivement plusieurs courbes, elles se superposent dans la même fenêtre.
  Pour supprimer les graphiques précédents, on utilise l'instruction `clf()` (clear figure).

- Si on désire superposer plusieurs représentations graphiques, on peut utiliser
  `plot2d(x,[y1,y2,...,yn])` où `x,y1,y2,...,yn` sont des vecteurs-**colonnes**.
  On trace ainsi les lignes brisées correspondant aux couples `(x,y1)`, `(x,y2)`, ..., `(x,yn)` sur un même graphique.

## 2.2 Tracé de la courbe représentative d'une fonction

Pour tracer la courbe représentative d'une fonction $f$ sur un intervalle $[a,b]$, on peut utiliser l'instruction `plot2d`. Pour cela, on précise le vecteur `x` des abscisses à considérer et le vecteur `y` des images par $f$ :

- Pour le vecteur des abscisses, on prend une « discrétisation » de l'intervalle $[a,b]$ avec un pas petit. Par exemple si $[a,b]=[0,1]$, on peut taper l'instruction (pour un pas de $1/100$) :

  ```
  x=linspace(0,1,100)
  ```

- Pour le vecteur des images par $f$ :
  - Si la fonction $f$ est usuelle ou si sa définition autorise un vecteur en paramètre d'entrée alors on peut l'appliquer directement au vecteur `x`.
  - Sinon, il faut l'appliquer à `x` à l'aide de `feval`.

**Exemples.** Taper les instructions suivantes.

```
-->x=linspace(-1,3,100); y=exp(x); plot2d(x,y)
```

Taper ensuite les instructions suivantes :

```
-->clf()
-->x=linspace(-1,3,100); y=P(x); plot2d(x,y)
```

On notera alors un message d'erreur. Pourquoi ?

Taper l'instruction suivante :

```
-->x=linspace(-1,3,6); y=feval(x,P); plot2d(x,y)
```

**Remarque.** Il est possible d'éviter le message d'erreur en changeant la définition de $P$, et en autorisant les opérations pour les vecteurs :

```
1 function y=P(x)
2       y=(1+3*x+2*x.^2-x.^4).*exp(-x)
3 endfunction
```

Généralement, il est plus simple d'utiliser l'instruction `fplot2d`.

**Définition.**

Soit `x` un vecteur et `f` une fonction.
`fplot2d(x,f)` trace une ligne brisée entre les points dont les abscisses sont données par le vecteur `x` et les ordonnées sont les images de `x` par `f`.

**Exemple.** Taper les instructions suivantes.

```
-->clf()
-->x=linspace(-1,3,100); fplot2d(x,P)
```

**Remarque.** Dans le cas où plusieurs représentations graphiques se superposent, on peut les différentier par couleur en utilisant par exemple `plot2d(x,y,3)` ou `fplot2d(x,P,5)`

# 3 Exercices

## Exercice 3.1 (★)

Compléter la fonction suivante afin qu'elle retourne le couple $a, b$, où $a = \sum_{i=1}^{n} \sqrt{i}$ et $b = \sum_{i=1}^{n} \lfloor \ln(i) \rfloor$.

```
function [a,b] = sommes(n)
      a = ...
      b = ...
endfunction
```

On utilise les opérations pointées, coefficients par coefficients, sur les matrices :

```
function [a,b] = sommes(n)
      a = sum(sqrt(1:n))
      b = sum(floor(log(1:n)))
endfunction
```

## Exercice 3.2 (★★)

1. Montrer que la série $\sum \frac{(-1)^n}{n^2}$ est convergente.
2. Créer un vecteur ligne `u`, tel que pour tout $1 \leq i \leq 50$, `u(i)` soit égal à $\frac{(-1)^i}{i^2}$. *On pourra utiliser pour cela une boucle for.*
3. Créer un vecteur `v` tel que pour $1 \leq i \leq 50$, `v(i)` soit égal à $\sum_{k=1}^{i} \frac{(-1)^k}{k^2}$.
4. Illustrer la convergence de la série en représentant graphiquement la suite de ses sommes partielles, et donner une valeur approchée à $10^{-1}$ près de $\sum_{n=1}^{+\infty} \frac{(-1)^n}{n^2}$.

1. Elle est absolument convergente (car la série $\sum \frac{1}{n^2}$ converge en tant que série de Riemann avec $\alpha = 2 > 1$), donc convergente.

2. On peut procéder ainsi :

```
//Méthode 1 avec boucle for
u = []
for k=1:50
    u = [u,((-1)^k)/(k^2)]
end
//Méthode 2 avec opérations pointées
u = ((-1).^(1:50))./((1:50).^2)
```

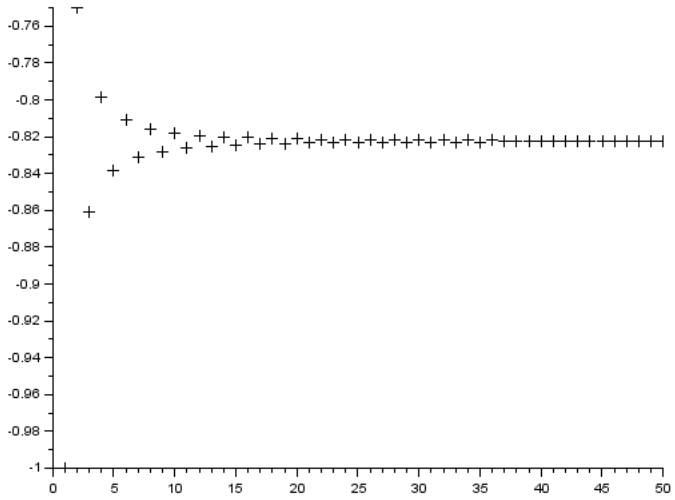
3. On utilise la commande `cumsum` :

```
v = cumsum(u)
```

4. On va placer en abscisse les entiers $n$ de 1 à 50, et en ordonnée la valeur de la somme partielle $S_n$ correspondante. On utilise pour cela la ligne de commandes suivante (l'argument $-1$ permet de ne pas relier les points) :

```
plot2d(1:50,v,-1)
```

On obtient le graphe suivant :




Par lecture graphique, on obtient que la somme de la série est environ égale à 0,82.

**Déjà vu ?**

Nous avons plusieurs fois rencontré des séries alternées en TD (série harmonique alternée) ou en DM. On a à chaque fois montré que les séries extraites paires et impaires de $(S_n)$ sont adjacentes, l'une croissante, l'autre décroissante, et qu'elles tendent vers la somme de la série. C'est ce qu'on remarque aussi ici graphiquement. Pour plus de détails sur les séries alternées, je vous renvoie au :

☞ **Complément de cours 1. Autour des séries alternées.**

---

## Exercice 3.3 (★)

1. Soit $(u_n)_{n\in\mathbb{N}}$ la suite définie par $u_0 = 1$ et pour tout $n \in \mathbb{N}$, $u_{n+1} = 2nu_n + 3$. Écrire une fonction en `Scilab` ayant pour paramètre un entier $n \geq 0$ et qui renvoie la valeur de $u_n$.
2. Même question avec $(u_n)_{n\in\mathbb{N}}$ définie par $u_0 = 1$, $u_1 = -2$ et pour tout $n \in \mathbb{N}$, $u_{n+2} = 2u_{n+1} - u_n$.

1. On va faire une boucle `for` :

```
function u = suite(n)
	u = 1
	for k=1:n
		u = 2*(k-1)*u+3
	end
endfunction
```

2. Voici un premier programme :

```
function v = suite(n)
	u = 1
	v = -2
	for k=2:n
		w = 2*v-u
		u = v
		v = w
	end
endfunction
```

Ce programme donne le bon résultat pour tout $n \geq 1$, mais pas pour $n = 0$ où il renvoie $u_1$ au lieu de $u_0$. On peut proposer l'amélioration suivante :

```
function v = suite(n)
	u = 1
	v = -2
	if n == 0 then
		v = 1
		else
		for k=2:n
			w = 2*v-u
			u = v
```

```
                    v = w
                end
            end
endfunction
```

**Exercice 3.4 (★)**

Représenter sur une même figure les représentations graphiques des fonctions sin, $x \mapsto x$, $x \mapsto x - \dfrac{x^3}{3!}$, $x \mapsto x - \dfrac{x^3}{3!} + \dfrac{x^5}{5!}$ sur l'intervalle $[-\pi, \pi]$. Que remarque-t-on ?

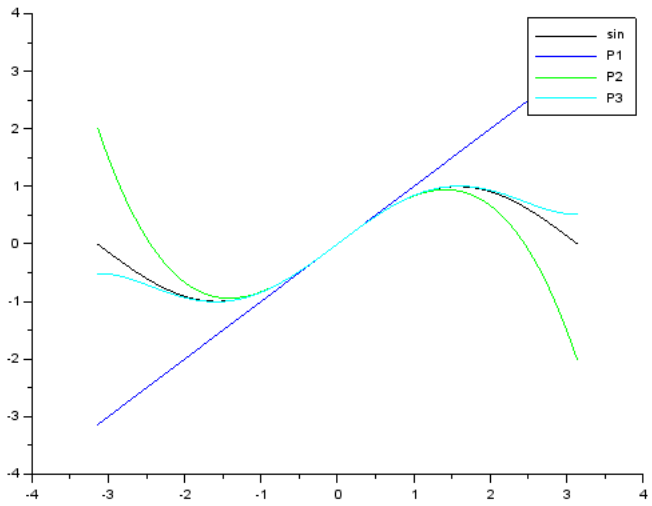
```
function y = P1(x)
    y = x
endfunction

function y = P2(x)
    y = x-(x.^3)/6
endfunction

function y = P3(x)
    y = x-(x.^3)/6+(x.^5)/120
endfunction

x=linspace(-%pi,%pi,100)
plot2d(x,sin(x),1)
fplot2d(x,P1,2)
fplot2d(x,P2,3)
fplot2d(x,P3,4)
hl=legend(['sin';'P1';'P2';'P3']);//Pour ajouter une légende
```

On obtient le graphe suivant :



On a ici sur un même graphe la courbe de sinus et de ses parties principales des développements limités d'ordre 1, 3 et 5. On remarque que plus l'ordre est grand, plus cette partie principale « colle » la courbe de sinus **au voisinage de 0**. On notera également que cette approximation de sinus par un polynôme n'est « intéressante » qu'au voisinage de 0, puisque les courbes des parties principales du développement limité s'éloignent de celle du sinus quand on n'est plus au voisinage de 0.

---

**Exercice 3.5 (Fonction réciproque - ★★)**

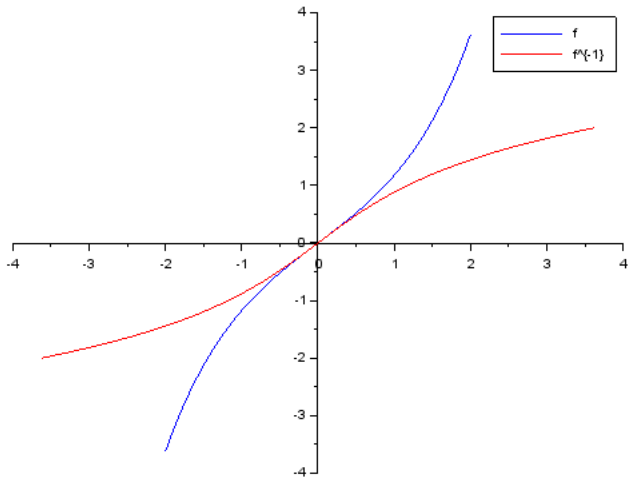
1. On considère la fonction $f : \mathbb{R} \to \mathbb{R}$ définie par $f(x) = \dfrac{e^x - e^{-x}}{2}$ (appelée *sinus hyperbolique*, et souvent notée *sh*). Montrer que $f$ réalise une bijection de $\mathbb{R}$ dans $\mathbb{R}$.
2. Définir une subdivision de l'intervalle $I = [-2, 2]$ en 100 sous-intervalles de même longueur.
3. Écrire une ligne de commandes permettant de tracer sur un même graphique la courbe représentative de la fonction $f$ définie sur $I$ ainsi que celle de sa fonction réciproque (sans chercher à déterminer cette dernière).

1. $f$ est continue sur $\mathbb{R}$ comme composée de fonctions continues. Elle est de plus dérivable pour les mêmes raisons, et on a :
$$\forall x \in \mathbb{R}, \quad f'(x) = \frac{e^x + e^{-x}}{2} > 0.$$
Donc $f$ est strictement monotone sur $\mathbb{R}$, et on a $\lim\limits_{x \to \pm\infty} f(x) = \pm\infty$. Par le théorème de la bijection, $f$ réalise une bijection de $\mathbb{R}$ sur $\mathbb{R}$.
2. On peut utiliser la commande `x = linspace(-2,2,100)`.
3. On sait que le graphe de $f^{-1}$ est le symétrique de celui de $f$ par rapport à la droite $y = x$. On utilise ce résultat ici :

```
function y = f(x)
      y = (exp(x)-exp(-x))/2
endfunction

x = linspace(-2,2,100)
y = feval(x,f) // donne le vecteur image des éléments de x par f
plot2d(x,y,2) // courbe de f
plot2d(y,x,5) // courbe de f^{-1}
hl=legend(['f';'f^{-1}']);//Pour ajouter une légende
```

On obtient le graphe suivant :




## Exercice 3.6 (★★ - Étude d'une suite récurrente)

1. Soit $f$ la fonction définie par $f(x) = \dfrac{2}{x} + \ln x$.
   (a) Montrer que l'intervalle $\left[\dfrac{3}{2}, 2\right]$ est stable par $f$. On donne $1 + \ln 2 \approx 1,69$ et $\frac{4}{3} + \ln\left(\frac{3}{2}\right) \approx 1,74$.
   (b) Montrer qu'il existe un unique $\alpha \in \left[\dfrac{3}{2}, 2\right]$ tel que $f(\alpha) = \alpha$.
   Tracer le graphe de $f$ ainsi que la droite d'équation $y = x$ à l'aide de `Scilab`, et déterminer graphiquement une valeur approchée de $\alpha$.
   (c) Montrer que pour tout $(x, y) \in \left[\dfrac{3}{2}, 2\right]^2$, $|f(x) - f(y)| \leq \dfrac{2}{9}|x - y|$.
2. Soit $(u_n)_{n\in\mathbb{N}}$ la suite définie par $u_0 = 2$ et pour tout $n \in \mathbb{N}$, $u_{n+1} = f(u_n)$.
   (a) Montrer que la suite $(u_n)_{n\in\mathbb{N}}$ est bien définie et pour tout $n \in \mathbb{N}$, $u_n \in \left[\dfrac{3}{2}, 2\right]$.
   (b) Montrer que pour tout $n \in \mathbb{N}$, $|u_{n+1} - \alpha| \leq \dfrac{2}{9}|u_n - \alpha|$ puis que $|u_n - \alpha| \leq \left(\dfrac{2}{9}\right)^n |u_0 - \alpha|$.
   (c) En déduire que la suite $(u_n)_{n\in\mathbb{N}}$ converge vers $\alpha$.
3. (a) Écrire une fonction en `Scilab` ayant pour paramètre d'entrée $n \geq 0$ et qui renvoie la valeur de $u_n$.
   (b) Écrire une fonction `Scilab` qui donne une approximation de $\alpha$ à $\varepsilon$ près pour $\varepsilon > 0$ donné.

1. (a) On étudie la fonction $f$. Elle est continue et dérivable sur $\mathbb{R}_+^*$ comme composées de fonctions continues sur cet intervalle, et on a pour tout $x > 0$ :
$$f'(x) = -\frac{2}{x^2} + \frac{1}{x} = \frac{x-2}{x^2}.$$
Donc $f$ est strictement décroissante sur l'intervalle $[3/2, 2]$. De plus, on a $f(2) > 3/2$ et $f(3/2) < 2$. Donc on a bien que pour tout $x \in [3/2, 2]$, $f(x)$ appartient à $[3/2, 2]$. Cet intervalle est donc stable par $f$.
   (b) Posons la fonction $g : x \in [3/2, 2] \mapsto f(x) - x$. $g$ est continue et dérivable comme

composée de fonctions qui le sont, et on a :

$$\forall x \in [3/2, 2], \quad g'(x) = f'(x) - 1 = -\frac{x^2 - x + 2}{x^2} < 0$$

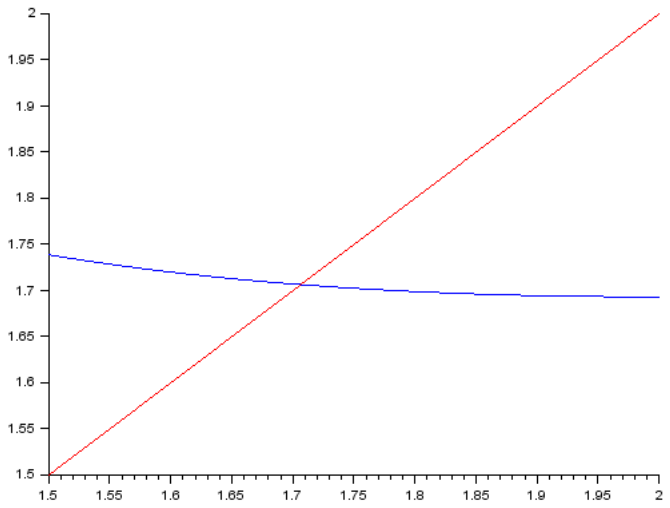
car le discriminant du polynôme au numérateur est égal à $(-1)^2 - 4 \times 2 = -7 < 0$. $g$ est donc strictement décroissante sur cet intervalle, et on a $g(3/2) > 0$ et $g(2) < 0$. Par le théorème de la bijection, l'équation $g(x) = 0 \Leftrightarrow f(x) = 0$ admet une unique solution $\alpha \in ]3/2, 2[$.

Pour tracer le graphe de $f$ et la droite d'équation $y = x$ sur l'intervalle $[3/2, 2]$, on peut utiliser le code suivant :

```
function y =f(x)
    y = (2./x)+log(x)
endfunction

x = linspace(3/2,2,100)
y = x
fplot2d(x,f,2)
plot2d(x,x,5)
```

On obtient le graphe suivant :




Par lecture graphique, on obtient $\alpha \approx 1,71$.

(c) Cette inégalité est clairement du « type » inégalité des accroissements finis. On l'applique donc ici : $f$ est continue sur $[3/2, 2]$ et dérivable sur $]3/2, 2[$, et on a :

$$\forall x \in ]3/2, 2[, \quad |f'(x)| = \frac{|x-2|}{x^2} \leq \frac{1/2}{(3/2)^2} = \frac{2}{9}.$$

Par l'inégalité des accroissements finis, on obtient que :

$$\forall x, y \in [3/2, 2], \quad |f(x) - f(y)| \leq \frac{2}{9}|x - y|.$$

2. (a) On le montre par récurrence sur $n \in \mathbb{N}$.

**Init.** C'est vrai pour $n = 0$ car $u_0 = 2$.

**Hér.** Soit $n \in \mathbb{N}$. On suppose que la propriété est vraie au rang $n$. Montrons la au rang $n+1$.

On a $u_n \in [3/2, 2]$ par hypothèse. D'où $u_{n+1} = f(u_n)$ est bien définie (on reste dans le domaine de définition de $f$) et appartient à $[3/2, 2]$ car cet intervalle est stable par $f$. D'où la propriété au rang $n+1$.

**Concl.** Par principe de récurrence, $u_n$ est bien définie et appartient à $[3/2, 2]$ pour tout $n \in \mathbb{N}$.

(b) On applique l'inégalité du 1.$(c)$ avec $x = u_n$ et $y = \alpha$ :

$$|u_{n+1} - \alpha| = |f(u_n) - f(\alpha)| \leq \frac{2}{9}|u_n - \alpha|.$$

On montre alors par une récurrence immédiate (je vous laisse le soin de la rédiger) que pour tout $n \in \mathbb{N}$, on a :

$$|u_n - \alpha| \leq \left(\frac{2}{9}\right)^n |u_0 - \alpha| \leq \frac{1}{2}\left(\frac{2}{9}\right)^n$$

car $u_0, \alpha \in [3/2, 2]$.

(c) On a $\lim\limits_{n\to+\infty} \left(\frac{2}{9}\right)^n = 0$. Par théorème des gendarmes, on en déduit que $\lim\limits_{n\to+\infty} u_n$ existe et vaut $\alpha$.

3. (a) On a déjà défini la fonction `f` plus haut. Utilisons la ici :

```
function u = suite(n)
    u = 2
    for k=1:n
        u = f(u)
    end
endfunction
```

(b) On a vu que $|u_n - \alpha| \leq \frac{1}{2}\left(\frac{2}{9}\right)^n$ pour tout $n \in \mathbb{N}$. Cherchons donc $n$ tel que $\frac{1}{2}\left(\frac{2}{9}\right)^n \leq \varepsilon$. On résout :

$$\left(\frac{2}{9}\right)^n \leq 2\varepsilon \Leftrightarrow n\ln(2/9) \leq \ln(2\varepsilon) \Leftrightarrow n \geq \frac{\ln(2\varepsilon)}{\ln(2/9)}.$$

On prendra donc pour $n$ la valeur $\lfloor \frac{\ln(2\varepsilon)}{\ln(2/9)} \rfloor + 1$. On obtient le programme suivant :

```
function alpha = approx(eps)
    n = floor(log(2*eps)/log(2/9))
    alpha = suite(n)
endfunction
```

En exécutant la commande `approx(0.001`, on obtient le résultat `1.706454`.

## Exercice 3.7 (Fonction de répartition de la loi normale centrée réduite - ★★★)

Pour tout $x \in \mathbb{R}$, on note $\varphi(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right)$ la densité de probabilité de la loi normale centrée

réduite $\mathcal{N}(0,1)$, et $\Phi$ sa fonction de répartition, définie par :

$$\forall x \in \mathbb{R}, \quad \Phi(x) = \int_{-\infty}^{x} \varphi(t)\,\mathrm{d}t.$$

1. Créer la fonction $\varphi$ en `Scilab` (on permettra que la variable d'entrée soit un vecteur). Représenter-la graphiquement.

2. (a) On rappelle que si $f$ est une fonction continue sur $[a,b]$ alors on a :

$$\int_a^b f(t)\,\mathrm{d}t = \lim_{x\to+\infty} \frac{b-a}{n} \sum_{k=1}^{n} f\left(a + k\frac{b-a}{n}\right).$$

Soit $x \in [0,+\infty[$. Pour $a$ « petit » et $n$ « grand » avec $\dfrac{x-a}{n}$ « petit », on a alors :

$$\frac{x-a}{n} \sum_{k=1}^{n} \varphi\left(a + k\frac{x-a}{n}\right) \approx \int_{-\infty}^{x} \varphi(t)\,\mathrm{d}t = \Phi(x).$$

Écrire une fonction `Phi` en `Scilab` qui prend comme argument des réels $x$ et $a$ et un entier $n$, et renvoie une approximation de $\Phi(x)$.

(b) En utilisant la fonction `Phi`, tracer la représentation graphique de $\Phi$ (choisir $a$ et $n$ de façon « adéquate »).

3. On va vérifier graphiquement la qualité de notre représentation graphique. Pour cela, on utilise la commande `Scilab` suivante.

`[P,Q]=cdfnor("PQ",x,m,sigma)` avec `x`, `m` et `sigma` vecteurs de même taille, donne $P = F(x)$ où $F$ est la fonction de répartition de la loi $\mathcal{N}(m,\sigma^2)$, et $Q = 1 - P$.

(a) Calculer $\Phi(0), \Phi(1), \Phi(1,96)$.

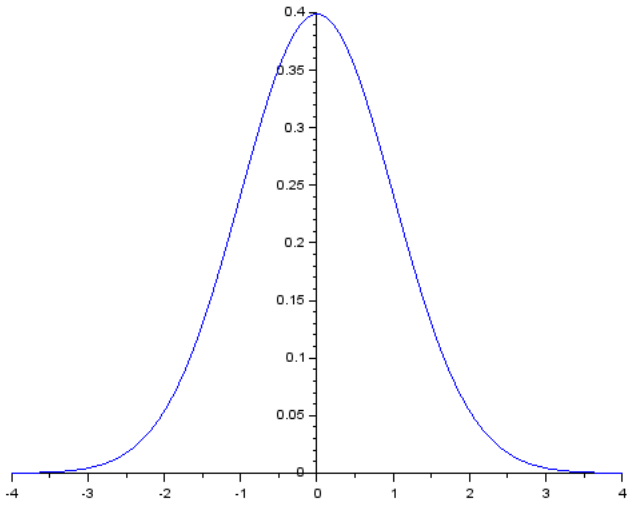
(b) Tracer la fonction de répartition $\Phi$ de la loi normale centrée réduite à l'aide de la commande `cdfnor`. Comparer les deux courbes (faites varier les paramètres $a$ et $n$).

1. On peut utiliser le script suivant :

```
function y = phi(x)
    y = (1/(sqrt(2*%pi)))*exp(-(x.^2)/2)
endfunction

x = linspace(-4,4,100)
fplot2d(x,phi,2)
```

qui renvoie le graphe suivant :




2. (a) On propose le script suivant :

```
function y = Phi(x,a,n)
    h = (x-a)/n)
    y = h*sum(phi(a+h*(1:n)))
endfunction
```

Cette fonction renvoie une approximation de $\Phi$ à condition de choisir $a$ « petit » et $n$ « grand ».

(b) On peut utiliser le script suivant (on utilise une boucle `for` pour calculer le vecteur image).

```
a = -4
n = 50
x = linspace(-4,4,100)
y = []
for k=1:100
    y = [y,Phi(x(k),a,n)]
end
plot2d(x,y,5)
```

3. (a) On exécute les commandes suivantes :

```
[P1,Q1] = cdfnor("PQ",0,0,1)
[P2,Q2] = cdfnor("PQ",1,0,1)
[P3,Q3] = cdfnor("PQ",1.96,0,1)
disp(P1,P2,P3)
```

On obtient les résultats suivants : $0,5$, $0,8413447$ et $0,9750021$. Notez que la première et la dernière valeur étaient prévisibles, puisqu'on sait que $\Phi(0) = 0,5$ et que $\Phi(1,96) \approx$

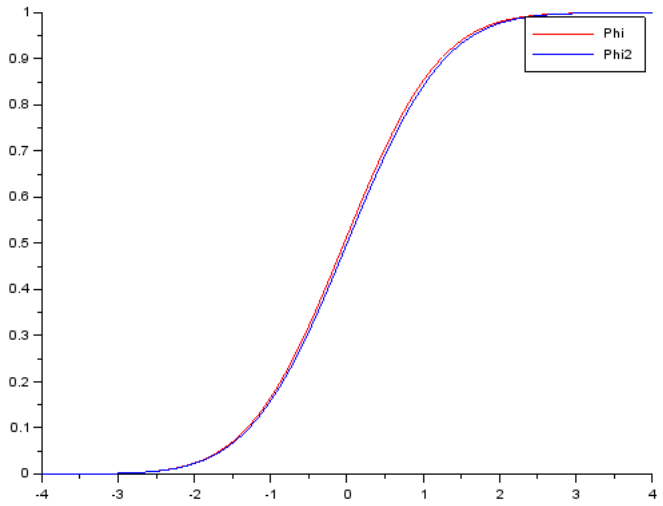
$0,975$ (souvenez vous des intervalles de confiances que vous avez découvert en terminale !).

(b) On trace les deux fonctions sur le même graphique à l'aide du script suivant :

```
function y = Phi2(x)
    [P,Q] = cdfnor("PQ",x,0,1)
    y = P
endfunction

x = linspace(-4,4,100)
y = []
for k=1:100
    y = [y,Phi2(x(k))]
end
plot2d(x,y,2)
hl=legend(['Phi';'Phi2']);//Pour ajouter une légende
```

On obtient le graphe suivant :




Pour une meilleur précision, on peut augmenter la valeur de $n$ : ici $n = 50$. Pour $n = 100$, les courbes sont confondues sur la fenêtre graphique.