Méthodes Numériques.

Licence 3ème année, UFR 27

D. PENNEQUIN[1]

Année Universitaire 2015-2016

[1]UFR Mathématiques et Informatique, Université Paris 1, 90 rue de Tolbiac, 75647 Paris CEDEX, France. E-mail : pennequi@univ-paris1.fr

Table des Matières

Introduction.
Ce cours intermédiaire entre l'Analyse Numérique et l'Informatique a pour buts de vous introduire au calcul numérique et d'illustrer les cours de mathématiques.

Remerciements : les auteurs remercient J-M. Bardet et S. Liu pour les éléments qu'ils ont apporté dans la rédaction du polycopié.

Bibliographie : mis à part des cours de mathmatiques gnrales, il est possible d'aller voir des livres d'analyse numérique, calcul scientifique, en faisant le tri. Voici à titre d'information quelques références possibles :

- Chancelier J-P. et alii. : Introduction à SCILAB, Lb 31.3INT
- Ciarlet P-G. : Introduction l'Analyse Numrique Matricielle et l'Optimisation (command par la bibliothque)
- Culioli J-C. : Introduction l'Optimisation (command par la bibliothque)
- Connan G. : Guide du calcul scientifique avec les logiciels libres ... : Lb31.3CON
- Demailly, J-P. : Analyse Numérique et Equations différentielles, Lb0DEM
- Guerre-Delabrière S. : Méthodes d'approximation, équations différentielles, applications SCILAB, Lb31.3GUE
- Marco et alii. : Cours de mathématiques L1 et L2 (maths générales) ; le chapitre 13 du L2 contient une partie commune avec ce cours. Lb0MAT
- Merrien J-L. : Analyse numérique avec MATLAB, Lb31.3MER
- Quateroni A. : Calcul scientifique ..., Lb31.3QUA
- Schatzmann M. : Numerical Analysis ; a mathematical introduction, Lb0SCH
- Stoer J., Bulirsch : Introduction to numerical Analysis, Lb0STO
- Vial G. ; voir sa page web sur le site de l'ENS Cachan, qui contient des complments de cours ainsi que des programmes utiles.
- Yger A., Weil J-A. et alii. : Mathmatiques appliques, L3, Pearson. L0MAT

Chapitre 1

Initiation à SCILAB (version 5).

Concernant l'utilisation de Scilab, vous pourrez également lire le résumé figurant dans le tome L3 de Pearson. Vous trouverez également beaucoup de documentation sur internet.

1.1 Introduction.

Il est tout d'abord possible d'utiliser SCILAB de manière interactive. Dans ce cas, SCILAB effectue directement les opérations que vous lui demandez. Il est ici possible de passer en ligne de commande des commandes systèmes. La première que nous allons passer concerne un changement de répertoire, afin que SCILAB sauvegarde nos fichiers dans son propre répertoire. Pour se déplacer dans son propre répertoire, on utilise la commande DOS ou UNIX usuelle `cd`.

Ainsi, **vous devez toujours au démarrage de SCILAB vous placer dans un répertoire que vous serez préalablement créé sur le lecteur logique D:**
Si par exemple vous vous êtes créé un répertoire `D:\L3\dupont`, vous devrez taper au début de chaque séance :
$>$ `chdir("D:\L3\dupont")`
Le symbole $>$ représente ce que l'on appelle *une invite* (elle est symbolisée par $-->$ dans SCILAB). Elle n'est pas à taper, elle signifie simplement que SCILAB attend que vous entriez des commandes.

Vous pouvez vérifier que vous êtes dans le bon répertoire en tapant la commande `pwd`, qui affiche le répertoire courant. Ces commandes sont également accessibles dans le menu `File` (`Change Directory`, ou `Get Current Directory`).

En utilisation interactive, il n'est pas possible d'enregistrer directement ce que l'on fait. On peut cependant enregistrer une session dans un fichier au format texte, qui contiendra nos entrées et les réponses de SCILAB. Pour enregistrer une séance dans un fichier (par exemple avec le nom `essai.txt`), on commence dès

le début à taper :
> `diary('essai.txt')` `[Entrée]`
(Désormais, on sous-entendra `[Entrée]`), puis à la fin de la séance, on tape :
> `diary(0)`
pour enregistrer sur le disque dans le répertoire par défaut.

Tout ce qui est compris entre ces deux commandes est mis dans un fichier texte, tel une photocopie. Ce fichier contient en particulier les entrées (y compris les invites) et les sorties. Si vous voulez le transformer en un *fichier script* afin de le rendre executable (cf. plus loin), vous devrez en particulier le débarasser de ses invites.

Il y a une liste de commandes à connaître absolument avant tout apprentissage de SCILAB :

- `//` permet d'entrer un commentaire. A partir de ce caractère, plus rien n'est interprété par SCILAB.

- `help (suivi d'un nom de fonction)` : se passe d'explications !

Commençons par étudier notre environnement de travail. Il est possible de passer certaines commandes MSDOS si vous êtes sur un système Microsoft, ou des commandes Unix à l'aide de la commande `unix` et ses dérivées (survolez `apropos unix`). Signalons par exemple la commande `unix_w` qui donne un résultat en sortie ou `unix_s` qui exécute sans afficher. Par exemple :
`unix_s('cd monrep')` : passe dans le répertoire nommé `monrep`.
`unix_s('copy ...')` sous Windows ou `unix_s('cp ...')` sous Unix fait une copie de fichiers (je suppose que vous connaissez les syntaxes de ces commandes).

On dispose aussi de commandes pour les variables qui sont dans le répertoire courant :
`clear var` : supprime la variable nommée `var`.
`clear` : supprime toutes les variables.
`who` : liste les variables. Une version avec renseignements sur les variables est `whos()`. Cette commande est aussi accessible depuis le menu `Applications, Browser Variables`.

On peut utiliser les racoucis Emacs, où le raccourci `C+` signifie que l'on appuie sur la touche Control et tout en la maintenant enfoncée, on appuie sur l'autre touche :
- `C+p` (ou ↑) rappelle la ligne précédente (p=previous).
- `C+n` rappelle la ligne suivante (n=next).
- `C+f` (ou →) avance d'un caractère (f=forward).

- `C+b` (ou ←) recule d'un caractère (b=backward).
- `C+d` supprime un caractère au niveau du curseur (d=delete).
- `C+a` va au début de la ligne.
- `C+e` va à la fin de la ligne.

Ces commandes sont également accessibles via le menu `Edit, History`.

Attention : évitez l'emploi du copier-coller brutal. Si jamais vous l'utilisez, n'oubliez pas d'enlever les éventuelles invites, cela évitera des désagréments. Mais en général, il est beaucoup plus efficace de rappeler une ligne antérieure à l'aide des raccourcis indiqués ci-dessus.

1.2 Premières opérations sur les réels et complexes.

Entrez ces lignes à l'invite de SCILAB. Déduisez-en les règles de priorité pour les calculs.

```
> 1+2*3
> (1+2)*3
> 2^(1/2)
> 2^1/2
```

Au vu des résultats précédents et d'autres que vous auriez pu essayer, proposez des règles de priorité dans SCILAB.

Passons maintenant aux opérateurs de division. On dispose de / et de \. Essayez les commandes suivantes et commentez :

```
> 2/3
> 2\3
> 3\2
```

Vous pouvez disposer ces commandes à la suite sur une même ligne, à condition de les séparer par une virgule. L'effet produit par ce qui suit est identique :

```
> 2/3, 2\3, 3\2
```

Par défaut, le format des nombres est fixée à 8 chiffres significatifs. Pour changer ce format, on écrit `format("v",n)`, où $n-2$ est le nombre de chiffres significatifs maximal[1] souhaités (donc par défaut, n vaut 10). On peut aussi écrire les nombres

[1]Le comportement de `format` est quelque peu curieux ; attention de grandes valeurs de n

en format mantisse-exposant. Pour tout réel non nul x, on peut trouver un entier relatif n et un réel a tels que $|a| \in [1, 10[$ et $x = a10^n$. a est la mantisse de x et n son exposant. a et x sont de même signe, et a et n se calculent par les formules $n = E(\log_{10}(|x|))$ puis $a = x10^{-n}$, où E désigne la fonction partie entière et $\log_{10}$ le logarithme de base 10 (on verra plus loin comment les calculer). Par exemple, avec 125.23, la mantisse est 1.2523 et l'exposant 2 puisque $125.23 = 1.2523 \times 10^2$. On peut écrire directement dans SCILAB les nombres sous cette forme, en utilisant `format('e',n)` au lieu de `format('v',n)`. Testez ces commandes avec 1234.56789 ; n'oubliez pas de revenir au format par défaut à l'issue en tapant `format('v',10)`.

On dispose de plusieurs fonctions d'arrondis :
`floor(x)` : plus grand entier inférieur ou égal à `x`. C'est la partie entière usuelle en mathématique.
`ceil(x)` : plus petit entier supérieur ou égal à `x`.
`fix(x)` : c'est `floor(x)` si $\texttt{x} \geq 0$, `ceil(x)` sinon.
`round(x)` : entier le plus proche de `x`.

Exercice 1.2.1 *Essayez ces commandes avec 1.23, -1.23, 1.72, -1.72 (par exemple).*

SCILAB dispose de beaucoup de fonctions usuelles (trigonométriques, logarithmiques, exponentielle...) et de fonctions dites spéciales intervenant en ingéniérie (fonctions eulériennes, de Bessel, elliptiques...). A titre d'information, signalons les fonctions trigonométriques `sin, cos, tan, ...` s'appliquant à un angle en radians, et les fonctions logarithmiques `log` (logarithme népérien), `log10` (logarithme décimal), `log2` (logarithme en base 2), `exp` (fonction exponentielle).

Enfin, pour calculer les factorielles, on sera amené à utiliser la fonction Γ. Cette fonction est en fait définie sur $\mathbb{C}\backslash\mathbb{Z}^-$, mais on utilisera surtout le fait que pour tout entier n, $n! = \Gamma(n+1)$, qui se calcule dans SCILAB par `gamma(n+1)` (attention au décalage). Cependant, le calcul n'est pas possible pour de grandes valeurs de `n` ; nous allons utiliser les outils de la notation scientifique et le fait que le logarithme transforme produits en sommes. Ainsi, pour calculer le nombre :

$$P = 500!,$$

il peut être judicieux de constater que :

$$Q = \log_{10}(P) = \sum_{k=1}^{500} \log_{10}(k).$$

aboutissent à des aberrations.

Exercice 1.2.2 *Calculez Q par la commande* `Q=sum(log10(1:500))` *et en déduire l'exposant et la mantisse de P. Combien de chiffres y a t-il dans P ?*

Les calculs précédents s'étendent au cadre complexe. Le nombre complexe i se note `%i`.

```
> (2+3*%i)/(5+2*%i)
```

Passons aux parties réelles, imaginaires, modules et argument.

```
> a=7-2*%i
> real(a) // partie reelle
> imag(a) // partie imaginaire
> abs(a) // module
> phasemag(a) // argument en degres
> conj(a) // complexe conjugué
> a+conj(a)-2*real(a)
```

Les fonctions réelles s'étendent au cas complexe. Par exemple, le sinus est défini pour tout $z \in \mathbb{C}$ par la série convergente :

$$\sin z = \sum_{k=0}^{+\infty} \frac{(-1)^k z^{2k+1}}{(2k+1)!}.$$

```
> sin(2+%i)
```

On fera attention au logarithme complexe et aux puissances non entières qui sont des fonctions multiformes et qui ne vérifient pas toutes les formules du cas réel :

```
> log(%i^4)-4*log(%i)
```

Dans ce cas, on n'obtient pas zéro comme attendu si la formule $\log(a^4) = 4\log(a)$ était vraie. La raison en est la suivante : tout nombre complexe non nul s'écrit de manière unique $z = \rho e^{i\theta}$ avec $\theta \in]-\pi, \pi]$ et on pose alors $\log(z) = \log(\rho) + i\theta$. Au vu de ceci, expliquez le résultat obtenu.

1.3 Affectations et désaffectations.

La variable prédéfinie `ans` contient le dernier résultat calculé et peut être utilisée :

```
> 5-4
> ans+2
```

Rappelez la dernière ligne par la flèche ↑ et ré-exécutez la dernière ligne. Que se passe t-il ? Commentaire.

Il peut être utile d'enregistrer le résultat d'un calcul dans une variable autre que

`ans`. On le fait en mettant le nom de la variable à gauche d'un signe égal qui à droite a le résultat du calcul :

```
>   a=sin(2)
```

Pour ensuite supprimer l'affectation à la variable nommée ici `a`, on utilise l'instruction `clear` :

```
>   clear a
```

Etudiez ce qui suit :

```
>   clear // ôte toutes affectations.
>   a=2
>   a/5
>   A/5
```

Comme le montre le dernier exemple, SCILAB distingue majuscules et minuscules.

1.4 Fichiers de données.

Vous pouvez tout d'abord sauvegarder vos données dans un fichier. Supposons que vous ayez créé deux matrices `A` et `B` et que vous souhaitiez les sauvegarder. La commande `save` enregistre ces matrices dans un fichier nommé ici `data.sav`, ces variables peuvent être récupérés ultérieurement :

```
> A=2, B=sqrt(7)
>   save('data.sav',A,B)
>   clear
> A
>   load('data.dat','A','B')
> A
```

1.5 Editeur et fichiers scripts.

Un fichier script est une suite de commandes SCILAB que vous avez tapé dans votre éditeur de texte favori et sauvegardé dans votre répertoire de travail avec une extension qui est par défaut `.sce`. SCILAB dispose de l'éditeur SCIPAD (que l'on peut ouvrir par le menu `Editor`), mais rien ne vous empêche d'en utiliser un autre si vous le souhaitez. L'appel en ligne de commande de ce fichier provoque alors son exécution.

Par exemple, ouvrez l'éditeur et enregistrez les lignes suivantes dans un fichier intitulé `courbe.sce` :

```
x=-2 :  0.01 :  2 ; y=x.^3-1;
plot(x,y)
```

Ces lignes sont destinées à tracer la fonction $x \mapsto x^3 - 1$ sur $[-2; 2]$. Revenez à l'environnement de travail et tapez :
> `exec("courbe.sce");`
alors vous verrez le graphe se dessiner. La commande `exec` se trouve aussi dans le menu `File`.

Attention :

- après chaque modification d'un script, il faut le charger à nouveau grâce à `exec`,
- dans un fichier script, les variables sont *globales*, ce qui signifie qu'elles sont utilisables une fois le programme exécuté.

Pour expliquer la notion de variable globale, faites l'expérience suivante. On donne à une variable `x` une valeur avant l'execution du script, puis on lance ce script qui lui affecte une autre valeur ; l'ancienne valeur est écrasée par la nouvelle. Essayez et commentez :
> `x=8`
> `exec("courbe.sce");`
> `x`

1.5.1 Création d'une fonction.

Une fonction peut tout d'abord être créée en ligne de commande. Ainsi pour créer la fonction `phi` telle que $phi(z) = z^3 - z + 1$, on écrit :
> `deff('t=phi(z)','t=z.^3-z+1');`
et l'on peut alors utiliser directement la fonction :
> `phi(5)`

Il est à noter cette fois que les variables sont *locales*, l'appel de `phi` ne modifiera pas les anciennes valeurs de `t` et `z` :
> `t=123, z=456`
> `phi(0)`
> `t,z`

Pour une fonction plus longue, on peut la rentrer sur plusieurs lignes, mais en général il est recommandé d'écrire les commandes dans un fichier de fonction, qui est un fichier texte d'extension par défaut `sci`, qui commence par le mot clef `function` et se termine par `endfunction`. Il est recommandé de donner le même nom à la fonction qu'au fichier. Par exemple, le fichier suivant `facto.sci`

retourne la valeur de la factorielle lorsqu'en entrée, l'utilisateur fournit un entier au moins égal à 1.

Nous allons entrer une fonction `facto` dans ces deux modes, permettant le calcul de la factorielle. La première solution est d'entrer en mode interractif :

```
function res=facto(n)
res=prod(1:n);
endfunction
```

On notera que les invites se sont rapprochées, jusqu'à l'entrée du mot clé `endfunction`. La fonction est alors immédiatement utilisable. On remarque que la première ligne est composée du mot clé `function`, suivi de la variable qui va contenir la sortie, puis du signe `=`, du nom de la fonction puis entre parenthèse la variable d'entrée. Ces variables peuvent être des matrices (cf. plus loin) et s'il n'y en a pas, écrire `[ ]`.

Si l'on rentre ces lignes dans un fichier à part, il faut l'enregistrer avec l'extension `sci` et reprendre le nom (ici `facto`) de la fonction. L'enregistrement se fait alors dans `facto.sci'`. Une fois ce programme enregistré, pour calculer 12!, l'utilisateur devra taper en ligne de commande[2] :

```
>  exec("facto.sci") // pour charger la fonction
>  facto(12)
```

Attention : après chaque modification d'une fonction, il faut la charger à nouveau grâce à `exec`.

1.6 Calcul vectoriel et matriciel.

Une spécificité de SCILAB est son traitement des matrices. **Cette section est d'ailleurs certainement la plus importante de ce chapitre.** Il est possible d'appliquer de manière simple et en générale efficace en terme de temps de calcul des opérations élément par élément d'une matrice. Si par exemple on veut calculer une somme :

$$\sum_{k=1}^{100} F(k)$$

avec une fonction F explicite (disons $F(t) = \sin(t)/(1+t^2)$ pour donner un exemple), le point de vue usuel consisterait à suivre l'ordre suivant :
on fait tout d'abord une boucle – pour k variant de 1 à 100, on calcule $F(k)$ – puis on ajoute les valeurs. Même si en SCILAB on peut suivre cette démarche, celle qui est plus dans la philosophie de ce logiciel, et qui est plus efficace en terme de temps et moyen de calcul, consiste plutôt à faire :

[2] Je rappelle que `exec` est accessible par les menus

on crée un vecteur de dimension 100 contenant les entiers de 1 à 100, on applique élément par élément la fonction F, puis on somme les composantes.

C'est un nouveau mode de pensée auquel vous devez vous habituer pour devenir des utilisateurs efficaces de SCILAB. Mais avant de revenir sur ce point, passons en revue les premières commandes utiles sur les matrices.

1.6.1 Saisie.

Saisie d'un vecteur ligne :
> `x=[1 3 4 9]`

Observez l'effet de : sur les exemples suivants :
> `y=[1:10]`
> `z=[1:2:7]`
> `z=[15:-2:3]`

Quel est-il ? Il s'agit donc d'une première commande très utile en vue d'appliquer le principe général exposé dans l'introduction de cette section. Exercez vous si besoin en prenant des petits exemples.

Assez proche, on dispose de `linspace`. Si a et b sont des réels tels que $a < b$ et n un entier naturel non nul, `linspace(a,b,n)` est le vecteur ligne à n composantes séparant l'intervalle $[a, b]$ en $(n-1)$ intervalles égaux, i.e. le $k-$ème terme est $a + \frac{k-1}{n-1}(b-a)$.

Saisie d'un vecteur colonne. Un vecteur colonne est vu comme une matrice donc chaque ligne est composée d'un seul élément, et dans SCILAB on sépare les lignes par des points-virgules :
> `z=[1;3;4;9] ;`

En fin de ligne, le symbole ; permet de ne pas afficher le résultat. x' désigne la transposée dans le cas réel (et la transconjuguée dans le cas complexe).
> `x'-z`

Donc `z=x'`. On peut accéder aux éléments individuels :
> `z(3)`.

Pour un vecteur, on peut obtenir la somme, le produit, la moyenne, etc. de ses coefficients par les commandes `sum, prod, mean`, etc. Nous verrons d'autres commandes plus loin dans cette section, et nous verrons leurs applications aux matrices. Exemple :
> `sum(z), prod(z), mean(z)` ;

Passons aux matrices.

La première matrice à signaler est la matrice vide : `[ ]`.
On entre les matrices ligne à ligne (espaces ou virgules entre éléments d'une même ligne), puis on change de ligne par un point-virgule.
> `A=[16 2 3 13 ; 5 11 10 8 ; 9 7 6 12 ; 4 14 15 1]`

Tester les commandes suivantes, et en déduire le sens des commandes `ones, eye, zeros` :
> `eye(3), eye(3,3), eye(3,5)`
> `ones(3), ones(3,3), ones(3,5)`
> `zeros(3), zeros(3,3), zeros(3,5)`

Ces commandes peuvent aussi prendre un vecteur ou une matrice comme entrée au lieu d'un couple de réels. Cela revient à mettre la dimension de la matrice entrée comme argument[3]. Par exemple, tapez la commande `B=eye(A)` qui va vous créer une matrice `B` de même taille que `A`.

La syntaxe s'applique aussi pour entrer des matrices par blocs. Essayez :
> `a=[1 2;4 5], b=[3;6], c=[7 8], d=[9]`
> `[a b;c d]`

1.6.2 Premières opérations.

Passons aux déterminants et à l'inverse :
> `C=A+B; det(C)`
> `inv(C)`

On dispose des opérateurs `*` et `^` pour calculer le produit de deux matrices et pour élever à une puissance :
> `C*A`
> `A^2`

Observez les résultats suivants et en déduire le sens de / et de \ pour les matrices carrées de bonnes dimensions et inversibles[4].
> `A/C-A*inv(C)`
> `C\A-inv(C)*A`

[3] dans `eye(3), ones(3), zeros(3)`, SCILAB interprête le chiffre 3 comme une matrice de taille $(1,1)$, le comportement de MATLAB est différent ici

[4] Il existe des extensions dans d'autres cas, nous en verrons une dans le cadre des moindres carrés

1.6.3 Extraction de sous matrices.

Pour extraire la première ligne de `A`, on écrit :
> `A(1,:)`
et pour extraire la seconde colonne, on écrit :
> `A(:,2)`
L'élément en troisième ligne et quatrième colonne s'obtient par :
> `A(3,4)`
Pour mettre sous forme d'un vecteur colonne les colonnes de `A` les unes après les autres, on écrit :
> `A(:)`
Pour obtenir la sous matrice formées des lignes 1 et 3 et des colonnes 1 et 2, on écrit :
> `A([1 3],[1 2])`
Enfin, pour supprimer des lignes ou des colonnes d'une matrice, on les remplace par une matrice vide. Par exemple, la commande suivante supprime la première ligne de `C` :
> `C(1,:)=[ ]`

Exercice 1.6.1 *On introduit la matrice $D = A/34$. Vérifier à l'aide de SCILAB que les sommes de chaque ligne et de chaque colonne font 1 (la matrice D peut donc s'interprêter comme une matrice de transition d'une chaîne de Markov). On utilisera la fonction* `sum` *qui s'applique à une matrice (taper* `help sum`*). Calculez les puissances successives de D. Que constatez-vous ?*

1.6.4 Opération élément par élément (ou vectorisation).

On en vient à l'une des spécificités de SCILAB signalée en introduction : il est possible d'effectuer des opérations élément par élément, ce qu'il est conseillé de faire dès que le problème s'y prête. Par exemple, la commande :
> `log(A)`
calcule le logarithme de chaque élément de la matrice `A`. L'effet d'application d'une fonction usuelle ou spéciale est identique. Par exemple, un vecteur `d1` contenant les exponentielles des nombres 0, 0.001, ..., 0.999, 1 se fera par :
> `d1=exp(0:0.001:1);`
(le point-virgule est mis pour éviter l'affichage de `d1`). Créez de même le vecteur `d2` de dimension 1001 dont les composantes sont $\sin(k)$ pour k variant de 0 à 1000.
Pour deux matrices de même dimension $A = (a_{ij})$ et $B = (b_{ij})$, il est possible d'effectuer une multiplication, une division ou une exponentiation élément par élément. Plus précisément, les matrices $(a_{ij}b_{ij})$, (a_{ij}/b_{ij}) et $(a_{ij}^{b_{ij}})$ s'obtiennent respectivement par :
> `A.*B`

```
> A./B
> A.^B
```

Essayez avec :

```
> A=[16 2 3 13 ; 5 11 10 8 ; 9 7 6 12 ; 4 14 15 1]; B=A/10 ;
```

Ainsi, ne confondez pas les opérations `.*` et `*` (ou `./` et `/`). A l'aide de `d1` et `d2`, créez le vecteur :

$$d3 = \left(\exp(k/1000)^{\sin(k)}\right)_{0\leq k\leq 1000}.$$

Ces opérations s'étendent au cas où l'une des matrices est un nombre. Dans ce cas, tout se passe comme si on avait à la place de ce nombre une matrice de même dimension que l'autre matrice et composée uniquement du nombre en question. Par exemple, pour retrancher 1 à chacune des composantes de `d1`, et élever au cube chacune des composantes de `d2`, il suffit d'écrire :

```
> e1=d1-1 ; e2=d2.^3 ;
```

Exercice 1.6.2 *Appliquez ces raccourcis pour créer :*

- *Un vecteur contenant les carrés des entiers de 1 à 10.*
- *Un vecteur contenant les 2^p pour p impair variant de 1 à 11.*

Remarque 1.6.3 *En fait SCILAB accepte, lorsqu'il n'y a pas d'ambuiguité, que l'utilisateur ommette le point pour ces opérations. C'est une habitude que je vous déconseille de prendre, car d'une part ceci n'est pas vrai de tous les logiciels, et d'autre part il est toujours bon de réfléchir précisément à l'opération que l'on souhaitait faire.* **A l'examen, nous exigerons une écriture avec point lorsque cela correspond à l'opération souhaitée, même si une écriture sans point serait tolérée par le logiciel.**

Attention : pour calculer les inverses des éléments d'un vecteur `u`, il faut écrire `(1)./u` et non `1./u`, ce dernier étant interprêté comme `(1.)/u` et (si `u` est en ligne) fournit un vecteur `v` tel que `uv=1`.

Reprenons l'exemple du calcul de la somme :

$$\sum_{k=1}^{100} F(k) \text{ avec } F(t) = \sin(t)/(1+t^2).$$

Le plus efficace, lorsque F est une fonction s'appliquant élément par élément (par exemple sin) est de créer le vecteur `k=[1:100]`, de lui appliquer notre fonction F, puis de sommer à l'aide de `sum` qui pour un vecteur donne la somme des composantes. Ici, opérons autrement. Je donne d'abord une version détaillée pour expliquer puis une version courte (sans affections inutiles) telle que vous

devriez l'écrire.
version détaillée. On génère un vecteur dont les composantes sont les entiers de 1 à 100. On calcule les numérateurs des termes $F(k)$, puis leurs dénominateurs, on fait le quotient élément par élément pour obtenir le vecteur $(F(k))_k$, puis on somme ses composantes :

```
> k=[1:100];
> num=sin(k);
> den=1+k.^2;
> Fk=num./den;
> sum(Fk)
```

version abrégée. Je choisis quand même de créer le vecteur nommé avant `k` car il apparaît deux fois, mais les autres affectations sont inutiles :

```
> k=[1:100];sum(sin(k)./(1+k.^2))
```

Citons un autre exemple. Dans l'esprit de SCILAB, la somme $\sum_{i=1}^{200} i^{-2}$ se calculera par :

```
> sum([1:200].^(-2)) ;
```

Exercice 1.6.4 *En suivant l'esprit des raccourcis SCILAB, écrivez la formule la plus courte possible pour calculer ces expressions (regarder l'aide pour* `prod`*) :*

1. $1 \times 2 \times ... \times 20$.
2. $\prod_{k=1}^{100}(1 + k^{-2})$.

Remarque : dans l'écriture d'une fonction, l'idéal est de profiter au maximum de la vectorisation (revoir l'exemple de la fonction phi) afin de pouvoir appliquer la fonction à un vecteur. Ainsi, si j'écris :

```
> deff("y=f(x)","y=x*sin(x)")
```

je ne pourrai pas calculer `f(1:5)`, alors que c'eût été possible en écrivant :

```
> deff("y=f(x)","y=x.*sin(x)")
```

Applications courantes (vecteurs réels) :
1. Si `x` et `y` sont deux vecteurs (réels) lignes de même taille, on peut calculer la somme $\sum_i x_i y_i$ en posant `x*y'`. Si l'on a affaire à des vecteurs colonnes, il faut transposer le premier `x'*y`. Noter que l'on ne peut pas se tromper dans l'ordre de transposition (s'aider des dimensions).
2. Si `x` et `y` sont deux vecteurs (réels) lignes de taille respectives $(1, p)$ et $(1, q)$ (p pouvant être ou non égal à q), la matrice de taille (p, q), $A = (x_i y_j)$ se calcule en posant `A=x'*y`. Là encore, en s'aidant des dimensions, il n'est pas possible de se tromper pour mettre la transposition.

Exercice 1.6.5 *Mettons ceci en pratique. Partons du vecteur ligne* `x=[1 2 3]`*.*
1. Je veux créer la matrice `A` *de taille* $(3, 4)$ *où toutes les colonnes de* `A` *sont des vecteurs* `x` *mis en colonnes. La commande* `x'*ones(1,4)` *convient (bien*

comprendre pourquoi).
2. Créer de même la matrice `B` *de taille* $(3,4)$ *où toutes les lignes de* `B` *sont des vecteurs* `y=[0.5 1 1.5 2]`.
3. A l'aide de `A` *et* `B`*, créer une matrice* $(3,4)$ *dont le terme d'indices* (i,j) *est* $x_i^{y_j}$.

On peut aussi créer des matrices ou des vecteurs dont les composantes sont aléatoires. Par exemple, `rand(p,q)` (ou `rand(p,q,"uniform")`) crée une matrice `p` $\times$ `q` dont toutes les composantes suivent une loi uniforme sur $[0;1]$. La commande `rand(p,q,"normal")` crée une matrice dont toutes les composantes suivent une loi normale centrée réduite. Nous verrons plus tard comment créer des matrices dont les coefficients suivent d'autres lois.

1.6.5 Fonctions complémentaires.

Voici quelques autres fonctions utiles sur les matrices :
> `[m,n]=size(A)` : `m` donne le nombre de lignes et `n` le nombre de colonnes.
> `length(A)` : donne le produit `m n` (i.e. le nombre d'éléments).
> `mean(A)` : retourne la moyenne des éléments de `A`.
> `st_deviation(A)` : retourne l'écart-type empirique – on divise par $n-1$ et non n– des éléments de `A`.
> `max(A)` : retourne le maximum des éléments de `A`.
> `min(A)` : retourne le minimum des éléments de `A`.
> `sum(A)` : retourne la somme des éléments de `A`.
> `prod(A)` : retourne le produit des éléments de `A`.

Toutes les commandes listées de `mean` à `prod` peuvent s'appliquer ligne à ligne ou colonne à colonne. Par exemple, pour obtenir un vecteur ligne (=row en anglais) dont le $i-$ème terme est la somme des termes de la colonne i, entrez : `sum(A,'r')`. De même, il est possible de créer un vecteur colonne dont le $i-$ème terme est la somme des termes de la ligne i, grâce à : `sum(A,'c')`.

Exercice 1.6.6 *Calculez le plus rapidement possible la moyenne des entiers de 1 à 100, de leurs carrés et de leurs cubes.*

Remarque : Etant donné dans $\mathbb{R}^n$, p vecteurs libres, si l'on note X la matrice dont les colonnes sont ces vecteurs, l'ensemble des Xb est en fait le s.e.v. F engendré par nos p vecteurs. Ainsi, le projeté orthogonal de u sur F est $X\hat{b}$. Utiliser cette remarque pour calculer le projeté orthogonal de $u=(1,2,1)'$ sur le s.e.v. engendré par $((4,5,6)',(7,8,9)')$.

1.6.6 Valeurs propres et exponentielle de Matrice.

Valeurs propres.

La commande `spec` permet le calcul de valeurs propres et de vecteurs propres.

Dans la syntaxe :
> `spec(A)`
la commande `spec` retourne les valeurs propres de la matrice `A`. Pour avoir aussi les vecteurs propres (dans le cas diagonalisable), on tape :
> `[P,D]=spec(A)`
ce qui donne un couple de matrices `P` et `D`, la seconde étant diagonale, telles que `AP=PD`. Lorsque la matrice `P` est inversible, il s'ensuit que la matrice `A` est diagonalisable, et l'on rappelle que dans ce cas, la $i-$ème colonne de `P` donne un vecteur propre associé à la $i-$ème valeur propre située sur la diagonale de `D`.

Exercice 1.6.7 *Prenez la matrice* `A=[16 2 3 13; 5 11 10 8; 9 7 6 12; 4 14 15 1]` *et à l'aide de la commande* `spec`*, indiquez les éléments propres de* `A` *et si* `A` *est diagonalisable.*

Prenons maintenant l'exemple d'une matrice non diagonalisable.
`A=[-14 -25 ; 9 16]; [P,D]=spec(A); rcond(P)`
La faible valeur de `rcond(P)` nous fait penser qu'elle n'est pas inversible. Vérifiez que l'on a bien `AP=PD` (aux erreurs d'arrondis près), mais tentez le calcul de `inv(P)*A*P-D`.

Exercice 1.6.8 (calcul approchée des valeurs propres et vecteurs propres). *Supposons que l'on ait une matrice A réelle ayant ses valeurs propres $\lambda_1, \cdots, \lambda_n$ satisfaisant $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|$.*
1. *Justifier que A est diagonalisable. On note $(e_1, \cdots, e_n)$ une base de vecteurs propres de sorte que pour tout i, e_i soit associé à λ_i.*
2. *Soit $x = \sum_{i=1}^{n} x_i e_i$ un vecteur non nul arbitraire, tel que $x_1 \neq 0$ (ce qui est le cas, à moins de vraiment mal choisir x). On introduit la suite de vecteurs $(y_p)_p$ définie par $y_p = A^p x$. Donner le terme dominant de y_p quand $p \to +\infty$, et en déduire une méthode de calcul de $|\lambda_1|$.*
3. *Déterminer la direction limite de $y_p/\|y_p\|$, et en déduire comment calculer la direction de e_1.*
4. *Soit B une matrice, on pose $A = {}^tBB$. Expliquer pourquoi A est diagonalisable et ses valeurs propres sont des réels positifs. Appliquer la méthode précédente à A après avoir choisi une matrice B aléatoire d'ordre 3 (faire* `B=rand(3,3);A=B'*B`*). La convergence semble t-elle rapide sur votre exemple ? Quelle est-elle en théorie ?*

Exponentielle, logarithme et racine carrée d'une matrice.

L'exponentielle d'une matrice carrée A est définie par la formule :

$$\exp(A) = \sum_{n=0}^{+\infty} \frac{A^n}{n!}$$

et intervient régulièrement en mathématiques, par exemple dans la résolution des équations différentielles linéaires (cf. la section sur la dynamique)). La matrice $\exp(A)$ n'est pas la même que la matrice dont les coefficients sont les exponentielles des coefficients de A, et se calcule par la fonction `expm`.

Attention : il ne faut donc pas confondre l'expression mathématique $\exp(A)$ qui se calcule en SCILAB par `expm(A)` avec l'expression SCILAB `exp(A)`.

Exercice 1.6.9 *Prenez une matrice carrée d'ordre 2 quelconque, et calculez son exponentielle. Comparez-la à la matrice dont les coefficients sont les exponentielles des coefficients de A.*

On dispose de même d'une fonction `sqrtm` et d'une fonction `logm` retournant respectivement une racine carrée et un logarithme de la matrice entrée. La première s'applique à une matrice symétrique, la seconde à une matrice symétrique ou diagonalisable (sans quoi les risques de réponses farfelues sont grands). Ces fonctions dépassent le cadre de notre programme.

1.7 Courbes du plan.

Il vous est conseillé de lire l'aide en ligne pour apprécier toutes les possibilités graphiques de SCILAB. La commande de base utilisée est la commande `plot`. Dans sa syntaxe la plus simple, elle prend deux vecteurs `x` et `y` de même dimension en entrée et retourne un graphique où sont reliés continûment les points (x_i, y_i). On peut s'en servir pour tracer des courbes paramétriques ou des graphes de fonctions (i.e. des équations de la forme $y = f(x)$).

Commençons par des graphes de fonctions. On veut tracer sur un même graphique les graphes des fonctions $x \mapsto x^2$ et $x \mapsto \sqrt{x}$. Le domaine de variation de x est l'intervalle $[0; 2]$. On crée tout d'abord trois vecteurs (remarquez que l'on utilise les raccourcis vectoriels sur lesquels on a déjà insisté) x, y, z :

```
> x=0:0.005:2; y=x.^2; z=x.^(1/2);
```

puis on trace avec cela le graphe de la fonction carrée.

```
> plot(x,y)
```

Une nouvelle fenêtre s'ouvre dans laquelle se trace le graphique. Revenez dans la ligne de commande SCILAB. Par défaut avec `plot`, le second graphique est

superposé. On peut donc superposer la seconde courbe par :
```
> plot(x,z)
```

Pour effacer la fenêtre graphique on dispose de `clf()`.

Enfin, lorsque `x` et `y` sont des matrices (non vecteurs) de même taille, on trace simultanément plusieurs courbes, la $i-$ème étant la $i-$ème colonne de `y` en fonction de la $i-$ème colonne de `x` :
```
> x=linspace(0,2*%pi,50); y=x.*sin(x); n=5; X=ones(n,1)*x; Y=[1:n]'*y;
> clf(); plot(X,Y)
> X=X'; Y=Y';
> clf(); plot(X,Y) // commenter la différence avec ce qui précède
```

A titre d'illustration de courbes paramétriques, on se propose de tracer :

$$\begin{cases} x(t) = \cos(2\pi t) \\ y(t) = \cos(3\pi t) \end{cases}$$

pour t parcourant $[0; 1]$. On commence par effacer le graphique précédent par :
```
> clf();
```
Il faut alors entrer :
```
> t=0:0.01:1 ; x=cos(2*pi*t) ; y=cos(3*pi*t);
> plot(x,y)
```

1.8 Booléens et opérateurs relationnels.

Un Booléen est une variable pouvant prendre deux valeurs : vraie (true en anglais) et fausse (false). Dans Scilab, vous écrivez respectivement `%t` et `%f` et dans les réponses du logiciel vous verrez apparaître `T` et `F`. Pour les calculs avec des nombres, il est convenu que `T=1` et `F=0`.

Les opérateurs relationnels sont :

$$< \quad > \quad <= \quad >= \quad == \quad \sim=$$

et signifient respectivement : strictement inférieur, strictement supérieur, inférieur ou égal, supérieur ou égal, égal et différent ($\sim$ est obtenu par la combinaison des touches `AltGr` et 2). Le résultat est 1 si l'assertion est vraie, 0 si l'assertion est fausse. Dans le cas de deux matrices de même dimension, ils donnent une matrice de même dimension où les relations sont vérifiées élément par élément. Dans le cas d'une matrice et d'un nombre, tout se passe comme si le nombre était une matrice de même taille que l'autre et dont tous les coefficients sont ce nombre.

On dispose des opérateurs logiques :

$$\& \quad | \quad \sim$$

signifiant respectivement `et, ou, non` (| est obtenu par la combinaison des touches `AltGr` et 6). Les syntaxes sont les suivantes : si `A` et `B` sont deux matrices de même format[5], `A & B` (resp. `A | B`, retourne une matrice de même format dont le terme d'indices ij est 1 si et seulement si chacun (resp. au moins l'un) des termes ij des matrices `A` et `B` est non nul (et 0 sinon).

Exemple : essayez :

```
>  x=[0 1 2]; y=[2 1 0]; x&y, x| y
```

Soit à tracer sur $[-2; 2]$ la fonction :

$$f(x) := \begin{cases} x^2 - 1 & \text{si } x \geq 0 \\ 0 & \text{sinon} \end{cases}$$

Une solution peut être fournie par la liste d'instructions :

```
>  x=-2:  0.01:  2;
>  y=(x.^2-1).*(x>=0) ;
>  plot2d(x,y)
```

Expliquons la seconde ligne. Quand x_i est positif, (x_i >= 0) est vraie donc donne la valeur `T` (i.e. 1 pour la multiplication), et sinon donne `F` (i.e. 0 pour la multiplication).

Voyons un autre exemple (avec un piège celui-ci). Soit à tracer la fonction :

$$f(x) = \begin{cases} x^2 + 1 & \text{si } x \geq 0 \\ \cos(x) & \text{si } x \leq 0 \end{cases}$$

Une solution est fournie par la liste d'instructions :

```
>  x=-2:  0.01:  2;
>  y=(x.^2-1).*(x>=0)+((cos(x)).*(x<0)) ;
>  plot2d(x,y)
```

où cette fois il faut noter que j'ai dû mettre `x<0` et non `x<=0` pour éviter de compter deux fois le cas $x = 0$. Essayez de bien comprendre ces exemples avant de traiter l'exercice suivant :

Exercice 1.8.1 *Tracez la fonction définie sur* $[0; 2\pi]$ *par :*

$$f(x) := \begin{cases} \sin(x) & \text{si } |\sin(x)| < 0.5 \\ 0.5 & \text{si } \sin(x) \geq 0.5 \\ -0.5 & \text{si } \sin(x) \leq -0.5 \end{cases} .$$

Vous pourrez superposer avec des symboles la fonction sinus (pour laquelle vous parcourerez l'axe des abscisses par pas de 0.2).

[5] Le cas d'une matrice et d'un nombre vous est maintenant familier.

1.9 Boucles et structures conditionnelles.

On va illustrer ceci avec le calcul de factorielle 10, c'est-à-dire du produit $1 \times \cdots \times 10$ (noté en général 10!). Les solutions proposées ici ne sont pas les plus efficaces avec SCILAB, mais ont pour but d'illustrer le propos.

1.9.1 Boucles for.

La syntaxe d'une boucle `for` est la suivante :

```
for compteur = A,
instructions
end
```

où `A` est une matrice (un vecteur le plus souvent). Le `compteur` prend successivement les valeurs des coefficients de la matrice `A`. Le cas le plus courant est celui où A=debut:pas:fin, avec comme d'habitude, un pas qui peut être omis s'il est de 1. Appelons `z` la variable dans laquelle nous allons stocker la réponse. Il ne faut pas oublier, et c'est une erreur classique en programmation, d'initialiser la variable. Voici donc le script correspondant au calcul de factorielle 10 par une boucle for :

```
z=1 ; for i=2:10 , z=z*i; end; z
```

Le dernier `z` est là pour provoquer l'affichage du résultat.

Remarque : vous pouvez, et cela est conseillé, rentrer les commandes sur plusieurs lignes. SCILAB n'effectuera aucune opération tant que vous n'aurez pas entré le `end`[6]. Cette remarque est valable pour les suivants.

Voici un script où l'on itère 100 fois la fonction logistique $x \mapsto 4x(1-x)$:

```
x=sqrt(2)/2 // initialisation
for i=1:100 // on débute ici la boucle qui sera répétée 100 fois
x=4*x*(1-x);
end // fin de la boucle
x,i // on affiche i pour vérifier
```

[6] D'ailleurs les invites sont rapprochées tant que `end` n'est pas entré

Exercice 1.9.1 *On considère la matrice de taille* $(4N) \times 6$ *suivante :*

$$A_N = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 2 & 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 1 & 0 & 0 \\ 1 & 2 & 0 & 0 & 1 & 0 \\ 1 & 2 & 0 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & N & 1 & 0 & 0 & 0 \\ 1 & N & 0 & 1 & 0 & 0 \\ 1 & N & 0 & 0 & 1 & 0 \\ 1 & N & 0 & 0 & 0 & 1 \end{pmatrix}$$

issue d'un problème économétrique. Construire la matrice en profitant des moyens SCILAB. On pourra proposer une boucle, en remarquant comment passer de A_{k-1} *à* A_k*.*

1.9.2 Instruction while.

La syntaxe d'une instruction `while` est la suivante :
`while conditions,`
`instructions`
`end,`
ce qui signifie que l'on répète les instructions tant que les conditions sont vraies.
Notre problème s'écrit avec l'instruction while :

```
i=1 ; z=1 ; while i<=10, z=z*i; i=i+1; end; z
```

Exercice 1.9.2 *Exécutez à nouveau cette ligne en enlevant l'affectation* `i=1`*, et expliquez ce qui s'est passé.*

Revenons à notre fonction logistique. Si $x_0 = \sqrt{2}/2$, on veut connaître le premier indice k pour lequel $x_k \geq 0.999$:

```
x=sqrt(2)/2 // initialisation
i=1 // compteur
while(x<0.999) // tant que x<0.999 faire ce qu'il y a jusqu'au end
x=4*x*(1-x); // ici on calcule x_i
i=1 // le compteur augmente, il devient n+1 au dernier coup
end // fin de la boucle
i-1 // on affiche n
```

1.9.3 Condition if.

La syntaxe est :

```
if conditions,
then instructions (si les conditions sont satisfaites)
else
instructions (si les conditions ne sont pas satisfaites)
end
```

La partie `else` est facultative. Le mot-clef `then` peut être remplacé par une virgule ou un passage à la ligne (que l'on a fait ici pour plus de lisibilité). De plus, si l'on veut remettre des conditions dans le cas où la première série de conditions n'est pas satisfaite, on dispose d'une instruction `elseif` (en un seul mot) dont la syntaxe est :

```
if conditions1
then instructions1 (si les conditions1 sont satisfaites)
elseif conditions2
then instructions2 (si les conditions2 sont satisfaites)

                                ...

elseif conditionsp
then instructionsp (si les conditionsp sont satisfaites)
else
instructions (dans les autres cas)
end,
```

Cette fois, on part d'un entier N. Lorsqu'il est pair, on le divise par 2 et s'il est impair, on calcule $3N + 1$, et on arrête le processus lorsqu'on arrive à 1 :

```
N=247 // par exemple
i=1 // compteur du nombre d'étapes pour arriver à 1
while(N>1)
if (modulo(N,2)==0)
test
then N=N/2;
else N=3*N+1
i=i+1
end
end, i-1
```

1.9.4 Tests d'arrêts.

Définir un critère d'arrêt lors de l'utilisation d'un algorithme dont la solution s'obtient théoriquement comme limite et dont on ignore la vitesse de convergence est assez utile. On donne un nombre d'itérations maximal (pour que l'algorithme s'arrête) not ici `itermax` et une précision limite : si l'algorithme change la valeur

de moins de ε lors d'une étape, on l'arrête. Soit par exemple à calculer :

$$S = \sum_{k=1}^{+\infty} \frac{1}{k^3}$$

(on connaît la vitesse de convergence de cette srie, mais on va oublier provisoirement cet aspect des choses). On procde alors ainsi :

```
itermax=10000;
eps=10^ (-5);
S=0;
n=0;
t=1;
while (n<itermax) & (t>eps)
n=n+1;
t=n^ (-3);
S=S+t;
end;
if n==itermax
disp('attention itermax atteint')
end
n,S
sum([1:10000].^ (-3))
```

La comparaison des deux derniers résultats montre la limite de cette approche : la somme totale ne sera pas connue 10^{-6} près ; pire, toutes les séries $\sum_n u_n$ avec $u_n \to 0$ semblent converger, ce qui est évidemment faux.

1.9.5 Une application aux mathématiques financières.

Voici une petite application simple de la boucle `for` et des simplifications matricielles apportées par SCILAB. Le problème est de construire un tableau d'amortissement d'un emprunt.

Le principe est le suivant. Supposons que l'on emprunte sur n mois une somme S au taux mensuel i, et que l'emprunt donne lieu à des mensualités $M_1, ..., M_n$ en fin de chaque mois. Ces mensualités doivent satisfaire :

$$S = \sum_{t=1}^{n} \frac{M_t}{(1+i)^t}.$$

Par exemple, dans le cas de mensualités constantes, un calcul simple montre qu'elles valent :

$$M = \frac{Si}{1 - \frac{1}{(1+i)^n}}.$$

Le principe est qu'à chaque fin de mois, la mensualité paie tout d'abord les intérêts sur le capital restant dû, puis le reste de la mensualité constitue l'amortissement du capital, c'est-à-dire est diminué du capital restant dû.

Ecrivez une fonction prenant en entrée n, i et S et dressant un tableau d'amortissement où les colonnes correspondent respectivement au capital restant dû, à la mensualité, à la partie de celle-ci payant les intérêts puis à l'amortissement. Pour vous aider, voici comment on remplit la première ligne, après avoir calculé la mensualité M : on met en première colonne $S_1 = S$, dans la seconde $M_1 = M$ (la seconde contient toujours M dans un cas de mensualité constante), dans la troisième $I_1 = S_1 i$, dans la quatrième $A_1 = M - I_1$. On continue ainsi sauf que bien entendu $S_2 = S_1 - A_1$. Application : rédigez le tableau d'amortissement pour un emprunt de 100000 sur 48 mois au taux mensuel de 0.00487 (représentant un taux annuel de 6%).

Voici maintenant, dans le cas de mensualités constantes, une solution plus rapide, dans laquelle vous vous efforcerez d'utiliser les spécificités de SCILAB :
1. Vérifiez la relation de récurrence $S_{t+1} = (1+i)S_t - M$. Il s'agit d'une suite arithmético-géométrique dont on peut démontrer que l'expression est :

$$S_t = (1+i)^{t-1}(S - M/i) + M/i.$$

Créez un vecteur colonne dont les composantes sont les S_t.
2. Créez le vecteur colonne des mensualités et utilisez ces deux vecteurs pour créer la fin du tableau.

1.10 Polynômes et fractions rationnelles.

Cette section sert de référence, elle n'est pas au programme.

On peut faire quelques opérations sur les polynômes à l'aide de SCILAB. Par défaut, la variable s'écrit `%s`. Je signale juste quelques commandes, voir `apropos poly` pour plus d'informations.

Tout d'abord, si `v` est un vecteur, `poly(v,'x')` (ou `poly(v,'x','r')`) crée le polynôme de variable `x` dont les racines sont les éléments de `v`. Le polynôme de variable `x` dont les coefficients (par ordre croissant) sont les éléments de `v` s'obtient par `poly(v,'x','c')`. Si `P` est un polynôme, on peut réciproquement obtenir ses coefficients (resp. ses racines, resp. ses facteurs irréductibles sur $\mathbb{R}$) par `coeff(P)` (resp. `roots(P)`, resp. `factors(P)`).

Exemples :

```
>  v=[1,2,3]
>  P1=poly(v,'z'), P2=poly(v,'z','c')
```

```
> roots(P1), coeff(P1)
> roots(P2), coeff(P2)
> factors(%s^4-1)
```
On commentera le dernier résultat.

1.11 Gestion du temps.

Pour évaluer le temps d'exécution d'un algorithme, on peut utiliser la commande `timer()`. On encadre les commandes dont on veut évaluer le temps d'exécution entre deux `timer()`, le premier déclenche un chronomètre et le second l'arrête. Exemple :
```
> timer(); sum(log10(1:10000));timer()
```
Vous pouvez comparer avec :
```
> timer(); s=0; for i=1:10000; s=s+log10(i); end;timer()
```
pour apprécier l'efficacité des raccourcis SCILAB.

Chapitre 2

Gestion des erreurs.

2.1 Erreurs.

2.1.1 Représentation des nombres réels.

Prenons x un réel non nul. Il s'écrit de manière unique sous la forme :

$$x = \varepsilon a 10^n, \qquad (\varepsilon, a, n) \in \{-1; 1\} \times [1; 10[\times \mathbb{Z}.$$

L'écriture de droite s'appelle la notation scientifique. ε, a, n s'appellent respectivement le signe, la mantisse et l'exposant de x. Connaissant x, on calcule $\varepsilon = x/|x|$, $n = E(\log_{10}(|x|))$ et enfin $a = |x|10^{-n}$.

Souvent les nombres réels sont représentés par une approximation x'. L'usage est alors d'arrondir le dernier chiffre au plus proche. Une représentation à p chiffres après la virgule de x c'est se donner un nombre x' arrondi à la dernière décimale de sorte que $|x - x'| \leq 0,5 \times 10^{-p}$. Par exemple, avec $p = 2$ on obtient les approximations suivantes :

$$12,4324 \longmapsto 12,43,$$

$$12,4373 \longmapsto 12,44,$$

$$7,99921 \longmapsto 8.$$

Le cas de $7,235$ est ambigu : à la fois $7,23$ et $7,24$ sont acceptables. Parfois, les logiciels ont des régles différentes selon la parité de l'avant dernier chiffre, ce qui fait qu'en moyenne les arrondis se compensent.

Une représentation à p chiffres significatifs consiste à ne retenir une approximation ne contenant que p chiffres, le dernier étant arrondi. À titre d'exemples, avec $p = 3$:

$$7,243 \longmapsto 7,24,$$

$$7,243 \longmapsto 7,24,$$

$$12,437 \longmapsto 12,4,$$
$$1248 \longmapsto 1250.$$

Lorsque $x \in]1,10[$ sa représentation à $p-1$ chiffres après la virgule est identique à sa représentation à p chiffres significatifs. En revanche, la représentation à p chiffres significatifs est toujours relative et la plus parlante : une représentation à $p-1$ chiffres après la virgule serait très précise pour un nombre très grand, au contraire sans aucun sens pour un nombre très petit. Par exemple, à 3 chiffres après la virgule, $0,000045$ se représente par 0 !

2.1.2 Erreurs absolues et relatives.

Considérons un nombre x représenté de manière approchée par un nombre x'. L'erreur absolue sur x est alors par définition :

$$\epsilon_x^a = x' - x.$$

Il est possible de faire d'autres conventions, telle $|x' - x|$. En général, celle-ci n'est pas connue exactement, on en a un encadrement. Si par exemple x' est l'approximation de x à p chiffres après la virgule (resp. à p CS), nous avons $|\epsilon_x^a| \leq 0,5 \times 10^{-p}$ (resp. $|\epsilon_x^a| \leq 0,5 \times 10^{n-p}$).

La plus parlante est plutôt l'erreur relative, que l'on peut exprimer d'ailleurs en pourcentage. Elle ramène l'erreur à la grandeur de x :

$$\epsilon_x^r = \frac{\epsilon_x^a}{|x|}.$$

Comme x n'est pas connu, on en prend souvent l'approximation $\frac{\epsilon_x^a}{|x'|}$.

On notera que la dernière approximation est justifiée si ϵ_x^r est petit ; vous ne devrez pas être surpris des calculs sur les erreurs, qui sont surtout utiles pour avoir des ordres de grandeurs. Parfois on fait des approximations, que l'on majore après. Ce n'est évidemment pas rigoureux (il est vrai que $0,9998 < 0,9999$ mais si l'on approche le premier par 1, on obtient l'inégalité fausse $1 < 0,9999$). Cependant, les majorations sont souvent suffisamment grossières pour que les encadrements à l'arrivée ne soient pas (trop) faux. Profitez-en, c'est un rare moment où en mathématiques on vous autorisera des calculs heuristiques !

2.1.3 Règles de calculs sur les erreurs.

Commençons par ce qui est additif. Donnons nous des nombres $x_1, \ldots, x_n$ connus approximativement, et $a_1, \ldots, a_n$ connus exactement. Alors si l'on pose $y = \sum_{i=1}^n a_i x_i$, nous avons :

$$\epsilon_y^a = \sum_{i=1}^n a_i \epsilon_{x_i}^a,$$

(et donc $|\epsilon_y^a| \leq \sum_{i=1}^n |a_i||\epsilon_{x_i}^a|$). Par exemple, si les x_i sont tous connus à p chiffres après la virgule :

$$|\epsilon_y^a| \leq \frac{\sum_{i=1}^n |a_i|}{2} 10^{-p}.$$

Le calcul sur les erreurs relatives est cependant moins aisé. Par exemple, si x_1+x_2 est proche de 0 et pas les x_i, l'erreur relative va exploser. On commence donc toujours ces calculs par l'erreur absolue, puis on passe au relatif après.

Regardons maintenant le cas d'un produit $z = xy$ de deux nombres *strictement positifs* connus approximativement par des approximations x' et y'. Comme approximation de z, on prend $z' = x'y'$, ce qui donne une erreur absolue :

$$\epsilon_z^a = z' - z = x\epsilon_y^a + y\epsilon_x^a + \epsilon_x^a\epsilon_y^a,$$

et donc :

$$\epsilon_z^r = \epsilon_y^r + \epsilon_x^r + \epsilon_x^r\epsilon_y^r.$$

On voit que la formule est bien plus agréable sur les erreurs relatives, et d'ailleurs dès que l'une est très inférieure à 1, on peut négliger le produit ce qui donne comme approximation :

$$\epsilon_z^r \approx \epsilon_x^r + \epsilon_y^r.$$

En pratique, on utilise celle-ci (tout en sachant les limites), et même on n'hésite pas à écrire :

$$|\epsilon_z^r| \leq |\epsilon_x^r| + |\epsilon_y^r|.$$

Plus généralement, considérons des nombres $x_1, \ldots, x_n$ strictement positifs connus approximativement, et $a_1, \ldots, a_n$ connus exactement. Alors si l'on pose $y = \prod_{i=1}^n x_i^{a_i}$, en supposant les erreurs relatives sur les x_i petites :

$$\epsilon_y^r \approx \sum_{i=1}^n a_i \epsilon_{x_i}^r.$$

Les formules précédentes sont des cas particuliers des formules de transformation des erreurs par une fonction. Nous allons le faire dans le cas d'un seul nombre connu approximativement, et illustrerons un cas à plusieurs nombres inconnus. Soit x un nombre connu approximativement par une approximation x', et f une fonction suffisamment régulière. On pose $y = f(x)$ qui s'approche par $y' = f(x')$. L'erreur absolue est alors :

$$\epsilon_y^a = f(x') - f(x).$$

Lorsque f est $k-$lipschitzienne sur $[x, x']$, on peut écrire :

$$|\epsilon_y^a| \leq k|\epsilon_x^a|.$$

Lorsque f est dérivable en x, si $f'(x) \neq 0$, supposant l'erreur absolue petite :

$$\epsilon_y^a \approx f'(x)\epsilon_x^a \approx f'(x')\epsilon_x^a.$$

Ces approximations peuvent être rendues rigoureuses. En supposant $|f''|$ majorée par une constante M_2 sur $[x, x']$, on obtient :

$$|\epsilon_y^a - f'(x)\epsilon_x^a| \leq \frac{M_2}{2}(\epsilon_x^a)^2,$$

à l'aide des formules de Taylor qu'il convient de réviser.

Donnons nous maintenant des nombres $x_1, \ldots, x_n$ connus approximativement. On pose $y = f(x_1, \ldots, x_n)$. La définition de la différentielle montre que :

$$\epsilon_y^a \approx \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}(x)\epsilon_{x_i}^a \approx \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}(x')\epsilon_{x_i}^a.$$

L'erreur obtenue par ses approximations se calculerait par une formule de Taylor (voir cours d'analyse S5).

2.1.4 Exercices.

1. On note $x_1 = 1000$, $x_2 = \ldots = x_{12} = 1$. Supposons que l'on travaille à trois chiffres significatifs. On veut calculer $s = \sum_{i=1}^{12} x_i$.

   (a) On calcule la suite $s_1 = x_1$, $s_j = x_j + s_{j-1}$, de sorte que $s = s_{12}$. Que donne un logiciel travaillant à trois chiffres significatifs ? Est-ce correct (à trois chiffres) ?

   (b) Mêmes questions en considérant la suite $s'_1 = x_{12}$, $s'_j = s'_{j-1} + x_{13-j}$ (de sorte que $s = s'_{12}$).

2. On donne une valeur approchée de $\pi \approx 3,14159$.

   (a) Donner une valeur approchée de π à 10^{-2} près. Et à quatre chiffres significatifs.

   (b) On approche $\sqrt{\pi}$ par $\sqrt{3,14159}$. Donner une estimation de l'erreur commise. Vérifier dans SCILAB.

   (c) On cherche à évaluer $\sqrt[3]{\pi}$ à trois chiffres significatifs. Quelle précision doit-on choisir pour π ?

2.2 Propagation des erreurs.

Voici une première illustration[1] du phénomène de propagation des erreurs. Le but est de calculer une valeur approchée de :

$$\int_0^1 \frac{t^{19}}{-10+t} dt.$$

Considérons pour ce faire la suite d'intégrales, pour $n \geq 1$:

$$I_n = \int_0^1 \frac{t^{n-1}}{-10+t} dt.$$

On remarque facilement les faits suivants :

- $\forall n, \quad -1/(9n) \leq I_n \leq 0.$
- $\forall n, \quad I_{n+1} = 10I_n + 1/n.$

Comme $I_{n+1} \leq 0$, la relation de récurrence combinée avec la première inégalité montre qu'en fait :

$$\forall n, \quad -\frac{1}{9n} \leq I_n \leq -\frac{1}{10n}$$

soit :

$$\forall n, \quad 9 \leq -90nI_n \leq 10.$$

De plus, on sait que $I_1 = \ln(9/10)$. Le script SCILAB suivant montre que l'encadrement est rapidement mis en défaut :

```
N=1; I=log(9/10); t=-90*I; T=[N,I,t];
while (t>=9)&(t<=10) do
I=10*I+1/N;
N=N+1;
t=-90*N*I;
T=[T;N,I,t];
end
T
```

La raison en est que nI_n est de l'ordre de $-1/n$. Par conséquent, alors qu'une suite issue d'une récurrence de la forme $J_{n+1} = 10J_n$ ne verrait pas l'erreur relative augmenter, au contraire ici en raison des compensations elle explose. Vous apprécierez d'ailleurs la qualité de la récurrence inverse, qui consisterait à partir du résultat faux $I_{40} = 0$. Mais les erreurs sur la première partie sont divisées par 10 à chaque étape, et sur $1/n$ elles sont dans la précision :

```
N=1; I=0;
```

[1] empruntée à l'excellent livre de J.P. Demailly, Analyse Numérique et Equations Différentielles.

```
for n=39:-1:1;
I=(I-1/n)/10;
end
abs(I-log(9/10))/log(9/10)
```

Une autre piste pour calculer I_n serait de poser $u = t - 10$ et d'appliquer la formule du binôme. On obtient ainsi, pour tout $n \geq 2$:

$$I_n = 10^{n-1}\left[\ln(9/10) - \sum_{k=1}^{n-1} \frac{(-1)^k}{k} \begin{pmatrix} n-1 \\ k \end{pmatrix} \left(1 - (9/10)^k\right)\right]$$

Le résultat est absurde (cf script joint) ; pourquoi ?

```
S=log(9/10);
N=20;
for k=1:N-1
S=[S,(-1)^ k*(gamma(N)/gamma(k+1)/gamma(N-k))*(1-(9/10)^ k)/k];
end;
I=10^ (N-1)*sum(S)
S'
```

2.3 Erreurs dans les systèmes linéaires.

De très nombreux problèmes scientifiques, une fois modélisés et éventuellement linéarisés, se ramènent à un problème de résolution d'un système linéaire ou à un problème de valeurs propres. Le but de cette section est de vous introduire au problème des erreurs dans le premier problème.

2.3.1 Normes vectorielles et matricielles.

Cette section est introductive à la suivante. Heuristiquement, les normes sont des indicateurs de l'ordre de grandeur d'un vecteur ou d'une matrice. Chercher à minimiser un vecteur revient à minimiser sa norme. Rappelons que l'on dispose de plusieurs normes, équivalentes entre elles (car on est en dimension finie).

Commençons par les normes vectorielles. Si x est un vecteur de $\mathbb{R}^n$ ou de $\mathbb{C}^n$, on a pour habitude de considérer plusieurs normes usuelles :

$$\|x\|_\infty = \max_i |x_i|$$

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p} \qquad (p \geq 1)$$

et la seconde étant considérée en général pour $p = 1$ ou $p = 2$. SCILAB retourne ces normes pour un vecteur `x` à l'aide respectivement des commandes :

```
> norm(x,%inf) // ou bien norm(x,'inf')
> norm(x,p)
```

Dans le second cas, lorsque `p=2`, on peut omettre le `p`. Ainsi, `norm(x)` est identique à `norm(x,2)`.

On peut aussi mettre sur l'espace des matrices plusieurs normes. Les matrices représentent des applications linéaires. Si les espaces de départ sont munis de normes (que l'on notera toutes $\|.\|$ pour éviter de surcharger), il peut être astucieux de mettre la norme subordonnée à ces normes définie par, pour toute matrice A :

$$\|A\| := \sup_{x\in\mathbb{C}^n,\ x\neq 0} \frac{\|Ax\|}{\|x\|}.$$

On notera que même dans le cas de matrices réelles, on prend le supremum sur les vecteurs non nuls de $\mathbb{C}^n$. Cette norme subordonnée a de bonnes propriétés :

- $\|I\| = 1$, où I est la matrice identité.
- $\|AB\| \leq \|A\| \ \ \|B\|$.

Lorsque les espaces de départ et d'arrivée sont munis de leurs normes `p` (`p` fini ou infini), SCILAB calcule la norme subordonnée par : `norm(A,p)`, avec toujours la possibilité de sous-entendre le `p` lorsqu'il vaut 2. Signalons d'ailleurs que pour $p \in \{1, 2, \infty\}$, cette norme a une expression explicite :

$$\|A\|_1 = \max_j \sum_i |a_{ij}|$$

$$\|A\|_\infty = \max_i \sum_j |a_{ij}|$$

$$\|A\|_2 = \sqrt{\text{plus grande valeur propre de } ({}^t\overline{A}A)}.$$

Exemple 2.3.1 *Prenons une matrice A carrée d'ordre 3. Le script suivant donne une minoration de $\|A\|_2$:*

```
s=[];
for i=1:10
x=rand(3,1);
s=[s,norm(A*x)/norm(x)];
end
max(s)
```

2.3.2 Effets des incertitudes.

Si les matrices A et b ne sont pas connues exactement, au lieu de résoudre le système :

$$Ax = b$$

on résout en réalité un système :

$$(A + \Delta A)y = (b + \Delta b)$$

où ΔA et Δb représentent les incertitudes, supposées de norme petites vis-à-vis de celles de A et b. Notons $x + \Delta x$ la solution du second système. Il se peut que même si ΔA et Δb sont de normes très petites (vis-à-vis de celles de A et b), que Δx soit très grand. Un tel système est dit mal conditionné.

Contrairement à ce que l'on pense en première intuition, un bon critère de conditionnement n'est pas le déterminant de la matrice. On trouve dans la littérature des contre-exemples à ce sujet. Je pense que le plus simple pour comprendre est encore de choisir un système linéaire de ce type, avec λ très grand :

$$\left\{ \begin{array}{l} \lambda x_1 = y_1 \\ \frac{1}{\lambda} x_2 = y_2 \end{array} \right. .$$

La matrice de ce système linéaire, de déterminant 1, a une norme très grande devant la matrice :

$$\Delta A = \begin{pmatrix} 0 & 0 \\ 0 & -\frac{1}{2\lambda} \end{pmatrix}$$

qui pourtant modifie considérablement la valeur de x_2 (en la multipliant par 2).

Citons un exemple célèbre dû à R. S. Wilson. Considérons les trois systèmes linéaires "proches" :

$$AX = b, \quad AX = b + \Delta b, \quad (A + \Delta A)X = b$$

avec :

$$A = \begin{pmatrix} 10 & 7 & 8 & 7 \\ 7 & 5 & 6 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 10 \end{pmatrix}, \quad \Delta A = \begin{pmatrix} 0 & 0 & 0.1 & 0.2 \\ 0.08 & 0.04 & 0 & 0 \\ 0 & -0.02 & -0.11 & 0 \\ -0.01 & -0.01 & 0 & -0.02 \end{pmatrix}, \quad b = \begin{pmatrix} 32 \\ 23 \\ 33 \\ 31 \end{pmatrix},$$

$$\Delta b = \begin{pmatrix} 0.1 \\ -0.1 \\ 0.1 \\ -0.1 \end{pmatrix}$$

dont les solutions respectives sont :

$$\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}, \quad \begin{pmatrix} 9.2 \\ -12.6 \\ 4.5 \\ -1.1 \end{pmatrix}, \quad \begin{pmatrix} -81 \\ 137 \\ -34 \\ 22 \end{pmatrix}.$$

On constate qu'elles sont très éloignées, alors que A est à coefficients entiers, de déterminant 1 (son inverse est donc aussi à coefficients entiers) ! Analysons plus précisément ceci.

Etant fixée une norme sur l'ensemble des vecteurs, qui donne une norme matricielle subordonnée, on peut démontrer que la bonne notion est *le conditionnement de la matrice* (relatif à la norme choisie), qui est défini pour toute matrice inversible A par :

$$\mathrm{Cond}(A) = \|A\| \, \|A^{-1}\|.$$

On peut démontrer que ce nombre est toujours supérieur ou égal à 1. Un système mal conditionné en est un pour lequel ce nombre est très grand.

Que le conditionnement réponde bien à la question posée se déduit de la formule suivante (voir exercice final) :

$$\text{Si } \|\Delta A\| < \frac{1}{\|A^{-1}\|}, \text{ alors } \frac{\|\Delta x\|}{\|x\|} \leq \frac{\mathrm{Cond}(A)}{1 - \|A^{-1}\|.\|\Delta A\|} \left[\frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta b\|}{\|b\|} \right].$$

Enfin, en SCILAB, voici les commandes permettant de calculer le conditionnement d'une matrice : `cond(A)` calcule le conditionnement de la matrice `A` relativement à la norme 2 (pour les autres, le calculer à partir de la définition et de `norm`). `rcond(A)` donne une estimation de l'inverse de Cond_1(A), un système mal conditionné donne donc un nombre proche de 0. Il arrive que si `A` est une matrice non inversible et que l'on demande `inv(A)`, on obtienne un message d'erreur faisant apparaître ce nombre en disant que le système est très mal conditionné. **Si un tel message apparaît, il y a toutes les chances pour que votre matrice soit non inversible et que l'inverse proposé soit farfelu** (il arrive que SCILAB propose quelque chose en raison des erreurs d'arrondis). Venons-en à la démonstration de la formule :

Exercice 2.3.2 *On veut établir la formule d'erreur. Pour cela, on considère une matrice carrée inversible d'ordre n, A, un vecteur non nul $n, 1$, b et l'on note x la solution du système linéaire :*

$$Ax = b.$$

1. *On admet que si S est une matrice carrée d'ordre n satisfaisant $\|S\| < 1$, alors $(I + S)^{-1}$ est inversible, et que de plus (ces majorations seront utiles dans la question suivante) :*

$$\|(I + S)^{-1}\| \leq \frac{1}{1 - \|S\|},$$

$$\|(I + S)^{-1} - I\| \leq \frac{\|S\|}{1 - \|S\|}.$$

Soit maintenant une matrice ΔA *telle que* $\|\Delta A\| < \frac{1}{\|A^{-1}\|}$. *Montrer que* $\|A^{-1}\Delta A\| < 1$, *et que* $A + \Delta A$ *est inversible, et enfin que :*

$$(A + \Delta A)^{-1} = (I + A^{-1}\Delta A)^{-1}A^{-1}.$$

2. *On suppose toujours que* $\|\Delta A\| < \frac{1}{\|A^{-1}\|}$, *on se donne* Δb *un vecteur de taille* $n, 1$, *et l'on note* $x + \Delta x$ *l'unique solution du système :*

$$(A + \Delta A)(x + \Delta x) = b + \Delta b.$$

Le but est d'estimer $\frac{\|\Delta x\|}{\|x\|}$ *en fonction de* $\frac{\|\Delta A\|}{\|A\|}$ *et de* $\frac{\|\Delta b\|}{\|b\|}$.
2.a. *Montrer que :*

$$\Delta x = \left[(I + A^{-1}\Delta A)^{-1} - I\right] A^{-1}b + (I + A^{-1}\Delta A)^{-1}A^{-1}\Delta b,$$

puis en déduire que :

$$\|\Delta x\| \leq \frac{\|A^{-1}\|\|\Delta A\|\|x\| + \|A^{-1}\|\|\Delta b\|}{1 - \|A^{-1}\|\|\Delta A\|}.$$

2.b. *On introduit* $\mathrm{cond}(A) = \|A\|\|A^{-1}\|$. *Déduire de ce qui précède la formule :*

$$\frac{\|\Delta x\|}{\|x\|} \leq \frac{\mathrm{cond}(A)}{1 - \|A^{-1}\|\|\Delta A\|}\left[\frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta b\|}{\|b\|}\right].$$

2.c. *Commenter cette formule, notamment lorsque* $\|\Delta A\| << \frac{1}{\|A^{-1}\|}$.

Chapitre 3

Calcul Scientifique.

3.1 Interpolation.

Etant donnés des réels $x_0, \ldots, x_p$ distincts, on cherche à déterminer le polynôme de degré minimal prenant des valeurs prescrites en ces points, avec éventuellement des valeurs fixées pour les dérivées. Si en tout on a N conditions, on cherche parmi les polynômes de degré au plus $N-1$, ce qui fait N coefficients à déterminer.

La situation fréquente est celle où l'on dispose d'une fonction f que l'on cherche à approcher par un polynôme. La valeur de la fonction en ces points et les valeurs des dérivées fournit les conditions que l'on cherche à réaliser.

Précisons un peu. On se donne des entiers $\alpha_0, \ldots, \alpha_p$, et l'on cherche les polynômes satisfaisant :

$$(\textbf{Interp}) \qquad \forall i = 0, \ldots, p, \quad \forall j = 0, \ldots, \alpha_i, \quad P^{(j)}(x_i) = f^{(j)}(x_i).$$

On notera que l'on a en tout N conditions, où $N = \sum_{i=0}^{p} (\alpha_i + 1)$.

Théorème 3.1.1 *Il existe un polynôme P_0 et un seul de degré au plus $N-1 = p + \sum_i \alpha_i$ satisfaisant les conditions* **(Interp)**. *L'ensemble des solutions est alors :*

$$\left\{ P_0 + Q \prod_{i=0}^{p} (X - x_i)^{\alpha_i + 1}; \quad Q \in \mathbb{R}[X] \right\}.$$

De plus, si f est de classe C^N sur $[a;b]$, pour tout $x \in [a,b]$, il existe $\xi_x \in]a,b[$ tel que :

$$f(x) - P_0(x) = \frac{f^{(N)}(\xi_x)}{N!} \prod_{i=0}^{p} (x - x_i)^{\alpha_i + 1}.$$

En voici une illustration informatique :

Exemple 3.1.2 *A l'aide de SCILAB, trouver les polynômes de degré minimaux satisfaisant:*

- $P(1) = P(3) = 1$, $P(2) = 2$ *et* $P(4) = 4$.
- $P(1) = 2$, $P'(1) = 3$ *et* $P(2) = -1$.

Pour le premier, on cherche un polynôme de degré au plus $4 - 1 = 3$. *On pose donc* $P(x) = a + bx + cx^2 + dx^3$. *L'écriture successive de* $P(1) = 1$, $P(2) = 2$, $P(3) = 1$ *et* $P(4) = 4$ *donne le système :*

$$\left\{\begin{array}{rcl} a+b+c+d & = & 1 \\ a+2b+4c+8d & = & 2 \\ a+3b+9c+27d & = & 1 \\ a+4b+16c+64d & = & 4 \end{array}\right. ,$$

d'où la réponse SCILAB (on aurait pu l'optimiser, vu que l'on a une Van der Monde à gauche) :

```
[1 1 1 1;1 2 4 8;1 3 9 27;1 4 16 64]\ [1;2;1;4]
```

Pour le second, on cherche un polynôme de degré au plus $3-1 = 2$. *On pose donc* $P(x) = a+bx+cx^2$. *L'écriture successive de* $P(1) = 2$, $P'(1) = 3$, $P(2) = -1$ *et* $P(4) = 4$ *donne le système :*

$$\left\{\begin{array}{rcl} a+b+c & = & 2 \\ b+2c & = & 3 \\ a+2b+4c & = & -1 \end{array}\right. ,$$

d'où la réponse SCILAB :

```
[1 1 1 ;0 1 2;1 2 4]\ [2;3;-1]
```

Nous allons spécifier et démontrer le théorème dans deux cas particuliers. On notera qu'en pratique, le ξ_x n'est pas connu, et donc on préfère une majoration du type :

$$|f(x) - P(x)| \leq \frac{M_N}{N!} \left| \prod_{i=0}^{p} (x - x_i)^{\alpha_i} \right|,$$

ou encore :

$$|f(x) - P(x)| \leq \frac{M_N}{N!} (b-a)^{N+1},$$

avec $M_N = \sup_{x \in]a,b[} |f^{(N)}(x)|$.

3.1.1 Conditions en un seul point.

On commence par la situation simple où $p = 0$. On se donne un entier $k \geq 0$ et l'on cherche un polynôme P de degré au plus k de sorte que pour tout $j \leq k$, $P^{(j)}(x_0) = f^{(j)}(x_0)$. La réponse est connue depuis la L1, on obtient le polynôme de Taylor :

$$P = \sum_{j=0}^{k} \frac{f^{(j)}(x_0)}{j!} (X - x_0)^j.$$

L'erreur est aussi connue et donnée par la formule de Taylor-Lagrange (ou Taylor avec reste intégral).

3.1.2 Conditions en plusieurs points avec $\alpha_i = 0$ pour tout i.

On se donne maintenant $p+1$ couples (x_i, y_i) pour i variant de 0 à p, où les x_i sont deux à deux distincts, et l'on cherche le polynôme P_0 de degré au plus p tel que pour tout i, on ait : $P_0(x_i) = y_i$. On cherche les coefficients $a_p, ..., a_0$ de P :

$$P_0 = \sum_{j=0}^{p} a_j X^j.$$

Par construction, les coefficients a_j doivent vérifier ces n équations :

$$\begin{cases} a_0 + a_1 x_0 + \ldots + a_{p-1} x_0^{p-1} + a_p x_0^p = y_0 \\ a_0 + a_1 x_1 + \ldots + a_{p-1} x_1^{p-1} + a_p x_1^p = y_1 \\ \vdots \\ a_0 + a_1 x_p + \ldots + a_{p-1} x_p^{p-1} + a_p x_p^p = y_p \end{cases}$$

Ce système est en fait un système de Van der Monde, il a donc une unique solution. Il y a une formule la donnant explicitement, qui n'est pas la plus pratique dès que p est grand. Pour cela, Lagrange a introduit des polynômes particuliers associés aux x_i :

$$L_i = \prod_{j \neq i} \frac{X - x_j}{x_i - x_j}.$$

On constate que $L_i(x_k) = \delta_{ik}$ et donc que :

$$P_0 = \sum_{i=0}^{p} y_i L_i.$$

Formule d'erreur. Le but ici est d'estimer l'erreur produite en remplaçant une fonction f par le polynôme l'interpolant aux x_i (i.e. tel que $y_i = f(x_i)$). On supposera que f est $p+1$ fois dérivable sur un intervalle I contenant les x_i. Soit un réel x fixé dans I distinct des x_i. On introduit la fonction :

$$R(t) = f(t) - P_0(t) - A \prod_{i=0}^{p} (t - x_i),$$

où l'on choisit A de sorte que $R(x) = 0$. En appliquant plusieurs fois le théorème de Rolle, on montre qu'il existe un $\theta \in I$ (dépendant de x) tel que $R^{(p+1)}(\theta) = 0$. De cela, il vient :

$$\forall x \in I, \quad \exists \theta \in I, \quad f(x) - P_0(x) = \frac{f^{(p+1)}(\theta)}{(p+1)!} \prod_{i=0}^{p} (x - x_i).$$

Exercice 3.1.3 *Trouver à l'aide de SCILAB le polynôme de degré minimal d'interpolation de* sin *aux points* $i\pi/4$ *(*$i = 0, \ldots, 4$*). Toujours à l'aide de SCILAB, comparer l'erreur théorique maximale sur* $[0; \pi]$ *avec l'erreur estimée empiriquement.*

Nous en venons à la démonstration de la formule d'erreur mixte, sur un exemple particulier, mais qui est suffisamment représentatif pour comprendre le cas général.

Exercice 3.1.4 *Soit* f *une fonction définie sur un intervalle ouvert* I*, trois fois dérivable sur* I*, et* P_0 *le polynôme d'interpolation de* f*, de degré minimal, satisfaisant* $P_0(x_1) = f(x_1)$*,* $P_0'(x_1) = f'(x_1)$ *et* $P_0(x_2) = f(x_2)$*. On fixe* x *dans l'intervalle* I *distinct des* x_i*, et l'on considère :*

$$g(t) = f(t) - P_0(t) - A(t - x_1)^2(t - x_2),$$

où A *va être choisi immédiatement.*
1. *On choisit* A *de sorte que* $g(x) = 0$*. Quelle est la valeur de* A *?*
2. *Montrer que* g' *s'annule en trois points distincts (on remarquera que* $g'(x_1) = 0$*).*
3. *En déduire qu'il existe* ξ_x *de sorte que* $g'''(\xi_x) = 0$*, puis aboutir à la formule d'erreur.*

3.2 Moindres Carrés.

On dispose de n observations (x_i, y_i) qui semblent être proches d'une courbe bien connue (droite, parabole). Contrairement à la section précédente où l'on cherchait l'expression d'une fonction passant par tous les points, cette fois on va chercher une courbe avec peu de paramètres mais passant près de chacun des points.

3.2.1 Un exemple.

Supposons que vous estimiez que trois variables x, y, z soient reliées par une relation du type $z = \alpha x + \beta y^2$ avec α et β inconnues que vous cherchiez à estimer. On observe n triplets (x_i, y_i, z_i) avec $n \geq 2$ (2 est le nombre de paramètres) si possible n beaucoup plus grand. Il est bien entendu peu probable que pour chaque i, on ait une relation exacte $z_i = \alpha x_i + \beta y_i^2$ avec les mêmes α et β. On écrit donc cette relation sous la forme :

$$z_i = \alpha x_i + \beta y_i^2 + u_i,$$

le terme u_i étant à interpréter comme un terme d'erreur. Si l'on pose $b = (\alpha, \beta)'$ (vecteur 2×1), $u = (u_1, ..., u_n)'$ (vecteur $n \times 1$ dit vecteur des résidus) et $Y = (z_1, ..., z_n)'$ (vecteur $n \times 1$), on peut écrire notre relation sous la forme :

$$Y = Xb + u$$

où X est une matrice $n \times 2$ dont la $i-$ème ligne est : $(x_i \ \ y_i)$. Bien entendu, dès que $n > 2$, il y a peu de chances que l'on puisse trouver une solution correspondant à $u = 0$. On va donc chercher un (le) b minimisant $\|u\|$, la norme euclidienne de l'erreur.

3.2.2 La théorie.

Un problème de Moindres Carrés (Ordinaires) s'écrit sous la forme :

$$Y = Xb + u$$

où Y est de taille $N \times 1$, X de taille $N \times K$ (avec $N > K$), b de taille $K \times 1$ et u est le vecteur des erreurs (appelé aussi vecteur des résidus) $N \times 1$. Ce problème est courant en économétrie. On va chercher à minimiser la norme de u. Pour des raisons pratiques, on choisit la norme euclidienne $\|.\|_2$. On considère donc la fonction :

$$\phi : b \mapsto \|Y - Xb\|_2^2$$

définie sur $\mathbb{R}^K$ et à valeurs réelles, dont on cherche le (un) minimum.

Avec la seconde partie du cours d'analyse S5, nous allons voir qu'une telle fonction est convexe, et que par conséquent b en est un minimum si et seulement si il annule le gradient donné par $\nabla\phi(b) = -{}^tX(Y - Xb)$. Afin d'éviter de trop anticiper sur le programme, nous allons nous contenter d'une démonstration partielle, en montrant que tout b^* satisfaisant ${}^tXXb^* = {}^tXY$ est bien un minimum de ϕ. En effet, prenant $h \in \mathbb{R}^K$ arbitraire, nous avons :

$$\phi(b^* + h) = \|(Y - Xb^*) - Xh\|_2^2 = \phi(b^*) - 2 < Y - Xb^*, Xh > + \|Xh\|_2^2.$$

Or $2 < Y - Xb^*, Xh >= 2 < {}^tX(Y - Xb^*), h >= 0$, et par conséquent :

$$\phi(b^* + h) - \phi(b^*) = \|Xh\|_2^2 \geq 0.$$

Il reste donc maintenant à se poser la question de savoir si l'équation d'inconnue b : ${}^tXXb = {}^tXY$ a bien des solutions. Cette fois-ci, la question n'est pas si absurde, car la matrice ${}^tXX$ est carrée (de taille $K \times K$). La réponse est fournie par ce lemme :

Lemme 3.2.1 *On a* $\mathrm{Ker}(X) = \mathrm{Ker}({}^tXX)$.

Bien entendu, si $Xv = 0$ alors ${}^tXXv = 0$ on a donc toujours $\mathrm{Ker}(X) \subset \mathrm{Ker}({}^tXX)$. Réciproquement, si ${}^tXXv = 0$, il vient alors $< v, {}^tXXv >= 0$ et donc $\|Xv\|_2^2 = 0$ d'où $Xv = 0$.

Comme conséquence, on voit que la matrice ${}^tXX$ est inversible ssi X est de rang K exactement, ce qui revient à dire que sa famille de vecteurs colonnes est libre. Sous cette hypothèse, le problème de moindres carrés a une solution unique, donné par $({}^tXX)^{-1}({}^tXY)$. On prendra garde à ne pas distribuer l'inverse à des matrices non carrées.

3.2.3 Interprétation géométrique.

Considérons $\mathbb{R}^N$ euclidien canonique, prenons F un sous-espace strict de dimension K dont on se donne une base (non nécessairement orthonormée) $(\varepsilon_1, \ldots, \varepsilon_K)$. Rangeons cette base dans une matrice X de taille $N \times K$ dont la $j-$ème colonne est constituée de ε_j. Alors cette matrice est de rang K exactement.

Lemme 3.2.2 *On a :*

$$F = \{Xb, b \in \mathbb{R}^K\}.$$

En effet, $Xb = \sum_{j=1}^{K} b_j \varepsilon_j$. De ce lemme, on déduit que chercher la projection orthogonale de $Y \in \mathbb{R}^N$ sur F, c'est la chercher sous la forme Xb. On cherche donc à minimiser sur $\mathbb{R}^K$ l'application $b \mapsto \|Y - Xb\|_2$. Ainsi, la solution est donnée par ce qui précède, et le projeté de Y sur F a pour expression :

$$\pi_F(Y) = X({}^tXX)^{-1t}XY.$$

3.2.4 Solution Scilab.

Dans SCILAB, $({}^tXX)^{-1t}XY$ se calcule immédiatement par la séquence :
`X\Y`

Exercice 3.2.3 *Soit $(x_i, y_i)_{1 \leq i \leq n}$, n couples de points. Trouver l'équation de la droite d'ajustement $y = ax + b$. Démontrer qu'elle passe par le point moyen $(\frac{\sum_i x_i}{n}, \frac{\sum_i y_i}{n})$.*

Exercice 3.2.4 *Trouver l'équation de la parabole approchant au mieux les points $(-1; 3, 1)$, $(-2; 7)$, $(1; 0, 8)$, $(2; 2, 5)$.*

3.3 Calcul de séries.

Dans cette section, on calcule des sommes partielles de différentes séries pour exhiber leur éventuelle convergence. Par la suite, on se sert de séries pour calculer des valeurs approchées de π. Enfin, on se rendra compte qu'un calcul numérique même très précis est souvent insuffisant pour connaître la nature d'une série.

3.3.1 Théorie.

Le problème posé dans cette section est simple. On considère une série $\sum u_n$ où $(u_n)_{n \in \mathbb{N}}$ est une suite de nombres réels. On considérera la suite $(S_n)_{n \in \mathbb{N}}$ des sommes partielles de la série définie par :

$$S_n = \sum_{k=0}^{n} u_k \quad \text{pour} \quad n \in \mathbb{N}.$$

Lorsque la série est convergente, on note S la somme et on introduit le reste d'ordre n de la série, défini par

$$R_n = \sum_{k=n+1}^{\infty} u_k \quad \text{pour} \quad n \in \mathbb{N}.$$

Le problème est le calcul approchée de la somme (infinie) de la série. Pour cela, on la remplace par une somme finie, l'erreur commise est alors égale au reste, qu'il s'agit d'estimer. Nous allons étudier deux situations particulières.

Le cas des séries alternées.

Dans ce qui suit, on suppose que $u_n = (-1)^n v_n$ ou $u_n = (-1)^{n+1} v_n$, où $(v_n)_n$ est une suite décroissante de réels positifs. Dans ce cas, on démontre que les suites extraites des sommes partielles $(S_{2n})_n$ et $(S_{2n+1})_n$ sont adjacentes, ce qui permet d'assurer que la série converge et que le reste est du signe de son premier terme et majoré par celui-ci (en valeurs absolue). Précisément :

$$|S - S_n| \leq |u_{n+1}| \qquad \text{et} \qquad \operatorname{sgn}(S - S_n) = \operatorname{sgn}(u_{n+1}).$$

Le cas des séries à termes décroissants.

Ici, nous supposerons que $u_n = f(n)$ où f est une fonction positive, décroissante de limite nulle dont on connaît explicitement les primitives. Le point crucial est le lemme suivant :

Lemme 3.3.1 *pour tout entier k, nous avons :*

$$f(k+1) \leq \int_k^{k+1} f(t)dt \leq f(k).$$

Ce lemme, de démonstration triviale, permet de démontrer que la série $\sum_n u_n$ et l'intégrale $\int^{+\infty} f$ sont simultanément convergentes ou divergentes. En plus :

- en cas de divergence, nous avons :

$$\int_0^{n+1} f(t)dt \leq S_n \leq u_0 + \int_0^n f(t)dt;$$

- en cas de convergence, nous avons :

$$\int_{n+1}^{+\infty} f(t)dt \leq R_n \leq \int_n^{+\infty} f(t)dt.$$

Ainsi, dans le cas de convergence, on a un encadrement du reste, et donc de $S - S_n$.

3.3.2 Exemples.

1. Soit $u_n = \dfrac{1}{(n+1)^4}$ pour $n \in \mathbb{N}$. Rappeler le résultat théorique concernant la nature de cette série. Faire un programme permettant de tracer S_n en fonction de n avec n variant de 1 à N. La suite (S_n) semble-t-elle converger ? Faire la même chose avec $u_n = \dfrac{1}{(n+1)^2}$. Quelle série semble converger le plus rapidement ?

2. On peut montrer (à l'aide des séries de Fourier) que :

$$\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6} \quad \text{et} \quad \sum_{k=1}^{\infty} \frac{1}{k^4} = \frac{\pi^4}{90}.$$

   En déduire, pour les deux séries, le tracé de R_n en fonction de n avec n variant de 1 à N (on fera les 2 tracés sur le même graphe). On conjecture que lorsque n tend vers l'infini, $R_n \sim \frac{a}{n^\beta}$ et l'on cherche à trouver β (qui est un indicateur de la vitesse de convergence). On fait, pour les deux séries, le tracé de $\ln(R_n)$ en fonction de $\ln(n)$ avec n variant de 1 à 100 ; justifiez ce choix. Quelles valeurs conjecturez-vous pour β dans chacun des cas ? Justifiez théoriquement le résultat obtenu à l'aide de la section précédente.

3. Sachant que lorsque p est un entier pair, $\sum_{k=1}^{\infty} \frac{1}{k^p}$ est de la forme $r_p \pi^p$ où r_p est un rationnel connu ($r_2 = 1/6$, $r_4 = 1/90$, $r_6 = 1/945$, $r_8 = 1/9450$,...), on peut pour tout p estimer π par la suite :

$$\pi_p(n) = \left(\frac{\sum_{k=1}^{n} \frac{1}{k^p}}{r_p} \right)^{1/p}.$$

   Donner un équivalent lorsque n tend vers l'infini de $\pi - \pi_p(n)$ et en déduire le choix de p optimal parmi les valeurs 2, 4, 6, 8.

4. Soit la série $\sum_{n=0}^{+\infty} u_n$ avec $u_n = (-1)^n \dfrac{4}{2n+1}$. On admettra que cette série a pour somme π (ce qui se montre facilement avec le cours sur les séries entières). Vérifier numériquement que S_{10000} est proche de π. Calculer R_n pour $n = 100, 1000, 10000$ et le comparer avec la majoration du reste obtenue à partir du critère de convergence des séries alternées. Calculer une approximation de π à partir de la série à 10^{-4} près.

3.3.3 Un exemple d'accélération de convergence.

On suppose connus les résultats de la section précédente. On veut calculer une valeur approchée de :

$$S = \sum_{n=0}^{+\infty} \frac{1}{(n+1)^2 + 1}.$$

On note $S_p = \sum_{n=0}^{p} \frac{1}{(n+1)^2+1}$ la somme partielle.
1. Trouver un entier N assurant que $|S - S_N| \leq 10^{-4}$. La convergence semble t-elle rapide ?
2. On note $S'_p = \sum_{n=0}^{p} \frac{1}{(n+1)^2}$, et l'on rappelle que S'_p tend vers $\pi^2/6$. Exprimer en fonction de S la limite S'' de $S''_p = S_p - S'_p$.
3. Trouver un entier N assurant que $|S'' - S''_N| \leq 10^{-4}$. En déduire un nombre $\hat{S}$ tel que $|S - \hat{S}| \leq 10^{-4}$.

3.3.4 Limites du calcul numérique.

Cette partie a pour but de montrer que quelque soient les performances des logiciels de calcul numérique, il y a encore un sens et de l'intérêt à faire des mathématiques "théoriques"...

1. Soit la série de terme général $u_n = \dfrac{1}{n+1}$. Rappeler le résultat théorique sur la convergence de cette série. Tracer S_n en fonction de n avec n variant de 1 à 10000. La suite (S_n) semble-t-elle converger ? tracer alors $S_n - \ln(n)$ en fonction de n avec n variant de 1 à 10000. Conclusions ? Expliquer théoriquement le résultat.

2. Soit les séries de termes généraux $u_n = \dfrac{n!}{1000^n}$ et $v_n = \dfrac{1000^n}{n!}$. Numériquement, déterminer quelle série converge et quelle série diverge. Répondre à la même question théoriquement. Conclusion ?

3.4 Racines et extrema de fonctions d'une variable.

On suppose que l'on veut résoudre une équation de type

$$f(x) = 0 \quad \text{pour} \quad x \in [0, 1]. \tag{3.1}$$

Ce type d'équations recoupe un grand nombre de problèmes mathématiques, comme celui de trouver la solution de $y_0 = g(x)$ avec $x \in [a, b]$, de déterminer un extremum local à une fonction (la condition nécessaire du premier ordre sur un ouvert porte sur l'annulation de la dérivée),... Dans la plupart des cas (à part les rares cas d'école habituels), on ne peut pas trouver une solution théorique à cette équation (voir l'exemple ci-dessous). Aussi est-on amené à calculer de façon approchée l'ensemble des solutions de cette équation. Par la suite, nous étudions différentes méthodes numériques classiques de résolution de cette équation.

Dans toute la suite nous considérerons l'exemple de la fonction f suivante :

$$f(x) = x^3 + x - 1. \tag{3.2}$$

3.4.1 Méthode de dichotomie.

La première méthode, la plus naturelle, pour obtenir une valeur approchée de $\overline{x}$ est une méthode d'approximation successive dite par "dichotomie". Elle consiste à d'abord fixer un intervalle $[a_0, b_0]$ dans lequel f admet une unique racine. Ainsi, $f(a_0).f(b_0) < 0$. Ensuite, on considère le milieu du segment $[a_0, b_0]$, soit $\frac{1}{2}(a_0 + b_0)$ et on calcule numériquement $f\left(\frac{1}{2}(a_0 + b_0)\right)$. Si $f(a_0).f\left(\frac{1}{2}(a_0 + b_0)\right) < 0$ alors x_1 est dans $\left[a_0, \frac{1}{2}(a_0 + b_0)\right]$, on pose $a_1 = a_0$, $b_1 = \frac{1}{2}(a_0 + b_0)$ et on recommence le méme procédé. Sinon, x_1 est dans $\left[\frac{1}{2}(a_0 + b_0), b_0\right]$, on pose $b_1 = b_0$, $a_1 = \frac{1}{2}(a_0 + b_0)$ et on recommence le méme procédé.
Programmer cette méthode (fichier *dicho.sci*) pour une fonction f quelconque définie dans un fichier *f.sci* et d'abord pour 10 itérations. Introduire ensuite dans le programme le calcul du nombre d'itérations nécessaires pour une approximation *Delta* donnée (par exemple $Delta = 0.0001$) en fonction de *Delta*, a_0 et b_0.

3.4.2 Méthode issue du théorème du point fixe.

Rappelons l'énoncé du théorème du point fixe, dans le cadre qui va nous intéresser.

Théorème du point fixe. Soit I un intervalle fermé de $\mathbb{R}$, $g : I \to I$ une fonction vérifiant :

$$\exists k \in [0; 1[, \quad \forall (x, y) \in I^2, \quad |g(x) - g(y)| \leq k|x - y|.$$

Alors l'équation $g(x) = x$ a une unique solution x^*, et pour toute valeur $x_0 \in I$, la suite récurrente $x_{n+1} = g(x_n)$ converge vers x^*. Plus précisément, on a :

$$|x_n - x^*| \leq \frac{k^n}{1 - k}|g(x_0) - x_0|.$$

On appréciera ce théorème qui, sous l'hypothèse de contraction, donne l'existence, l'unicité de la solution à l'équation $g(x) = x$ avec en plus un mode explicite de calcul, et la vitesse de convergence. En pratique, pour vérifier la condition de Lipschitz, lorsque g est de classe C^1, on calcule $k = \sup_I^o |g'|$. Cette constante k est la meilleure possible sur I. Donc la fonction est contractante ssi cette constante est strictement inférieure à 1. On notera aussi que, théoriquement, la convergence est d'autant plus rapide que k est petit.

Si l'on veut utiliser le théorème du point fixe, il faut donc écrire notre équation $f(x) = 0$ sous la forme $x = g(x)$, avec une constante $\sup_I |g'|$ strictement inférieure à 1, et aussi petite que possible (voir exercice conclusif).

3.4.3 Méthode de la sécante et méthode de Newton.

On se place encore dans le cadre où l'on sait que la fonction admet une unique racine dans $[a_0, b_0]$. Ces deux méthodes sont basées sur un principe commun au départ, qui consiste à remplacer la fonction par un polynôme d'interpolation dont on calcule les racines. Pour la facilité de calcul, on prend des polynômes de degré 1. La seule différence entre les deux méthodes consiste en le choix du polynôme.

Méthode de la sécante. On donne pour valeur approchée x_1 de $\overline{x}$ le point d'intersection entre la droite passant par les points $(a_0, f(a_0))$ et $(b_0, f(b_0))$ et l'axe des abscisses, c'est-à-dire que l'on remplace f par son polynôme d'interpolation de Lagrange aux points a_0 et a_1. Ce polynôme est une droite d'équation :

$$\frac{y - f(a_0)}{x - a_0} = \frac{f(b_0) - f(a_0)}{b_0 - a_0}.$$

Lorsque x vaut x_1, y vaut 0 ce qui donne l'expression de x_1 :

$$x_1 = a_0 - f(a_0).\frac{b_0 - a_0}{f(b_0) - f(a_0)}.$$

On cherche maintenant à établir une formule d'erreur, en supposant que f est de classe C^2 sur $[a_0, b_0]$. On pose $m_1 = \inf_{x\in[a_0,b_0]} |f'(x)| > 0$ et $M_2 = \sup_{x\in[a_0,b_0]} |f''(x)|$.

1. Soit P le polynôme de degré au plus 1 tel que $P(a_0) = f(a_0)$ et $P(b_0) = f(b_0)$. On sait par la formule d'erreur sur l'interpolation que pour tout $t \in [a_0, b_0]$, on a :

$$|f(t) - P(t)| \leq \frac{M_2}{2}(t - a_0)(b_0 - t).$$

L'étude de la fonction $t \mapsto (t - a_0)(b_0 - t)$ montre que sa valeur maximale est $(b_0 - a_0)^2/4$, ce qui fournit :

$$\forall t \in [a_0, b_0], \quad |f(t) - P(t)| \leq \frac{M_2}{8}(b_0 - a_0)^2.$$

On en déduit en particulier que :

$$|P(\overline{x})| \leq \frac{M_2}{8}(b_0 - a_0)^2.$$

Grâce au théorème de Thalès, on sait que :

$$\frac{\overline{x} - x_1}{\overline{x} - a_0} = \frac{P(\overline{x}) - P(x_1)}{P(\overline{x}) - P(a_0)} = \frac{P(\overline{x})}{P(\overline{x}) - P(a_0)} = \frac{P(\overline{x})}{\overline{x} - a_0}\frac{\overline{x} - a_0}{P(\overline{x}) - P(a_0)},$$

donc :

$$\frac{|\overline{x} - x_1|}{|\overline{x} - a_0|} = \frac{|P(\overline{x})|}{|\overline{x} - a_0|}\left|\frac{\overline{x} - a_0}{P(\overline{x}) - P(a_0)}\right|.$$

Or :

$$\left|\frac{\overline{x}-a_0}{P(\overline{x})-P(a_0)}\right| = \left|\frac{b_0-a_0}{P(b_0)-P(a_0)}\right| = \left|\frac{b_0-a_0}{f(b_0)-f(a_0)}\right| \leq \frac{1}{m_1}.$$

Par conséquent :

$$|\overline{x}-x_1| \leq \frac{M_2}{8m_1}(b_0-a_0)^2.$$

Méthode de Newton. On donne pour valeur approchée x_1 de $\overline{x}$ le point d'intersection entre la tangente à f en a_0 et l'axe des abscisses (on prend donc le polynôme de Taylor de degré 1 en a_0). Montrer qu'alors $x_1 = a_0 - \dfrac{f(a_0)}{f'(a_0)}$. En adaptant la démarche précédente, on montre que si f est de classe C^2 sur $[a_0, b_0]$, si $m_1 = \inf\limits_{x\in[a_0,b_0]} |f'(x)| > 0$ et $M_2 = \sup\limits_{x\in[a_0,b_0]} |f''(x)|$ alors

$$|\overline{x}-x_1| \leq \frac{M_2}{m_1} \times \frac{(b_0-a_0)^2}{2}.$$

3.4.4 Itération des méthodes.

La méthode de la sécante exige deux points pour calculer le point suivant. On peut l'itérer de deux façons :

1. A l'étape n, on dispose d'un intervalle $[a_n, b_n]$. On calcule l'intersection c_n du polynôme interpolateur avec l'axe des abscisses, puis on choisit pour $[a_{n+1}, b_{n+1}]$ celui des deux intervalles $[a_n, c_n]$ ou $[c_n, b_n]$ comme pour la dichotomie.

2. On pose $x_{-1} = a_0$, $x_0 = b_0$, et on calcule par récurrence x_{n+1} en se servant des points x_n et x_{n-1}.

L'avantage de la première méthode est qu'elle fournit à chaque étape un intervalle contenant la racine. Dans le cas d'une fonction monotone qui demeure convexe ou concave, l'une des bornes reste fixe tandis que l'autre converge vers la solution, la suite étant monotone bornée. L'avantage de la seconde est que, en cas de convergence, sous certaines hypothèses la vitesse de convergence de la méthode est impressionnante : à chaque étape, le nombre de décimales exactes est approximativement et asymptotiquement multiplié par $\frac{1+\sqrt{5}}{2} \approx 1,62$.

La méthode de Newton s'itère de la manière suivante :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}.$$

A condition de partir du bon côté, elle est toujours convergente dans le cas d'une fonction monotone qui demeure convexe ou concave, la suite étant monotone

bornée. Sous certaines hypothèses la vitesse de convergence de la méthode est impressionnante : à chaque étape, le nombre de décimales exactes est approximativement et asymptotiquement multiplié par 2.

3.4.5 Méthode de Newton et point fixe.

L'objectif de cette méthode est de déterminer une suite (u_n) convergeant vers x_1. Pour cela, on considère une fonction ϕ telle que pour $x \in [a,b]$, $f(x) = 0 \iff \phi(x) = x$ et si ϕ vérifie des propriétés de type $\phi([a_0,b_0]) \subset [a_0,b_0]$ et $\sup_{x\in[a_0,b_0]} |\phi'(x)| < 1$. On définit alors (u_n) par $u_{n+1} = \phi(u_n)$ avec $u_0 \in [a_0,b_0]$ et on a bien (u_n) qui converge vers x_1, unique solution de $x = \phi(x)$. Cette méthode est appelée méthode de Newton lorsque la fonction ϕ considérée est :

$$\phi(x) = x - \frac{f(x)}{f'(x)}.$$

Ce choix est justifié par l'exercice suivant :

Exercice. On veut résoudre $f(x) = 0$ par une méthode de point fixe sur un intervalle compact I dans lequel a été localisée une racine x^*. On supposera f dérivable sur I et que la dérivée ne s'annule pas.
1. Soit φ une fonction dérivable sur I ne s'annulant pas. Montrer que $f(x) = 0$ est équivalente à

$$x = G_\varphi(x),$$

où $G_\varphi(x) = x + \varphi(x)f(x)$.
2. On cherche à avoir la plus petite constante de Lipschitz au voisinage de x^*. Justifier ce choix. On se propose alors de trouver une fonction φ telle que $G'_\varphi(x^*) = 0$. Déterminer alors $\varphi(x^*)$ en fonction de $f'(x^*)$. Proposer une fonction φ solution (elle devra bien entendu ne dépendre que de f, pas de x^*). Quel est le rapport avec la méthode de Newton ?

3.4.6 Convergence des méthodes de la sécante et de Newton.

En général, les méthodes de la sécante et de Newton ne sont pas convergentes. Cependant, avec des hypothèses de monotonie et de convexité, un dessin vous aide à choisir un point de départ assurant que la méthode est convergente. Il est possible de démontrer sous certaines hypothèses la convergence, ainsi que l'erreur $\varepsilon_n = |x_n - \overline{x}|$ au rang n satisfait asymptotiquement une inégalité de la forme :

$$\varepsilon_{n+1} \leq C\varepsilon_n^{\frac{1+\sqrt{5}}{2}},$$

dans le cas de la méthode de la sécante, et

$$\varepsilon_{n+1} \leq C\varepsilon_n^2,$$

pour la méthode de Newton. Ces inégalités expliquent l'extrême rapidité de ces méthodes, lorsqu'on est dans le domaine où elles sont valides :

Exercice On considère un procédé de calcul approché d'une solution x^* d'un problème par une suite $(x_n)_n$ que l'on suppose convergente vers x^*, sans prendre la valeur x^*. On note $\varepsilon_n = |x_n - x^*|$ l'erreur absolue au rang n, et l'on suppose qu'il existe des constantes $C > 0$ et $\alpha > 1$ tels que, pour n assez grand :

$$\varepsilon_{n+1} \leq C\varepsilon_n^\alpha.$$

On dit alors que pour n assez grand, le nombre de décimales exactes est en gros multiplié par α à chaque étape. Expliquez pourquoi (on regardera l'inéquation de récurrence satisfaite par $\lambda_n = -\log_{10}(\varepsilon_n)$).

3.4.7 Un exercice conclusif.

Exercice. On considère le polynôme $f(x) = x^3 + x - 1$.
1. Montrer que f admet une unique racine réelle x^*, et que $x^* \in]0;1[$. On va s'intéresser à calculer x^* par diverses méthodes.
2. Trouver un encadrement d'amplitude 10^{-6} de x^* en programmant la méthode de dichotomie dans SCILAB.
3. Retrouver une valeur approchée par la méthode de la sécante, puis par la méthode de Newton. Comparer empiriquement les vitesses de convergence des trois méthodes.
4. Désormais, on va utiliser des méthodes de point fixe sur $I = [0;1]$. On cherche donc une fonction $g : I \to I$ contractante telle que l'équation $f(x) = 0$ soit équivalente à $x = g(x)$.
4.a. Montrer que $P(x) = 0$ peut s'écrire $x = g(x)$ avec chacune des fonctions g suivantes:

- $g(x) = 1 - x^3$.
- $g(x) = \sqrt[3]{1-x}$.
- $g(x) = \frac{1}{x^2+1}$.

Dans chacun des cas, étudier si les hypothèses du théorème du point fixe sont satisfaites. Si tel est le cas, appliquer la méthode pour avoir un intervalle d'amplitude 10^{-6} contenant x^*.
4.b. Soit $\lambda \in \mathbb{R}^*$. On introduit $g_\lambda : x \mapsto x + \lambda f(x)$.
4.b.i. Montrer que pour tout $\lambda \in \mathbb{R}^*$, $f(x) = 0$ est équivalent à $x = g_\lambda(x)$.
4.b.ii. Trouver l'ensemble J des valeurs de λ de sorte que $g_\lambda(0)$ et $g_\lambda(1)$ soient dans I.
4.b.iii. On considère l'ensemble $\Lambda = [-\frac{1}{4};0[$. Montrer que si $\lambda \in \Lambda$, on a $g_\lambda(I) \subset I$ et g_λ contractante sur I.
4.b.iv. On peut donc choisir toute valeur $\lambda \in \Lambda$ pour appliquer le théorème du point fixe. Comment allez vous en pratique choisir λ pour avoir une bonne vitesse de convergence ?

3.5 Calcul approchée de dérivées.

3.5.1 Introduction.

Le calcul approché de dérivées peut-être présenté à partir des formules de Taylor. Supposons que l'on cherche à approcher $f'(x_0)$ et que f soit deux fois dérivable. Les formules de Taylor nous apprennent que si k est petit, $f(x_0+k)$ est approximativement égal à $f(x_0)+f'(x_0)k$, et que de plus l'erreur est majorée par $\frac{M_2k^2}{2}$, c'est-à-dire que l'on a :

$$|f(x_0+k)-(f(x_0)-f'(x_0)k)| \leq \frac{M_2k^2}{2}.$$

Divisant par h supposé non nul, on obtient :

$$\left|f'(x_0)-\frac{f(x_0+k)-f(x_0)}{k}\right| \leq \frac{M_2}{2}|k|.$$

Ainsi, une bonne approximation de $f'(x_0)$ est donnée par le taux d'accroissement $\frac{f(x_0+k)-f(x_0)}{k}$, avec une erreur majorée par $\frac{M_2}{2}|k|$ si f est deux fois dérivable au voisinage de x_0.

En général, on se donne le pas $h>0$ petit. Le calcul précédent donne alors deux formules possibles d'approximation de $f'(x_0)$ selon que l'on prenne directement h ou $-h$ pour k :

$$f'(x_0) \approx \frac{f(x_0+h)-f(x_0)}{h},$$
$$f'(x_0) \approx \frac{f(x_0)-f(x_0-h)}{h}.$$

Dans les deux cas, l'erreur est majorée par $\frac{M_2}{2}h$. En faisant la moyenne des deux, on trouve une troisième formule :

$$f'(x_0) \approx \frac{f(x_0+h)-f(x_0-h)}{2h}.$$

Cette formule symétrique donne de meilleurs résultats numériquement, comme nous allons le voir un peu plus bas.

3.5.2 Généralités sur les méthodes.

Afin d'écrire de manière unifiée ces formules, notons qu'elles s'écrivent toutes sous la forme :

$$A_1f(x_0+\theta_1h)+A_2f(x_0+\theta_2h),$$

où $h>0$ est petit, les deux $x_0+\theta_jh$ (θ_j distincts) sont appelés les points de la formule. De manière générale, une formule n points prend la forme :

$$\sum_{j=1}^{n} A_jf(x_0+\theta_jh).$$

On dira qu'elle est d'ordre $p \in \mathbb{N}$ si pour tout polynôme P de degré au plus p, on a exactement :

$$P'(x_0) = \sum_{j=1}^{n} A_j P(x_0 + \theta_j h).$$

Théorème 3.5.1 *Lorsque $n \geq 2$, il existe un choix et un seul des A_j de sorte que la formule soit d'ordre $n-1$.*

En effet, on aboutit à un système de Van der Monde. De manière générale, on peut approcher ainsi toute dérivée d'ordre au plus $n-1$, de manière exacte sur les polynômes d'ordre au plus $n-1$.

3.5.3 Retour sur les exemples.

Cherchons pour approcher la dérivée première une formule à 2 points :

$$f'(x_0) \approx A_1 f(x_0 + \theta_1 h) + A_2 f(x_0 + \theta_2 h).$$

Le fait qu'elle soit d'ordre au moins 1 aboutit au système :

$$\begin{cases} A_1 + A_2 = 0 \\ (A_1\theta_1 + A_2\theta_2)h = 1 \end{cases}$$

Ceci donne la formule :

$$f'(x_0) \approx \frac{f(x_0 + \theta_2 h) - f(x_0 + \theta_1 h)}{(\theta_2 - \theta_1)h}.$$

Pour contrôler l'erreur, on fait un développement de Taylor d'ordre 2 du terme de gauche :

$$f(x_0 + \theta_2 h) - f(x_0 + \theta_1 h) = (\theta_2 - \theta_1)hf'(x_0) + \frac{\theta_2^2 - \theta_1^2}{2} f''(x_0)h^2 + O(h^3).$$

Par conséquent :

$$\frac{f(x_0 + \theta_2 h) - f(x_0 + \theta_1 h)}{(\theta_2 - \theta_1)h} - f'(x_0) = \frac{\theta_2 + \theta_1}{2} f''(x_0)h + O(h^2).$$

On voit ainsi que si $\theta_1 + \theta_2 \neq 0$, l'erreur est en h (et dans nos deux premiers exemples serait de l'ordre de $\frac{f''(x_0)}{2}h$), tandis que pour une formule symétrique elle serait en h^2. En fait, pour être plus précis, on devrait utiliser Taylor-Lagrange : dans les deux premiers cas, l'erreur est majoré par $\frac{M_2}{2}h$, et pour le dernier en $\frac{M_3}{6}h^2$.

Venons en au calcul approchée d'une dérivée seconde. La formule de Taylor à l'ordre 2 permet décrire que pour λh petit :

$$f(x_0 + \lambda h) \approx f(x_0) + f'(x_0)\lambda h + \frac{f''(x_0)}{2}\lambda^2 h^2.$$

On va se débarasser du terme $f'(x_0)$ qui ne nous intéresse pas ici. Pour cela, on ajoute les formules obtenues avec $\lambda = 1$ et $\lambda = -1$, ce qui permet de dire que pour $h > 0$ petit :

$$f'(x_0 + h) + f(x_0 - h) \approx 2f(x_0) + f''(x_0)h^2,$$

d'où une formule à 3 points :

$$f''(x_0) \approx \frac{f(x_0 + h) + f(x_0 - h) - 2f(x_0)}{h^2},$$

qui est d'ordre 3, avec une erreur majorée par $\frac{M_4}{12}h^2$.

Exercice 3.5.2 *On considère la fonction f définie sur $\mathbb{R}$ par $f(x) = x^3$.*
1. *Déterminer des valeurs approchées de $f'(1)$ par les trois formules vues en cours (diminuer petit à petit le h). L'une d'elle semble converger plus vite, pourquoi ?*
2. *Estimer $f''(1)$ par la formule vue en cours. Que se passe t-il lorsqu'on diminue h ? Pourquoi ?*

3.5.4 Fonctions de plusieurs variables.

La théorie sera exposée oralement. Voici un exercice :

Exercice 3.5.3 *On considère la fonction :*

$$f(x, y) = x^2 + xy + y^2 + x + y.$$

Déterminer de manière approchée son gradient et sa hessienne au point $a = (-\frac{1}{3}, -\frac{1}{3})$. Quelle est la signature de la hessienne en a ?

3.6 Calcul d'intégrales.

3.6.1 Introduction.

En dehors de cas d'école présentés pendant les cours de mathématiques, il est en général impossible de calculer théoriquement la valeur numérique d'une intégrale définie. On doit alors utiliser des méthodes de calcul numérique pour donner une valeur approchée d'intégrales. Sauf dans le cas des méthodes de Gauss, nous considérerons des intégrales du type :

$$I(f) = \int_a^b f(t)dt,$$

avec $-\infty < a < b < +\infty$ et f intégrable au sens de Riemann sur $[a, b]$ (nous exigerons souvent une régularité plus forte pour obtenir des formules d'erreur).

Les méthodes que nous allons présenter ici partent toutes d'un principe général, qui consiste à choisir un entier $p \geq 1$ et des points[1] $x_1, \ldots, x_p$ deux à deux distincts dans $[a, b]$, et d'approcher l'intégrale $I(f)$ par une formule de la forme :

$$I_p(f) = \sum_{i=1}^{p} a_i f(x_i).$$

Ainsi, comme pour le calcul de dérivées, nous allons choisir une formule à p points.

Pour conserver l'homogénéité, nous allons introduire les $\theta_i \in [0; 1]$ tels que pour tout i, $x_i = a + \theta_i(b - a)$. Les θ_i indiquent la position relative des x_i par rapport aux points extrêmes $[a, b]$. Ainsi, $\theta_i = 0$ indique que $x_i = a$, $\theta_i = 1$ indique que $x_i = b$, $\theta_i = 1/2$ indique que x_i est le milieu du segment.

3.6.2 Généralités sur les méthodes élémentaires.

Présentation des méthodes.

Ces méthodes consistent, une fois p et les x_i fixés, à choisir les a_i de sorte que la formule d'approximation

$$I(f) \approx I_p(f)$$

donne une égalité pour f polynômiale de degré le plus grand possible. Le degré maximal pour lequel il y a égalité s'appelle *l'ordre* de la méthode.

Il paraît notamment raisonnable d'exiger que la méthode soit exacte pour les polynômes de Lagrange associés aux x_i, les L_i. L'égalité $I(L_i) = I_p(L_i)$ détermine les a_i de manière unique :

$$\forall i = 1, \ldots, p, \qquad a_i = I(L_i).$$

Avec ce choix des a_i, on montre grâce à la linéarité que la méthode est au moins d'ordre $p - 1$. On peut montrer que l'ordre maximal d'une telle méthode est $2p - 1$ et qu'un seul choix des couples $(a_i, x_i)_{1 \leq i \leq p}$ permet d'atteindre cet ordre (voir la section méthodes de Gauss).

Remarque 3.6.1 *La méthode étant d'ordre au moins $p - 1$, on peut déterminer également les a_i en écrivant successivement que*

$$I(1) = I_p(1), I(X) = I_p(X), \ldots, I(X^{p-1}) = I_p(X^{p-1}).$$

Cela donne un système linéaire de type Van der Monde en les a_i. Il admet donc une solution unique.

[1] On notera ici que l'indice commence à 1, il y a donc p points en tout.

Notons c_i les coefficients obtenus lorsque $a = 0$ et $b = 1$ (c'est-à-dire lorsque $x_i = \theta_i$ pour tout i). On a donc une formule d'approximation :

$$\int_0^1 F(\theta)d\theta \approx \sum_{i=1}^{p} c_i F(\theta_i).$$

Posons $F(\theta) = f((1-\theta)a + \theta b)$, puis posons $t = (1-\theta)a + \theta b$ dans l'intégrale. On obtient :

$$\int_a^b f(t)\frac{dt}{b-a} \approx \sum_{i=1}^{p} c_i f(x_i).$$

Ainsi, on voit que $a_i = (b-a)c_i$, où les c_i ne dépendent que des θ_i. D'où une formule de la forme :

$$\int_a^b f(t)dt \approx (b-a)\sum_{i=1}^{p} c_i(\theta_1, \ldots, \theta_p) f(x_i).$$

Ainsi, pour calculer les a_i, on peut déterminer les c_i (en faisant $a = 0$ et $b = 1$, puis les multiplier par $b - a$, ce qui parfois simplifie les calculs dans le système linéaire de la remarque ci-dessus).

Exercice 3.6.2 *On considère une formule d'une méthode élémentaire :*

$$\int_a^b f(x)dx \approx (b-a)\sum_{i=1}^{p} c_i(\theta_1, \ldots, \theta_p) f(x_i)$$

basée sur p points. On note q l'ordre de cette méthode. On rappele que $q \geq p-1$. On note Q le polynôme :

$$Q(X) = \prod_{i=1}^{p}(X - x_i)^2.$$

En utilisant Q, montrer que $q < 2p$. Ainsi l'ordre d'une méthode est toujours au plus égal à $2p-1$. Il est possible de montrer qu'il existe une méthode et une seule d'ordre $2p-1$, elle est basée sur les méthodes de Gauss.

Formule d'erreur.

La méthode étant d'ordre au moins $p-1$, en notant que $I_p(f) = I(P_0)$, où P_0 est le polynôme d'interpolation aux x_i, on obtient facilement une majoration du type :

$$|I(f) - I_p(f)| \leq C(\theta_1, \ldots, \theta_p)\frac{M_p(b-a)^{p+1}}{p!},$$

pour f de classe C^p, où

$$C(\theta_1, \ldots, \theta_p) = \int_0^1 \left|\prod_{j=1}^{p}(t - \theta_j)\right|$$

est une constante inférieure à 1[2] ne dépendant que des θ_i. Si la méthode est d'ordre q supérieur $p-1$, on peut obtenir une majoration du type :

$$|I(f) - I_p(f)| \leq C\frac{M_{q+1}(b-a)^{q+2}}{(q+1)!},$$

pour f de classe C^q, où C est une constante dans $]0;1]$. Cette forme ne semble pas nécessairement à cet instant un gain, mais nous allons voir que dans les méthodes composées le fait d'augmenter la puissance de $(b-a)$ est essentiel.

3.6.3 Exemples classiques.

Méthodes à 1 point.

Commençons sur l'intervalle $[0;1]$. Une méthode à $p=1$ point est d'ordre 0 ou 1. L'approximation prend la forme :

$$\int_0^1 f(t)dt \approx c_1 f(\theta_1).$$

La formule étant exacte si $f=1$, on trouve que nécessairement c_1, ce qui donne la formule :

$$\int_0^1 f(t)dt \approx f(\theta_1).$$

On constate qu'il n'y a égalité pour $f(t)=t$ que lorsque $\theta_1 = 1/2$, ce qui fait que la méthode est d'ordre 0 si $\theta_1 \neq 1/2$ et d'ordre 1 si $\theta_1 = 1/2$. Les trois cas célèbres sont :

- la formule des rectangles à gauche : $\int_0^1 f(t)dt \approx f(0)$;
- la formule des rectangles à droite : $\int_0^1 f(t)dt \approx f(1)$;
- la formule du point milieu : $\int_0^1 f(t)dt \approx f(1/2)$.

Lorsque $\theta_1 \neq 1/2$, le majorant de l'erreur est :

$$\frac{M_1}{1!}\int_0^1 |t-\theta_1|dt = \frac{\theta_1^2 + (1-\theta_1)^2}{2}M_1,$$

et pour $\theta_1 = 1/2$, on peut prendre :

$$\frac{M_2}{2!}\int_0^1 (t-\theta_1)^2 dt = \frac{M_2}{24}.$$

Pour résumer, voici les formules obtenues sur un intervalle $[a,b]$:

[2]bien entendu, si on ne veut pas la calculer, on peut la prendre égale à 1.

- pour $\theta_1 \neq 1/2$:

$$\left| \int_a^b f(t)dt - f(a + \theta_1(b-a)) \right| \leq \frac{\theta_1^2 + (1-\theta_1)^2}{2} M_1 (b-a)^2,$$

- pour $\theta_1 = 1/2$:

$$\left| \int_a^b f(t)dt - f\left((a+b)/2\right) \right| \leq \frac{M_2}{24}(b-a)^3.$$

La formule des trapèzes.

Il s'agit d'une méthode à deux points basés sur les extrémités :

$$\int_0^1 f(t)dt \approx c_1 f(0) + c_2 f(1).$$

Le fait qu'elle soit d'ordre au moins 1 donne $c_1 = c_2 = 1/2$. Ainsi la formule devient :

$$\int_0^1 f(t)dt \approx \frac{f(0) + f(1)}{2}.$$

On vérifie qu'elle est d'ordre 1 exactement, et qu'un majorant de l'erreur est alors :

$$\frac{M_2}{2!} \int_0^1 t(1-t)dt = \frac{M_2}{12}.$$

Sur un intervalle $[a, b]$, on obtient :

$$\left| \int_0^1 f(t)dt - \frac{f(a) + f(b)}{2} \right| \leq \frac{M_2}{12}(b-a)^3.$$

La formule de Simpson.

Il s'agit d'une méthode à trois points basés de la forme :

$$\int_0^1 f(t)dt \approx c_1 f(0) + c_2 f(1/2) + c_3 f(1).$$

Le fait qu'elle soit d'ordre au moins 2 donne les c_j. On trouve :

$$\int_0^1 f(t)dt \approx \frac{f(0) + 4f(1/2) + f(1)}{6}.$$

On vérifie qu'elle est d'ordre 3 exactement, et qu'un majorant de l'erreur est alors :

$$\frac{M_4}{4!} \int_0^1 t(1-t)(1-t/2)^2 dt = \frac{M_4}{2880}.$$

Sur un intervalle $[a, b]$, on obtient :

$$\left| \int_0^1 f(t)dt - \frac{f(a) + 4f((a+b)/2) + f(b)}{6} \right| \leq \frac{M_4}{2880}(b-a)^5.$$

3.6.4 Méthodes composées.

Théorie.

Les méthodes précédentes souffrent d'un défaut que font apparaître les formules d'erreur. Il faut augmenter le nombre de points et recalculer les a_i pour augmenter la précision. Aussi, au lieu de les appliquer directement, on applique souvent une méthode composée. Celle-ci consiste à choisir un entier n, à découper l'intervalle $[a, b]$ en des intervalles $J_1, \ldots, J_n$ d'intérieurs deux à deux disjoints et dont la réunion fait $[a, b]$. On a alors :

$$I(f) = \sum_{j=1}^{p} \int_{J_j} f(t)dt.$$

On applique alors une méthode élémentaire sur chaque J_j.

En général, on fait un découpage régulier : $J_j = [a + (j-1)/n(b-a); a + j/n(b-a)]$ et l'on applique la même méthode sur chaque intervalle. Suivant ce principe général, si la méthode appliquée sur chaque sous intervalle est d'ordre $q \geq p$, on obtient une erreur totale de la forme[3] :

$$C\frac{M_{q+1}(b-a)^{q+2}}{n^{q+1}(q+1)!}.$$

On voit ici l'interêt d'avoir q le plus grand possible (en raison du terme $1/n^{q+1}$).

Exemples classiques.

On découpe l'intervalle $[a, b]$ de manière régulière. Notons $t_k = a + k\frac{b-a}{n}$ et $t_{k+1/2} = \frac{t_k + t_{k+1}}{2} = a + (k + 1/2)\frac{b-a}{n}$. Nous avons :

- **Formule des rectangles à gauche** :

$$R_g = \frac{b-a}{n} \sum_{k=0}^{n-1} f(t_k),$$

erreur :

$$|I - R_g| \leq \frac{M_1(b-a)^2}{2n}.$$

- **Formule des rectangles à droite** :

$$R_d = \frac{b-a}{n} \sum_{k=1}^{n} f(t_k),$$

erreur :

$$|I - R_d| \leq \frac{M_1(b-a)^2}{2n}.$$

[3] A ce niveau, le fait que C ne dépende que des θ_i est essentiel.

- **Formule du point milieu** :

$$PM = \frac{b-a}{n}\sum_{k=0}^{n-1} f(t_{k+1/2}),$$

erreur :

$$|I - R_g| \leq \frac{M_2(b-a)^3}{24n^2}.$$

- **Formule des trapèzes** :

$$Trap = \frac{b-a}{n}\left(\frac{f(a)}{2} + \sum_{k=1}^{n-1} f(t_k) + \frac{f(b)}{2}\right) = \frac{Rg + Rd}{2},$$

erreur :

$$|I - Trap| \leq \frac{M_2(b-a)^3}{12n^2}.$$

- **Formule de Simpson** :

$$Simps = \frac{b-a}{3n}\left(\frac{f(a)+f(b)}{2} + \sum_{k=1}^{n-1} f(t_k) + 2\sum_{k=0}^{n-1} f(t_{k+1/2})\right) = \frac{Trap + 2PM}{3}.$$

erreur :

$$|I - Trap| \leq \frac{M_4(b-a)^5}{2880n^4}.$$

Testons ces formules sur un exercice.

Exercice 3.6.3 *On veut calculer une valeur approchée de l'intégrale :*

$$I = \int_0^1 e^{-t^2} dt.$$

1. *Donner une valeur approchée de I à 10^{-7} près (soit une valeur a t.q. $|a - I| \leq 5 \times 10^{-8}$) en exprimant cette intégrale sous forme d'une série rapidement convergente.*
2. *Tester les méthodes vues précédemment et les comparer (on admettra que $M_1 = \sqrt{2}e^{-1/2}$, $M_2 = 2$ et $M_4 = 12$). On cherchera à réaliser un calcul approché à 10^{-5} près, grâce au majorant de l'erreur, puis on comparera avec le choix de n qui convient en pratique.*

Exercice 3.6.4 *On veut calculer une valeur approchée de l'intégrale :*

$$I = \int_0^1 f(t) dt,$$

où $f(t) = \min\{2t; 1\}$ (bien entendu, cette intégrale se calcule à vue). On veut faire un découpage avec $n = 3$ et appliquer la formule des rectangles à gauche sur chaque sous intervalle.
1. (Répondre sans aucun calcul). *Expliquer pourquoi le découpage en parties égales n'est pas optimal. Quel découpage proposeriez vous ?*
2. *Connaissant la valeur exacte, trouver un meilleur découpage que le régulier.*

3.6.5 Calcul d'intégrales : méthodes de Gauss.

Les méthodes de Gauss relèvent d'un esprit assez différent, puisque cette fois nous n'allons pas fixer avant les x_i. On a vu auparavant que si l'on fixe les x_i, on a un choix et un seul des a_i assurant que la méthode est au moins d'ordre p, et que parfois la méthode est d'ordre supérieur. Nous allons voir qu'il y a un choix et un seul des couples (x_i, a_i) assurant que la méthode soit (au moins) d'ordre $2p-1$. Ce choix fournit alors une méthode d'ordre $2p-1$ exactement. Il s'agit donc d'une méthode élémentaire, mais il est bien entendu possible de la composer.

La formule de Gauss est en fait d'une grande souplesse. Elle s'applique à des calculs d'intégrales :

$$I(f) = \int_a^b f(t)\omega(t)dt$$

où l'intervalle d'intégration n'est pas nécessairement borné[4]. On supposera la fonction ω continue, positive et non constamment nulle sur $]a, b[$. On suppose enfin que l'on connaît explicitement les intégrales suivantes, et qu'elles sont finies :

$$J_k = \int_a^b t^k \omega(t)dt, \quad k \in \mathbb{N}.$$

On cherche donc une formule d'interpolation de la forme :

$$I(f) \approx I_p(f) = \sum_{i=1}^{p} a_i f(x_i)$$

(avec pour tout i, $a_i > 0$ et $x_i \in]a, b[$), de sorte que la formule soit juste pour les polynômes de degré le plus grand possible. Heuristiquement, on dispose de $2p$ valeurs à choisir, et on peut donc espérer avoir une formule valide pour les polynômes de degré (au plus) $2p-1$. Bien entendu ceci n'est pas rigoureux (on a notamment un système non linéaire en les x_i). Il est remarquable que l'intuition soit correcte :

Théorème 3.6.5 *Il existe un choix unique des a_i et des $x_i \in]a, b[$ de sorte que pour tout polyôme P de degré au plus $2p-1$, on ait :*

$$I(P) = I_p(P).$$

De plus, les a_i sont strictement positifs. Enfin, il existe un polynôme de degré $2p$ tel que $I(P) \neq I_p(P)$ (donc on ne peut pas faire mieux).

Aussi surprenant que cela puisse paraître au premier abord, la méthode utilise l'algèbre bilinéaire de L2. On munit l'espace vectoriel des polynômes du produit

[4]Le cas antérieur consiste à choisir $\omega = 1$ et $[a, b]$ compact.

scalaire :

$$< P, Q >= I(PQ) = \int_a^b P(t)Q(t)\omega(t)dt.$$

Le procédé de Gram-Schmidt appliqué à la base canonique permet de fabriquer l'unique famille $(P_n)_n$ de polynômes de sorte que la famille soit orthogonale et que pour tout k, P_k soit unitaire de degré k. On montre alors que les racines de P_k sont toutes réelles distinctes dans l'intervalle $]a, b[$. Le fait que pour tout $j \leq p-1$, $0 =< X^j, P_p >= I(X^j P_p) = I_p(X^j P_p)$ impose de choisir pour x_i les racines de P_p d'où l'unicité des x_i. En écrivant alors que pour tout $j \leq p-1$, $J_j = I(X^j)$ on obtient un système linéaire de van der Monde ayant une solution unique en les a_i. Par construction, on vérifie que l'unique formule possible convient. Le polynôme P_p^2 montre que $I = I_p$ n'est pas valable pour tous les polynômes de degré $2p$. Enfin, désignant par $(L_j)_j$ les polynômes de Lagrange associés aux (x_i), la formule appliquée à L_j^2 montre que $a_j = I(L_j^2) > 0$.

Pour un certain nombre d'exemples, les polynômes $(P_n)_n$ (éventuellement modifiés à une constante multiplicative près) sont connus et tabulés. Voici quelques exemples :

1. Lorsque $a = 0$, $b = 1$ et $\omega = 1$, on obtient les polynômes de Legendre.
2. Lorsque $a = 0$, $b = +\infty$ et $\omega = [t \mapsto e^{-t}]$, on obtient les polynômes de Laguerre.
3. Lorsque $a = -\infty$, $b = +\infty$ et $\omega = [t \mapsto e^{-t^2}]$, on obtient les polynômes d'Hermite.
4. Lorsque $a = -1$, $b = 1$ et $\omega = [t \mapsto 1/\sqrt{1-t^2}]$, on obtient les polynômes de Tchebitchev de seconde espèce.

On notera le fait remarquable que sauf dans le premier cas, on traite essentiellement des situations d'intégrales impropres. C'est d'ailleurs ce qui justifie le choix des exemples d'applications vus ci-dessous.

Cas de Gauss-Laguerre.

Ici on a donc

$$I(f) = \int_0^{+\infty} e^{-t} f(t)dt.$$

Les polynômes de Laguerre sont connus notamment par la formule suivante :

$$Lag_n(x) = e^x \frac{d^n}{dx^n}\left(e^{-x}x^n\right),$$

mais nous ignorerons cet aspect des choses. On va écrire un programme permettant de calculer, p étant fourni en entrée, les x_i et les a_i. L'exercice ci-dessous est décomposé afin de vous aider à écrire le programme. Enfin, il ne me reste plus qu'à

vous dire que dans ce cas $J_j = j!$ (je rappelle que $j!$ se calcule (vectoriellement !) par `gamma(j+1)` dans SCILAB).

Exercice 3.6.6 *Soit n un entier fixé.*
1. *Dans cette première étape, on cherche le polynôme P_p sous la forme :*

$$P_p = X^p + \sum_{j=0}^{p-1} c_j X^j$$

(bien entendu les c_j dépendent aussi de p). Pour tout $k \leq p-1$, X^k et P_p sont orthogonaux, cela fournit un système linéaire en les c_j. L'écrire puis en déduire le moyen, à partir de p, de récupérer les c_j, puis P_p (utilisez la commande SCILAB `poly`*).*
2. *En déduire comment récupérer les x_i (utilisez la commande SCILAB* `roots`*). Poser le système en les a_i puis écrire comment le résoudre dans SCILAB.*
3. *Terminer l'écriture de la procédure.*
4. *L'utiliser pour calculer $I_p(f)$ avec $f(t) = \sqrt{t}$. Comparer les valeurs obtenues à la valeur exacte ($\sqrt{\pi}$).*

Cas de Gauss-Tchebitchev.

Ici on a donc

$$I(f) = \int_{-1}^{1} \frac{f(t)}{\sqrt{1-t^2}} dt = \int_0^{\pi} f(\cos(\theta)) d\theta.$$

Les polynômes de Tchebitchev de seconde espèce sont connus notamment par le fait que T_n est l'unique polynôme (de degré n) tel que :

$$\forall \theta \in \mathbb{R}, \quad T_n(\cos(\theta)) = \cos(n\theta).$$

Par exemple, $T_2(X) = 2X^2 - 1$ puisque pour tout θ, $\cos(2\theta) = 2\cos^2(\theta) - 1$.

Les intégrales J_j se calculent, lorsque p est pair, par les intégrales de Wallis. On trouve :

$$J_j = \begin{cases} 0 & \text{si } j \text{ est impair} \\ \frac{\Gamma((j+1)/2)\sqrt{\pi}}{\Gamma(j/2+1)} & \text{si } j \text{ est pair} \end{cases}$$

Dans SCILAB vous pourrez calculer J_j par :
`deff("z=Jj(j)","z=(1-modulo(j,2)).*sqrt(%pi).*gamma((j+1)/2)./gamma(j/2+1)")`
qui fonctionne avec des vecteurs `j`.

Toute personne à l'aise avec la trigonométrie n'aura aucun mal à montrer que les T_n sont orthogonaux, qu'ils sont de norme $\sqrt{\pi/2}$ (pour $n \geq 1$) et que leur coefficient dominant est $2^{n-1}X^n$ (ainsi $P_n = T_n/2^{n-1}$). Enfin les racines de T_n sont les :

$$x_i = \cos\left(\frac{2i+1}{2n}\pi\right), \quad i = 1, \cdots, n.$$

On admettra ces points.

1. Le polynôme $T_n(X)$ se calcule dans SCILAB par `chepol(n,'x')`. A partir de là, reprendre les questions 2 et 3 de l'exercice précédent (en prenant directement les valeurs des x_i indiquées ci-dessus). Testez plusieurs petites valeurs de n. Que constatez-vous pour les a_i ? Peut-on prévoir leur valeur (déterminer leur somme).
2. En fait ce qui constaté dans la question 1 concernant les a_i est vrai. Ainsi dans le cas de Tchebitchev, on a une formule explicite :

$$I_n(f) = \frac{\pi}{n} \sum_{i=1}^{n} f\left(\cos\left(\frac{2i+1}{2n}\pi\right)\right).$$

3. Utilisez directement cette formule pour approcher

$$I = \int_{-1}^{1} \frac{\cos(t)}{\sqrt{1-t^2}} dt = \int_0^{\pi} \cos(\cos(\theta)) d\theta.$$

La convergence semble t-elle rapide ?

Estimation de l'erreur dans la méthode de Gauss.

On suppose que la fonction f est de classe C^{2p} sur l'intervalle d'intégration. Soit P le polynôme de degré $2p-1$ au plus vérifiant :

$$\forall i = 1, \ldots, p, \quad P(x_i) = f(x_i) \quad \text{et} \quad P'(x_i) = f'(x_i).$$

On rappelle la formule d'erreur suivante :

$$\forall x \in]a,b[, \ \ \exists \xi_x \in]a,b[, \ \ f(x) - P(x) = \frac{f^{(2p)}(\xi_x)}{(2p)!} \left[\prod_{i=1}^{p} (x - x_i)\right]^2 = \frac{f^{(2p)}(\xi_x)}{(2p)!} P_p^2(x).$$

En pratique ξ_x n'étant pas connu, on majore $|f^{(2p)}(\xi_x)|$ par $M_{2p} = \sup_{t \in]a,b[} |f^{(2p)}(t)|$. En notant que $I_p(P) = I_p(f)$ (car P et f ont les mêmes valeurs aux x_i) et $I_p(P) = I(P)$ car la formule est exacte pour P, on arrive finalement à la majoration :

$$|I(f) - I_p(f)| \leq \frac{M_{2p}}{(2p)!} \|P_p\|^2.$$

Exercice 3.6.7 *1. On a vu un avantage de la méthode de Gauss, son applicabilité à des intégrales impropres. Mais que pensez vous de celle-ci au vu de la formule d'erreur ? Est-elle intéressante pour l'exemple de la racine carrée dans le cas de Gauss-Laguerre ?*

2. Spécifier la formule d'erreur dans le cas de Gauss-Tchebichev, puis dans le cas particulier de $f = \cos$ (traité dans l'exercice précédent). Que pensez-vous de la convergence de la méthode dans ce cas ?

3.7 Résolution d'Equations Différentielles du premier ordre.

On suppose maintenant que l'on veut résoudre l'équation différentielle suivante

$$\left\{ \begin{array}{rcl} y'(x) & = & f(x, y(x)) \quad \text{pour} \quad x \in [0, T], \\ y(0) & = & y_0 \quad \text{avec} \quad y_0 \in \mathbb{R}, \end{array} \right. \tag{3.3}$$

où f est une fonction de classe C^1 sur $[0, T] \times \mathbb{R}$ avec $T > 0$. En général, on ne peut pas trouver la solution théorique à une telle équation différentielle (en revanche, on sait qu'il existe une solution unique à cette équation, au moins si T n'est pas trop grand). Aussi est-on amené à utiliser des méthodes numériques d'approximation de la solution. Nous voyons dans ici la méthode d'Euler.

3.7.1 La fonction `ode` de SCILAB.

SCILAB propose plusieurs routines de résolution numérique approchée, dont nous ne détaillerons pas les différences qui seraient complexes à exposer. Prenons l'exemple de la routine `ode`. La syntaxe de base est :
> `ode(y0,x0,x,F0)`
où `F0` est une fonction contenant l'équation (voir plus bas), `x` est le vecteur des instants ou sera calculée la solution, `y0` est la condition initiale en `x0`.

A titre de premier exemple, résolvons $y' = y^2$ avec la condition initiale $y(0) = 1$ sur $[0; 1/2]$. Vous pourrez comparer à la solution exacte qui est $x \mapsto \frac{1}{1-x}$. Première étape, on définit la fonction :
> `deff("yprime=fct(x,y)","yprime=y.^2")`
Vous noterez qu'il est indispensable de faire figurer la variable x, même si l'équation n'en dépend pas. On donne ensuite les paramètres :
> `y0=1 ; x0=0; x=0:0.05:0.5;`
enfin on calcule la solution :
> `ode(y0,x0,x,fct)`

3.7.2 Méthode d'Euler (explicite).

Nous allons maintenant programmer la méthode d'Euler, qui est la plus méthode la plus simple pour résoudre de manière approchée une équation différentielle.

Exercice 3.7.1 *On s'intéresse à l'équation particulière suivante :*

$$\left\{ \begin{array}{rcl} y'(x) & = & -2y(x) \quad \textit{pour} \quad x \in [0, T], \\ y(0) & = & y_0 \quad \textit{avec} \quad y_0 \in \mathbb{R}. \end{array} \right. \tag{3.4}$$

1. *Résoudre théoriquement cette équation.*
2. *En utilisant la commande* `ode`*, déterminer une approximation de la solution sur l'intervalle* $[0, 5]$ *en parcourant l'intervalle avec un pas de* $0,05$ *(on prendra comme condition initiale* $y_0 = 1$ *puis* $y_0 = -5$*).*
3. *Représenter la solution approchée de l'équation (3.4) pour une résolution de* 100*. Déterminer (empiriquement) la distance maximale entre les solutions exactes et approchées.*

Pour résoudre numériquement l'équation (3.3) par une méthode d'Euler (explicite), l'idée est d'abord de diviser l'intervalle $[0, T]$ en N sous-intervalles à partir de $0 = x_0 < x_1 < \cdots < x_N = T$. On cherche alors une approximation y_i) de $y(x_i)$ pour $0 \leq i \leq N-1$, ce qui constituera notre approximation de y sur $[0, T]$. En notant $h_i = x_{i+1} - x_i$, on utilise les formules d'approximation des dérivées, pour $i = 0, 1, \cdots, N-1$:

$$y'(x_i) \approx \frac{y(x_{i+1}) - y(x_i)}{h_i} \approx \frac{y_{i+1} - y_i}{h_i}.$$

En général, on prend $h_i = T/N$. L'équation permet alors de déterminer récursivement les h_i :

$$f(x_i, y(x_i)) = y'(x_i) \simeq \frac{y_{i+1} - y_i}{h_i},$$

d'où :

$$y_{i+1} = y_i + h_i f(x_i, y_i).$$

Une variante est la méthode implicite :

$$y_{i+1} = y_i + h_i f(x_i, y_{i+1}).$$

Cette dernière exige à chaque étape la résolution d'une équation, mais présente le mérite d'avoir de meilleures propriétés numériques.

Exercice 3.7.2 *(***Exploitation de la méthode d'Euler explicite***) Faire un programme qui trace la différence entre les solutions théorique et approchées de l'équation (3.4) pour les mêmes paramètres (*T*,* N *et* y_0*). En augmentant progressivement* N*, que remarquez-vous ?*

3.8 Résolution d'Equations Différentielles du second ordre.

3.8.1 Problèmes de Cauchy.

Nous allons prendre un problème modèle, la recherche de solutions de l'équation différentielle :

$$-u'' + u = f(x),$$

où f est une donnée, continue sur l'intervalle d'étude $I = [a, b]$ et u est l'inconnue. On découpe l'intervalle $[a, b]$ en $N + 1$ sous-intervalles réguliers, en posant :

$$x_i = a + i\frac{b-a}{N+1}, \quad i = 0, \ldots, N+1.$$

On note u_i les valeurs approchées que l'on prendra pour $u(x_i)$. Ici le pas est $h = \frac{b-a}{N+1}$, donc pour approximation de $-u'' + u$ on prend :

$$-\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} + u_i = \frac{1}{h^2}\left[-u_{i+1} + (2+h^2)u_i - u_{i-1}\right].$$

Si notre problème est un problème de Cauchy :

$$\begin{cases} -u'' + u = f(x), \\ u(a) = \alpha, \quad u'(a) = \beta \end{cases}$$

alors il est raisonnable de poser $u_0 = \alpha$ et $u_1 = u(a+h) \approx u(a) + u'(a)h = \alpha + \beta h$. Connaissant u_0 et u_1, la formule donnant l'équation approchée :

$$\frac{1}{h^2}\left[-u_{i+1} + (2+h^2)u_i - u_{i-1}\right] = f(x_i), \quad i = 1, \ldots, N$$

permet de calculer récursivement les termes u_i.

Exercice 3.8.1 *Appliquer la méthode vue pour résoudre le problème :*

$$\begin{cases} -y'' + y = 0 \\ y(0) = 1 \\ y'(0) = 0 \end{cases}$$

On comparera à la solution exacte, qui est $x \mapsto \frac{e^x + e^{-x}}{2}$ *(qui porte le nom de* cosinus hyperbolique*).*

3.8.2 Problèmes aux limites.

La méthode porte le nom de méthode des différences finies (en dimension 1). Considérons maintenant un problème aux limites :

$$\begin{cases} -u'' + u = f(x), \\ u(a) = \alpha, \quad u(b) = \beta \end{cases}$$

Dans ce cas, on remplace u_0 par α et u_{N+1} par β dans la série de relations :

$$\frac{1}{h^2}\left[-u_{i+1} + (2+h^2)u_i - u_{i-1}\right] = f(x_i), \quad i = 1, \ldots, N$$

ce qui donne le système :

$$A \begin{pmatrix} u_1 \\ \vdots \\ u_N \end{pmatrix} = \begin{pmatrix} f(x_1) + \frac{\alpha}{h^2} \\ f(x_2) \\ \vdots \\ f(x_{N-1}) \\ f(x_N) + \frac{\beta}{h^2} \end{pmatrix},$$

où la matrice A est tridiagonale, avec les termes $1 + \frac{2}{h^2}$ sur la diagonale, et $-1/h^2$ sur la sur et sous diagonale.

Exercice 3.8.2 *On considère le problème :*

$$\begin{cases} -u'' + u = x, \\ u(0) = 0, \quad u(1) = 0 \end{cases}$$

1. *Trouver la solution exacte.*
2. *Mettre en oeuvre la procédure de calcul approché décrite ci-dessus. Comparer avec la solution exacte.*

3.9 Etude des systèmes 2×2 autonomes.

Cette section sera rédigée ultérieurement.

3.10 Résolution d'EDP : méthode des différences finies.

Partie non rédigée cette année, voir cours oral.

Chapitre 4

Probabilités et Statistiques.

4.1 Introduction

Ce chapitre vise à mettre en avant le concours que peuvent apporter les méthodes numériques à l'illustration et à la résolution de problèmes faisant intervenir une composante aléatoire. Par cela, l'on entend généralement un aspect non entièrement *déterminé* — et il peut donc être utile de s'arrêter quelques instants sur la question du hasard et de sa nature. Quelles caractéristiques attend-on d'un événement ou d'un processus aléatoire ? Si A énonce une suite de nombres "au hasard", quelles propriétés de cette suite vont-elles convaincre B de son caractère *vraiment* aléatoire ? La genèse de la suite numérique en question est évidemment cruciale, et c'est précisément par ce point que nous commencerons ce chapitre. Sachant qu'un ordinateur ne fait jamais qu'exactement ce que nous lui demandons, est-il possible de mettre au point un algorithme déterminé — et donc déterministe — produisant une suite numérique aléatoire ou du moins *ressemblant* à de l'aléatoire, autrement dit *pseudo*-aléatoire ? Nous verrons plus loin des exemples de générateurs remplissant cet emploi.
Une question qui vient ensuite naturellement est celle de la possibilité de produire du hasard se pliant à une certaine forme. Il s'agit là d'être capable de *simuler* numériquement des distributions de probabilité non nécessairement uniformes, sur des ensembles discrets ou continus. Ainsi munis d'*éprouvettes numériques* nous permettant en quelque sorte de simuler expérimentalement les lois usuelles, nous explorerons certains résultats célèbres de la théorie mathématique des probabilités, tels que la loi des grands nombres et le théorème central limite.

4.2 Nombres pseudo-aléatoires.

4.2.1 Un système chaotique.

Creez le fichier `iterlogis.sci` :

```
function y=iterlogis(x,n);
```

```
y=x;
for i=2:n
x=4x.*(1-x);
y=[y;x];
end; endfunction
```

Ce fichier prend en entrée un vecteur ligne x et un entier n au moins égal à 2, et sort un vecteur de taille $n \times 2$ de sorte que chaque colonne soit les itérées par la fonction logistique $x \mapsto 4x(1-x)$ de l'élément de la même colonne présent en première ligne. A l'aide de ce programme, comparez les suites issues des nombres 0.3 et 0.301 (on regardera l'évolution de l'erreur relative moyenne en fonction du nombre de termes calculés, et on pourra tracer sur un même graphique les deux orbites).

On peut tracer un histogramme des valeurs prises par la fonction par les 3000 premiers termes issus de $x_0 = 0.3$ en 30 classes par :
`clf();X=iterlogis(0.3,3000);histplot(30,X)`
En changeant le nombre de points et le x_0, vous constaterez que sauf aux bornes, les valeurs prises semblent approximativement bien réparties (par exemple si l'on se restreint aux valeurs entre 0.3 et 0.7).

En fait, pour $a \in [0,4]$, la fonction $x \mapsto ax(1-x)$ est bien un système dynamique sur $[0,1]$, chaotique pour a assez voisin de 4. Ce qui est remarquable lorsque $a = 4$, c'est que l'on peut calculer x_t en fonction de x_0 :
Supposant que $x_t = \sin^2(\theta_t)$, trouver une relation de récurrence sur θ_t. En déduire qu'en posant $\theta_0 = \arcsin(\sqrt{x_0})$, on a :

$$x_t = \sin^2\left(2^t\theta_0\right).$$

De cette formule, avec quelques connaissances élémentaires sur les sous groupes de $\mathbb{R}$, on explique pourquoi lorsque x_0 et x'_0 sont proches, pour tout N, il existe des indices plus grands pour lesquels x_n et x'_n seront éloignés (ou proches).

4.2.2 Nombres pseudo-aléatoires.

Le problème que nous nous posons maintenant est : comment générer des nombres pris "au hasard" entre 0 et 1 ? En termes plus mathématiques, la question se pose sous la forme suivante : si $X : \Omega \to [0,1]$ est une variable aléatoire de distribution uniforme sur $[0,1]$, comment produire une réalisation de X, c'est-à-dire $X(\omega)$ avec $\omega \in \Omega$? (ce qu'on appellera nombre pseudo-aléatoire par la suite) Et si $(X_1, \ldots, X_n)$ sont n variables indépendantes de même loi que X, comment générer une réalisation de $(X_1, \ldots, X_n)$, soit $(X_1(\omega), \ldots, X_n(\omega))$? Ces questions se posent car un processeur fonctionne de façon intégralement déterministe, donc a priori sans possibilité de produire du "hasard". Pourtant, et c'est aussi ce

que permet la fonction logistique vue plus haut, certaines fonctions déterministes peuvent approcher le "hasard".

$modulo(27, 10)$ (fonction modulo) : on calcule 27 modulo 10, soit le reste de la division euclidienne de 27 par 10 : on obtient 7 car $27 = 2 \times 10 + 7$. Testez aussi $modulo(19, 7)$; faire suffisamment d'exemples jusqu'à avoir compris *modulo*.

Ouvrir un fichier `pseudo1.sci` et y créer une fonction `pseudo1` telle que : $y = pseudo1(x) = modulo(13x, 100)$. Retourner alors sur la fenêtre de commandes Scilab. Générer la suite (x_n) avec $n = 0, 1, \ldots, 15$ telle que $x_{n+1} = pseudo1(x_n)$ et $x_0 = 1$. Par ce biais, on a généré des nombres entiers qui semblent se comporter comme des chiffres entre 0 et 99 pris au "hasard" de façon équiprobable et indépendante.

Cependant, on s'aperçoit, si on génère (x_n) avec $n = 0, 1, \ldots, 25$ qu'un motif se répète de façon périodique. Lequel ? Quelle période ? Pourquoi ? Ceci met en défaut la prétention à générer ainsi des nombres "aléatoires". SILAB dispose de deux fonctions permettant de générer des nombres "aléatoires", `rand` et `grand`, ils sont basés notamment sur de telles suites congruentes, mais avec des nombres plus grands (voir l'aide de `grand`).

4.3 Les lois usuelles.

4.3.1 Lois discrètes.

On veut générer une réalisation d'une variable de Bernoulli X telle que $P(X = 1) = p$ et $P(X = 0) = 1 - p$, avec $p \in [0, 1]$. Que se passe-t-il si $p = 1$? Quelles sont les valeurs possibles pour X ? Quelle est la fonction de répartition de X, définie par $F_X(x) = P(X \leq x)$? Créer alors une fonction appelée `bernoulli.sci` :

```
function x=bernoulli(p);
x=0;
u=rand() ;// On génére une réalisation u d'une va uniforme sur [0,1]
if (u<=p) // Test sur u
x=1; // Dans le cas où u > p, on reste avec x = 0
end;
endfunction;
```

Essayons : `bernoulli(0.3)`. Faire d'autres essais. Comment faire pour vérifier empiriquement que l'on a bien généré une variable de Bernoulli ? Montrer théoriquement que c'est bien le cas (pour cela, il suffit de calculer F_X en posant $X = 1$ si $U \geq p$, et $X = 0$ sinon). Modifier votre fonction de façon à générer maintenant n réalisations indépendantes de X.

Passons maintenant à la génération d'une variable aléatoire discrète quelconque. On commence par un cas où la v.a. peut prendre trois valeurs (0, 1 et 2 pour fixer). Creer le fichier `VAdisc.sci` suivant, générant une réalisation d'une v.a. Y qui suit une loi discrète à valeurs dans $\{0,1,2\}$ étant données les probabilités d'avoir 0, 1 et 2 ($p0 = P(X=0)$, $p1 = P(X=1)$, $P(X=2) = 1-p0-p1$) :

```
function   [y]=VAdiscrete(p0,p1)
u=rand(); y=0;
if (u>=p0)  &  (u<p0+p1)
y=1;
elseif  (u>=p0+p1)
y=2;
end;
endfunction;
```

Faire des essais, puis transformer la fonction pour simuler k réalisations indépendantes de Y.

Exercice 4.3.1 *Suivant la procédure précédente, simuler k réalisations indépendantes d'une v.a. suivant une loi binomiale $\mathcal{B}(2, 0.3)$. N'y-aurait-il pas un autre moyen de simuler une telle variable (penser à une somme de v.a.) ? En déduire une fonction simulant k réalisations indépendantes d'une v.a. suivant une loi binomiale $\mathcal{B}(n,p)$, où n et p sont des paramètres.*

4.3.2 Lois uniformes.

Si X suit une loi uniforme sur $[a;b]$, alors $(X-a)/(b-a)$ suit une loi uniforme sur $[0;1]$. L'étude des lois uniformes se ramène donc à l'étude de la loi uniforme sur $[0;1]$. Pour celle-ci, tout est simple :

- la densité de la loi est $1_{[0;1]}$,
- sa fonction de répartition est (sous forme résumée) $x \mapsto \max\{0; \min\{x;1\}\}$.

De plus, on dispose dans SCILAB d'un générateur de matrices dont les coefficients suivent la loi uniforme, il s'agit de `rand`.

Vous aurez souvent à tracer des histogrammes d'observations issues de réalisation de lois. Vous disposez pour cela de la commande `histplot`. `histplot(n,x)` trace un histogramme en regroupant en `n` classes égales les valeurs de `x`.

Exercice 4.3.2 *Simulez plusieurs réalisations de la loi uniforme sur $[0;1]$. Tracez des histogrammes, et comparez les moyennes et variances empiriques aux valeurs exactes, qui sont respectivement $1/2$ et $1/12$.*

4.3.3 Lois normales.

Si X suit une loi normale $N(m, \sigma^2)$, alors $(X - m)/\sigma$ suit une loi $N(0; 1)$: on se ramène donc à la loi $N(0; 1)$. Cette fois, la situation est un peu moins agréable, puisque, si la densité est la fonction $x \mapsto \frac{1}{\sqrt{2\pi}} \exp(-x^2/2)$, sa fonction de répartition ne s'exprime pas à l'aide des fonctions usuelles, c'est :

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp(-u^2/2) du.$$

SCILAB dispose de la fonction d'erreur, notée `erf`, qui est définie par :

$$\forall x \in \mathbb{R}, \quad \texttt{erf}(x) = \frac{2}{\sqrt{\pi}} \int_{0}^{x} \exp(-t^2) dt$$

qui est donc une fonction régulière, impaire, strictement croissante et tendant vers 1 à l'infini. SCILAB dispose aussi de la fonction `erfinv` qui est la bijection réciproque de `erf`. Un calcul élémentaire montre que Φ peut s'exprimer à l'aide de `erf` et que plus précisément :

$$\Phi(x) = \frac{1}{2}(1 + \texttt{erf}(x/\sqrt{2})).$$

Rappelons que l'on peut simuler une matrice dont les coefficients suivent des lois normales centrées réduites par l'instruction `rand` avec l'option `"normal"`.

Une commande particulièrement pratique est la commande `cdfnor`. Il en existe d'ailleurs des équivalentes pour d'autres lois que l'on peut voir apparaître listées grâce à `apropos cdf`. Etant donnée une v.a. Y suivant une loi normale de moyenne `Mean` et d'écart-type `std`, on note pour chaque `X`, `P` la proba que $Y \leq X$ et $Q = 1 - P$. La connaissance de trois des quatre variables parmi $Mean$, Std, X, (P, Q) permet la connaissance de la dernière, par :

- `[P,Q]=cdfnor("PQ",X,Mean,Std)` (`cdfnor("PQ",X,Mean,Std)` donne `P`).
- `[X]=cdfnor("X",Mean,Std,P,Q)`
- `[Mean]=cdfnor("Mean",Std,P,Q,X)`
- `[Std]=cdfnor("Std",P,Q,X,Mean)`

Ainsi, l'utilisation de `erf` est en fait inutile.

Exercice 4.3.3 *On suppose que les notes des étudiants à un partiel sont réparties suivant une loi normale $N(11; 2)$.*
1. *Quelle est le pourcentage d'étudiants dépassant la note de 14 ?*
2. *Quelle est la note pour laquelle on a 60% de l'effectif au dessus ?*
3. *Simulez un échantillon de 10000 valeurs de la loi normale $N(11; 2)$ et comparez avec les valeurs obtenues ci-dessus.*

4.3.4 Simulation d'autres lois usuelles.

On dispose tout d'abord d'un générateur généralisant `rand`, il s'agit de `grand` (cf. aide). Pour des raisons purement pédagogiques, je vais plutôt illustrer la méthode de la fonction de répartition. Supposons que l'on veuille simuler d'autres lois dont on connaisse la fonction de répartition F et explicitement son inverse. Alors, si X suit une loi uniforme sur $[0;1]$, la loi de $F^{-1}(X)$ est la loi de fonction de répartition F (F^{-1} désigne la bijection réciproque de F et non $1/F$).

Exercice 4.3.4 *La loi exponentielle de paramètre λ a pour fonction de répartition $F_\lambda(x) = (1-e^{-\lambda x})1_{\mathbb{R}^+}(x)$. Simulez pour plusieurs valeurs de λ plusieurs échantillons suivant la loi exponentielle de paramètre λ. Même question avec la loi de Cauchy dont la densité sur $\mathbb{R}$ est : $x \mapsto \frac{1}{\pi(1+x^2)}$.*

4.4 Théorèmes limites en Calcul des Probabilités.

Il faut voir ces deux résultats comme étant complémentaires. Heuristiquement, la loi des grands nombres exprime le fait que si l'on répète un grand nombre de fois la même expérience, on s'attend à ce que le comportement moyen se rapproche du comportement théorique : si l'on jète un million de fois une pièce, il devrait y avoir grosso-modo autant de piles que de faces. Ceci n'est bien entendu pas vrai si l'on ne fait que peu de lancers. C'est donc un résultat de convergence. Le théorème central limite précise d'une certaine manière la vitesse de convergence : en multipliant par $\sqrt{n}$ le terme tendant vers 0 dans la loi des grands nombres, on observe quelque chose d'intéressant.

4.4.1 Loi des Grands Nombres.

On rappelle l'énoncé de ce théorème (appelé "Loi" car on a longtemps cru que c'était un résultat de physique et non de mathématiques). On considère des variables aléatoires $(X_n)_{n\in\mathbb{N}^*}$, indépendantes et de même loi qu'une variable X. Si on suppose que $\mathbb{E}|X| < \infty$, alors :

$$\bar{X}_n = \frac{1}{n}(X_1 + X_2 + \cdots + X_n) \to m \quad \text{avec } m = \mathbb{E}X.$$

Intuitivement, cela signifie que la moyenne empirique converge[1] vers la moyenne théorique quand on considère un nombre croissant de variables aléatoires.

Exercice 4.4.1 *Faire un programme calculant la moyenne d'un nombre n (arbitraire) de réalisations de variables uniformes sur $[-2,1]$, et qui trace en fonction de n cette moyenne. Que constate-t-on? Faites la même chose avec des réalisation indépendantes d'une v.a. distribuée d'abord selon une loi normale $\mathcal{N}(2,3^2)$, puis*

[1] en un sens qui sera précisé dans le cours de probabilités du S6.

suivant une loi de Cauchy (que l'on génère à partir de la tangente d'une variable uniforme sur $]-\pi/2, \pi/2[$*). Conclusions ?*

4.4.2 Application de la Loi des Grands Nombres : distribution d'une variable aléatoire.

Une application intéressante de la Loi des Grands Nombres est le fait que lorsque l'on a de nombreuses réalisations indépendantes d'une variable aléatoire, alors l'histogramme, s'il possède suffisamment de classes, se rapproche de la loi (densité) de probabilité. Cela s'explique par le fait que la proportion empirique de variables dans $[a, b]$ tend vers la probabilité théorique que la variable appartienne à $[a, b]$ (cette probabilité est en fait l'espérance de la fonction qui vaut 1 si la variable est dans $[a, b]$ et 0 sinon).

Répétez plusieurs fois cette ligne, en changeant éventuellement le nombre de v.a. simulées ou le nombre de classes de l'histogramme :
`X=rand(100,1); xbasc(); hist(10,X)`
Commentez le résultat obtenu. Reprendre la même question avec une loi normale puis une loi de Cauchy. Expliquer pourquoi cela "marche" pour Cauchy alors que $\mathbb{E}|X| = \infty$.

4.4.3 Théorème Central Limite.

Rappelons d'abord un énoncé de ce théorème, qui est en fait une sorte de raffinement par rapport à la Loi des Grands Nombres : on considère des variables aléatoires $(X_n)_{n\in\mathbb{N}^*}$, indépendantes et de même loi qu'une variable X. Si on suppose que $\mathbb{E}|X| < \infty$, et que $\mathbb{E}|X|^2 < \infty$, alors :

$$\sqrt{n}\left(\frac{\bar{X}_n - m}{\sigma}\right) = \sqrt{n}\left(\frac{\frac{1}{n}(X_1 + X_2 + \cdots + X_n) - m}{\sigma}\right) \to \mathcal{N}(0, 1),$$

avec $m = \mathbb{E}X$ et $\sigma^2 = V(X)$. Cela signifie, de façon intuitive, que quand n est grand alors la moyenne empirique converge[2] vers la moyenne théorique suivant une répartition gaussienne $\mathcal{N}(0, \frac{\sigma^2}{n})$.

On recommence la situation d'un jeu de pile ou face. Cela correspond aussi à la situation d'un sondage d'opinion pour choisir entre A et B. Ainsi, si la proportion d'intention de vote pour A sur l'ensemble de la population est p, on peut modéliser chaque votant par X_i, et lorsque $X_i = 1$ se traduit par un vote pour A. Alors la somme des X_i pour $i = 1$ à n représente le nombre total d'intentions de vote

[2]en un sens qui sera précisé dans le cours de probabilités du S6.

pour A parmi n personnes et la moyenne empirique $\bar{X}_n$ représente la fréquence des intentions de vote pour A parmi les n personnes. On considère n fois 100 personnes sondées. La probabilité de voter pour le candidat A est p. La ligne suivante trace l'histogramme de X_n :

```
n=100;p=0.3;X=(rand(n,100)<p)+0; xbasc();histplot(10,mean(X,'c'))
```

Faites plusieurs essais avec ces valeurs, puis augmentez n. Que constatez-vous ? Mêmes questions en changeant p. Refaire des histogrammes (comme ci-dessus) sur la moyenne empirique dans le cas d'une loi exponentielle de paramètre 2, puis une loi de Cauchy. Conclusions ?

4.5 Estimation ponctuelle et ensembliste.

Cette section illustre les résultats d'estimations ponctuelles et ensemblistes.

On rappelle que la moyenne d'une loi exponentielle est $1/\lambda$ et sa variance est $1/\lambda^2$. Désignant par $\bar{X}_n$ la moyenne empirique dans le modèle d'échantillonage, ceci implique que $\sqrt{n}(\bar{X}_n - 1/\lambda)$ tend en loi vers une loi normale $N(0, 1/\lambda^2)$. Posons $Y_n = 1/\bar{X}_n$. L'application du théorème Δ assure alors que $\sqrt{n}(Y_n - \lambda)$ tend en loi vers une loi normale $N(0, \lambda^2)$.

Exercice 4.5.1 *Simulez n réalisations de loi exponentielle de paramètre λ connu ($n \geq 1000$). Calculez les estimateurs $\bar{X}_n$ et Y_n de $1/\lambda$ et de λ. Vous semblent-ils biaisés ?*

D'après ce que nous avons dit avant, $\Pi_n = \sqrt{n}(Y_n - \lambda)/Y_n$ tend en loi vers une $N(0; 1)$. C'est donc une fonction pivotale asymptotique pour λ. Etant donné un seuil d'erreur α, on sait que :

$$\lim_{n \to +\infty} P\left(|\Pi_n| \leq u_{1-\alpha/2}\right) = 1 - \alpha.$$

En général, on s'affranchit du passage à la limite. Prenons $\alpha = 5\%$, auquel cas $u_{1-\alpha/2} = 1.96$. On a alors, avec une probabilité d'erreur 5% :

$$Y_n \left(1 - \frac{1.96}{\sqrt{n}}\right) \leq \lambda \leq Y_n \left(1 + \frac{1.96}{\sqrt{n}}\right).$$

Exercice 4.5.2 *Simulez des lois exponentielles de paramètre connu, et donnez des intervalles de confiance. La vraie valeur fait-elle partie de l'intervalle de confiance obtenu ?*

4.6 Simulation.

Dans ces exercices, on se propose de répondre à des questions de calcul d'espérances via des simulations.

Exercice 4.6.1 *On tire de manière iid des* $(X_i)_i$ *qui valent* 1 *avec la probabilité* $2/3$ *et* -1 *avec la probabilité* $1/3$. *On note* $T = \inf\{i, X_i = -1\}$ *(qui est donc aléatoire) et*

$$S_T = \sum_{i=1}^{T} X_i.$$

1. *Calculer l'espérance de* S_T. *(On notera que l'on peut répondre exactement puisque* $S_T = T - 2$ *et que* T *suit une loi géométrique).*

2. *On note*

$$S = \sum_{i=1}^{50} X_i.$$

Calculer la probabilité pour que $S \in [-5, 5]$. *(On notera que l'on peut répondre exactement ou asymptotiquement puisque* S *est la transformée affine d'une binômiale, ou en considérant que* 50 *est grand, en appliquant le TCL aux* X_i*).*

Exercice 4.6.2 *On tire de manière i.i.d. des couples* $(X_k, Y_k)_k$ *où* X_k *suit une loi uniforme sur* $[1/2; 3/2]$ *et* Y_k *suit une loi uniforme sur* $[-\pi/2, \pi/2]$. *Notons*

$$Z_k = X_k \exp(iY_k) = (X_k \cos(Y_k), X_k \sin(Y_k)).$$

On note $S_n = \sum_{k=1}^{n} Z_k$ *et* N *le plus petit indice* n *de sorte que la première coordonnée de* S_n *soit au moins égale à* 10. *Calculer* $\mathbb{E}(S_N)$.

4.7 La méthode de Monte-Carlo et ses applications.

Nous présentons ici la méthode de Monte-Carlo de calcul d'espérances, et nous donnons des illustrations de ses applications aux calculs approchés d'intégrales et à la résolution d'une EDP issue de la finance.

Le principe de la méthode de base est simple : on est amené à estimer une espérance $\mathbb{E}_P(\phi(X))$ où la loi P a une densité f, dans le but de calculer l'intégrale :

$$\int \phi(x) f(x) dx = \int \phi(x) dP(x) = \mathbb{E}_P(\phi(X)).$$

On simule n réalisations i.i.d. de notre v.a., $X_1, \ldots, X_n$ et notant $Y_i = \phi(X_i)$ on remplace l'intégrale (qui vaut $\mathbb{E}_P(Y)$) par le moment empirique :

$$\bar{Y_n} = \frac{1}{n} \sum_{i=1}^{n} Y_i.$$

La loi des grands nombres nous apprend la convergence de la moyenne empirique vers la moyenne théorique. En dimension 1 pour nous, le théorème central-limite nous indique la vitesse de convergence :

$$\frac{\sqrt{n}}{\sqrt{V(Y)}}\left(\frac{1}{n}\sum_{i=1}^{n}\phi(X_i)-\mathbb{E}_P(Y)\right)\to N(0,1).$$

Il est donc possible de construire des intervalles de confiance, puisque si l'on désigne par :

$$S_n=\frac{1}{n-1}\sum_{i=1}^{n}(Y_i-\bar{Y_n})^2$$

un estimateur de la variance, la fonction :

$$\Pi_n=\sqrt{n}S_n^{-1/2}\left(\frac{1}{n}\sum_{i=1}^{n}\phi(X_i)-\mathbb{E}_P(\phi(X))\right)$$

est pivotale asymptotique de loi limite $N(0,1)$.

4.7.1 Application aux calculs d'intégrales.

Sous la forme la plus simple, lorsqu'on calcule des intégrales sur des intervalles compacts de fonctions sympathiques, on peut considérer que la loi de P est la loi uniforme sur le compact considéré.

Exercice 4.7.1 *En simulant n v.a. uniformes sur* $[0;1]$ *(augmenter n en partant de 1000), donner plusieurs valeurs approchées de l'intégrale :*

$$\int_0^1\sqrt{1-x^2}dx.$$

1. *On comparera les valeurs obtenues à la valeur exacte qui est* $\pi/4$ *en calculant l'erreur relative.*
2. *Construire des intervalles de confiance à 95%. La vraie-valeur en fait-elle partie ?*

On prend maintenant le problème d'une intégrale multiple (ici double). On simule donc à chaque étape un couple de v.a. uniformes.

Exercice 4.7.2 *Calculer de manière analogue une valeur approchée de l'intégrale double :*

$$\int_0^1\int_0^1 x^y dxdy$$

et comparez avec la valeur exacte ($\ln(2)$*).*

On n'est pas obligé de considérer des lois uniformes. Par exemple, si l'on veut intégrer sur un intervalle non borné, il est plus judicieux de choisir d'autres lois :

Exercice 4.7.3 1. *La loi exponentielle de paramètre λ a pour densité :*

$$\lambda \exp(-\lambda x) 1_{\mathbb{R}^+}(x).$$

Utilisez cette remarque pour calculer pour plusieurs valeurs de λ :

$$\int_0^{+\infty} e^{-\lambda x} \sqrt{x} dx$$

et comparez à la valeur exacte $\sqrt{\pi}/(2\lambda^{3/2})$.
2. *En utilisant des lois normales, calculez une valeur approchée de l'intégrale :*

$$\int_{-\infty}^{+\infty} e^{-x^2} \cos(x) dx.$$

4.7.2 Application aux EDP issues de la finance.

La formule de Feynman-Kac permet d'exprimer dans certains cas les solutions d'une EDP sous forme d'une espérance conditionnelle. Considérons par exemple le problème parabolique issu du modèle de Samuelson (Rational Theory of warrant prices, *Indust. Manag. Rev.*, 6, pp.13-31, 1965) :

$$\begin{cases} \frac{\partial v}{\partial t}(t,x) + \frac{\sigma^2}{2} \frac{\partial^2 v}{\partial x^2}(t,x) = 0 \text{ sur } [0;T[\times\mathbb{R} \\ v(T,x) = \phi(x) \text{ sur } \mathbb{R}. \end{cases}$$

La solution est donnée par :

$$v(t,x) = \mathbb{E}\phi(x + \sigma(W_T - W_t)),$$

où $(W_t)_t$ est un Brownien standart. Prenons le cas où $T = 1$ et $\phi(x) = (x - 0.5)^+$. On est donc intéressé par la fonction $x \mapsto v(0,x)$. Remarquant que $W_1 - W_0$ suit une loi normale $N(0,1)$, on calcule une valeur approchée de $v(0,x)$ en simulant n v.a. X_i suivant $N(0,1)$ et l'on calcule la moyenne des $\phi(x + \sigma X_i)$.

Exercice 4.7.4 *Prenons $\sigma = 1$. Tracer les fonctions $x \mapsto v(0,x)$ et ϕ sur un même graphique.*

Chapitre 5

Sujets posés antérieurement.

5.1 Interrogation de décembre 2005.

On s'intéresse à donner une formule d'approximation de $I(f) = \int_{-1}^{1} f(t)dt$ de la forme :

$$I(f) \approx I_5(f) = c_{-1}f(-1) + c_{-1/2}f(-1/2) + c_0 f(0) + c_{1/2}f(1/2) + c_1 f(1).$$

Pour cela, on appelle P_f le polynôme d'interpolation de f aux points $-1, -1/2, 0, 1/2, 1$ et l'on pose par définition $I_5(f) = I(P_f)$ (on rappelle que le degré de P_f est au plus égal à 4).

1. Montrer que la méthode est d'ordre 4 au moins.

2.
2.a. Calculer $I(1), I(X), I(X^2), I(X^3), I(X^4)$. En déduire que les coefficients c_i sont solution du système linéaire suivant :

$$\begin{cases} c_{-1} + c_{-1/2} + c_0 + c_{1/2} + c_1 = 2 \\ -c_{-1} - \frac{1}{2}c_{-1/2} + \frac{1}{2}c_{1/2} + c_1 = 0 \\ c_{-1} + \frac{1}{4}c_{-1/2} + \frac{1}{4}c_{1/2} + c_1 = \frac{2}{3} \\ -c_{-1} - \frac{1}{8}c_{-1/2} + \frac{1}{8}c_{1/2} + c_1 = 0 \\ c_{-1} + \frac{1}{16}c_{-1/2} + \frac{1}{16}c_{1/2} + c_1 = \frac{2}{5} \end{cases}$$

2.b. Quelle commande SCILAB proposez-vous pour résoudre ce système ?
2.c. La réponse SCILAB est $[0.1555556; 0.7111111; 0.2666667; 0.7111111; 0.1555556]$. Que constatez-vous pour les c_{-i} par rapport aux c_i ? Démontrer ceci (on pourra écrire un système linéaire dont est solution $(c_1 - c_{-1}, c_{1/2} - c_{-1/2})$).
2.d. (cette question n'est pas nécessaire pour la suite). Sachant que $0,111\cdots = \frac{1}{9}$, quelles sont, au vu des résultats de SCILAB, les valeurs prévisibles pour les coefficients ?

3. Au vu du résultat à la question **2.c**, la formule d'approximation prend la forme :

$$I(f) \approx I_5(f) = c_0 f(0) + c_{1/2}(f(1/2) + f(-1/2)) + c_1(f(1) + f(-1)).$$

3.a. Montrer que la méthode est d'ordre 5 (au moins).
3.b. On suppose désormais f de classe C^6 sur $[0,1]$ et l'on pose $M_6 = \sup |f^{(6)}|$. On considère le polynôme Q de degré minimal tel que $Q(x) = f(x)$ pour $x \in \{-1, -1/2, 0, 1/2, 1\}$ et $Q'(0) = f'(0)$. On rappelle que pour tout t :

$$|f(t) - Q(t)| \leq \frac{M_6}{6!} \left| t \prod_{j=-2}^{2} (t - j/2) \right|.$$

Montrer alors que l'erreur prend la forme :

$$|I(f) - I_5(f)| \leq C M_6,$$

où C est une constante que l'on exprimera à l'aide d'une intégrale dans laquelle le produit aura l'expression la plus simple possible (sans $\prod$ et en groupant les termes opposés).
3.c. Donner un majorant simple de C. Comment feriez-vous pour calculer C à l'aide de SCILAB ?

5.2 Partiel janvier 2006.

Exercice 1. Soit α un nombre réel strictement supérieur à 1. L'équation :

$$x^2 + 2\alpha x + 1 = 0$$

a deux racines réelles x_1 et x_2 distinctes (dépendant de α) telles que $x_1 + x_2 = -2\alpha$ et $x_1 x_2 = 1$.
1. On rappelle que pour ε petit, on a : $\sqrt{1+\varepsilon} \approx 1 + \frac{\varepsilon}{2}$. D'où vient cette approximation (on ne demande pas de la démontrer, juste de donner l'argument crucial) ?
2. Donner les expressions de x_1 et x_2, en prenant pour x_1 la plus petite des deux racines (on aura $x_1 < x_2 < 0$).
3. On suppose que α est très grand. Donner une valeur approchée de x_1 et de x_2. Laquelle des deux racines est la plus facile à calculer dans un logiciel à précision limitée en utilisant les formules du **2** ? (on pourra tenter de prévoir ce que dirait un logiciel travaillant avec 3 chiffres significatifs lorsque $\alpha = 10^{10}$, si l'on fait le calcul direct).
4. Ayant déterminé la racine la plus simple, comment feriez-vous pour calculer la seconde ? Proposez une fonction SCILAB prenant α en entrée (supposé grand) et retournant les deux racines.

Exercice 2. Soit $I = [0; 1]$. A tout entier $n \geq 2$ et $k \in \{0, \ldots, n-1\}$, on associe l'ensemble :

$$A_k^n = \left[\frac{k}{n}; \frac{k+1}{n}\right[.$$

On note $\chi_{n,k}$ la fonction définie sur I telle que $\chi_{n,k}(x) = 1$ lorsque $x \in A_k^n$ et 0 sinon. Ainsi, pour tout n, la fonction $\sum_{k=0}^{n-1} \chi_{n,k}$ est la fonction qui vaut 1 sur $[0; 1[$ et 0 en 1. On rappelle que pour f continue sur un compact K de $\mathbb{R}^p$ à valeurs réelles, le module de continuité ω_f défini par :

$$\omega_f(\delta) = \sup\{|f(x) - f(y)|, \quad (x, y) \in K^2, \quad \|x - y\| \leq \delta\}$$

tend vers 0 lorsque δ tend vers 0, et que $|f(x) - f(y)| \leq \omega_f(\|x - y\|)$.

1. Soit f continue sur le compact I (ici $p = 1$). On approche f par la fonction :

$$P_n = \sum_{k=0}^{n-1} f(k/n)\chi_{n,k}$$

(c'est-à-dire que P_n vaut $f(k/n)$ sur A_k^n).
1.a. Montrer que :

$$\forall t \in I, \quad |f(t) - P_n(t)| \leq \omega_f(1/n).$$

1.b. On approche $\int_0^1 f(t)dt$ par $\int_0^1 P_n(t)dt$. Ecrire la formule d'approximation obtenue. Reconnaissez-vous la méthode aboutissant à cette formule ? Retrouver à l'aide de **1.a** que lorsque n tend vers l'infini, l'approximation tend vers la valeur exacte. Prenant $f = \sqrt{\ }$ (pour lequel $\omega_f(\delta) = \sqrt{\delta}$), programmer dans SCILAB une fonction prenant la précision souhaitée ε en entrée et donnant une valeur approchée de l'intégrale à ε près en sortie.

2. Soit g continue sur le compact $I^2 = I \times I$ (ici, $p = 2$ et on munit I^2 de la distance $d((x_1, y_1), (x_2, y_2)) = |x_1 - x_2| + |y_1 - y_2|$). On approche g par la fonction :

$$Q_n : \left[(x, y) \mapsto \sum_{0 \leq k,l \leq n-1} f(k/n, l/n)\chi_{n,k}(x)\chi_{n,l}(y)\right]$$

(c'est-à-dire que Q_n vaut $g(k/n, l/n)$ sur $A_k^n \times A_l^n$).
2.a. Montrer que :

$$\forall (x, y) \in I^2, \quad |g(x, y) - Q_n(x, y)| \leq \omega_g(2/n).$$

2.b. On approche $\int_0^1 (\int_0^1 g(x, y)dx)dy$ par $\int_0^1 (\int_0^1 Q_n(x, y)dx)dy$. Ecrire la formule d'approximation obtenue. Retrouver à l'aide de **2.a** que lorsque n tend vers l'infini, l'approximation tend vers la valeur exacte.
2.c. Programmer la méthode dans SCILAB.

Exercice 3. Dans le problème de la sensibilité aux erreurs d'arrondis pour le calcul des valeurs propres, on peut montrer que le critère important est la borne inférieure des conditionnements des matrices de passage de la base canonique à une base de vecteurs propres. On se donne une matrice carrée A de taille n dont on cherche à déterminer les valeurs propres. On note $\mathcal{P}$ l'ensemble des matrices de passages de la base canonique à une base de vecteurs propres de A. On cherche donc à estimer $\inf\{\text{cond}(P), P \in \mathcal{P}\}$.
1. Soit P un élément de $\mathcal{P}$. On note $C_1, \ldots, C_n$ les colonnes de P. Montrer que pour tout $n-$uplet $(\alpha_1, \ldots, \alpha_n) \in (\mathbb{R}^*)^n$, la matrice dont les colonnes sont $\alpha_1 C_1, \ldots, \alpha_n C_n$ est aussi un élément de P (indication : quelle est l'interprétation des colonnes de la matrice de passage ?).
2. Prenons par exemple $n = 2$. On a aboutit à la matrice

$$P = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}.$$

Déterminer le conditionnement de la matrice P relativement à la norme 1. Déterminer parmi les matrices :

$$P_{\alpha,\beta} = \begin{pmatrix} 2\alpha & \beta \\ \alpha & 3\beta \end{pmatrix}$$

(issues de la question 1) celle(s) de conditionnement minimal ; on se limitera à α et β strictement positifs.

5.3 Partiel septembre 2006.

Exercice 1. Soit α un nombre réel strictement supérieur à 1. L'équation :

$$x^2 + 2\alpha x + 1 = 0$$

a deux racines réelles x_1 et x_2 distinctes (dépendant de α) telles que $x_1 + x_2 = -2\alpha$ et $x_1 x_2 = 1$.
1. On rappelle que pour ε petit, on a : $\sqrt{1+\varepsilon} \approx 1 + \frac{\varepsilon}{2}$. D'où vient cette approximation (on ne demande pas de la démontrer, juste de donner l'argument crucial) ?
2. Donner les expressions de x_1 et x_2, en prenant pour x_1 la plus petite des deux racines (on aura $x_1 < x_2 < 0$).
3. On suppose que α est très grand. Donner une valeur approchée de x_1 et de x_2. Laquelle des deux racines est la plus facile à calculer dans un logiciel à précision limitée en utilisant les formules du **2** ?
4. Ayant déterminé la racine la plus simple, comment feriez-vous pour calculer la seconde ?

Exercice 2. Le but est de calculer numériquement la constante d'Euler γ, définie par :

$$\gamma = \lim_{n \to +\infty} \left(\left(\sum_{k=1}^{n} \frac{1}{k} \right) - \ln(n) \right).$$

Les deux premières questions sont indépendantes, et préliminaires à la suite de l'exercice.

1. Montrer que pour tout $x \in \mathbb{R}^+$, on a :

$$(\mathbf{1}) \quad \frac{x^2}{2} - \frac{x^3}{3} \leq x - \ln(1+x) \leq \frac{x^2}{2}$$

et que pour $x \in [0; \frac{3}{2}]$, on a :

$$(\mathbf{2}) \quad 0 \leq x - \ln(1+x) \leq \frac{x^2}{2}.$$

2. Soit $\alpha > 1$. Montrer que :

$$\sum_{k=n+1}^{+\infty} \frac{1}{k^\alpha} \leq \frac{1}{(\alpha-1)n^{\alpha-1}}.$$

3. On introduit pour tout $k \geq 1$:

$$u_k = \frac{1}{k} - \ln\left(\frac{k+1}{k}\right).$$

3.a. Montrer que $\lim_{N\to+\infty} \sum_{k=1}^{N} u_k = \gamma$.
3.b. A l'aide de **1** (relation **(2)**) et de **2.**, donner un encadrement de :

$$\sum_{k=N+1}^{+\infty} u_k.$$

En déduire, $\varepsilon > 0$ étant donné, comment choisir $N(\varepsilon) \in \mathbb{N}^*$ de sorte que :

$$0 \leq \sum_{k=N+1}^{+\infty} u_k \leq \varepsilon.$$

3.c. En déduire, ε étant donné, comment fournir un encadrement d'amplitude ε de γ. La convergence de la méthode vous semble t-elle rapide ?
3.d. Programmer la méthode dans SCILAB.

4. On rappelle que $\sum_{k=1}^{+\infty} \frac{1}{k^2} = \pi^2/6$. On introduit la suite :

$$v_k = u_k - \frac{1}{2k^2}.$$

4.a. Déterminer la limite de $\lim_{N\to+\infty} \sum_{k=1}^{N} v_k$.
4.b. En s'inspirant de **3.b-c.**, montrer comment obtenir un encadrement d'amplitude ε de γ à l'aide de ce qui précède (on fera les adaptations nécessaires, en utilisant notamment **(1)** et non **(2)**, et l'on prendra garde au fait que $v_k \leq 0$). Comparer la convergence des deux méthodes.

5.4 Interrogation de novembre 2006.

Le barème est fait de sorte que l'on dépassera 20 en faisant raisonnablement deux exercices. Il est donc conseillé de lire l'énoncé en entier pour choisir vos deux exercices, et pour repérer l'articulation entre les questions.

Exercice 1. On souhaite déterminer une valeur approchée de la somme :

$$S = \sum_{k=1}^{+\infty} \frac{1}{k^6}.$$

Pour cela, on introduit, pour $N \geq 1$ les sommes partielles :

$$S_N = \sum_{k=1}^{N} \frac{1}{k^6},$$

et le reste de la série : $R_N = S - S_N$.
Nous considérons aussi la série :

$$S' = \sum_{k=1}^{+\infty} \frac{(-1)^k}{k^6}.$$

avec, pour $N \geq 1$ ses sommes partielles :

$$S'_N = \sum_{k=1}^{N} \frac{(-1)^k}{k^6},$$

et les restes : $R'_N = S' - S'_N$.

1.
1.a. Déterminer, pour toute valeur de k :

$$\frac{1}{k^6} + \frac{(-1)^k}{k^6}$$

(on distinguera selon que k est pair ou impair), et en déduire la relation :

$$S + S' = \sum_{p=1}^{+\infty} \frac{2}{(2p)^6}.$$

1.b. En déduire que :

$$S = -\frac{32}{31} S'.$$

Pour calculer une valeur approchée de S, on peut donc aussi déterminer une valeur approchée de S'. Nous allons maintenant voir, entre les deux, la méthode

qui semble la plus raisonnable. Pour la suite, il est suggéré de chercher des arguments simples. Presque tout se déduit du cours ou du TD.

2.

2.a. Montrer que $R_N \geq 0$ (suggestion possible : quelle est la monotonie de S_N ?), et que :

$$R_N \leq \frac{1}{6N^5}.$$

2.b. Justifier que

$$|R'_N| \leq \frac{1}{N^6}.$$

2.c. Au vu des résultats précédents, quelle série semble converger la plus vite ?

3. L'une des méthodes présente l'avantage que deux sommes partielles consécutives encadrent la somme totale. Préciser laquelle des deux, et expliquer si cela est pour vous un avantage ou un inconvénient. Au vu de la réponse à **2.c** et à **3**, laquelle des deux séries calculeriez-vous ? Justifier.

Exercice 2. On s'intéresse à résoudre $f(x) = 0$ sur un intervalle $I = [a, b]$ dans lequel on suppose que l'équation admet une unique solution.

1. Rappeler les avantages et inconvénients des méthodes suivantes : dichotomie, sécante et Newton. On s'intéressera notamment à leur convergence éventuelle, leur vitesse de convergence, les hypothèses qu'elles exigent sur f.

2. Donner un exemple où l'application de la méthode de Newton aboutit à la suite $((-1)^n)_n$.

3. On veut calculer $\sqrt{3}$ par la méthode de Newton appliquée à la fonction $f(x) = x^2 - 3$, en partant de $x_0 = 2$. On peut montrer que dans ce cas, la suite $(x_n)_n$ est strictement décroissante, et convergente de limite $\sqrt{3}$, ce qui n'est pas demandé. On admettra que $\sqrt{3} > 1,7$ pour la suite de l'exercice. La méthode aboutit à la suite :

$$x_{n+1} = \frac{x_n^2 + 3}{2x_n}.$$

3.a. On note $\varepsilon_n = x_n - \sqrt{3} \in]0; 0,3[$ l'erreur au rang n. Vérifiez que :

$$\varepsilon_{n+1} = \frac{\varepsilon_n^2}{2x_n},$$

et en déduire que :

$$\varepsilon_{n+1} \leq \frac{\varepsilon_n^2}{a},$$

avec $a = 3,4$.

3.b. En déduire que :

$$\varepsilon_n \leq \frac{\varepsilon_0^{2^n}}{a^{2^n-1}},$$

puis, en remarquant que $\varepsilon_0/a < 0,1$, que :

$$\varepsilon_n \leq \frac{a}{10^{2^n}}.$$

Quelle valeur de n choisir pour avoir $x_n - \sqrt{3} \leq 10^{-p}$?

Exercice 3.
1. Soit g une fonction dérivable en un point a. Rappeler pourquoi, si x est proche de a, $g(x)$ peut être approché par $g(a) + g'(a)(x-a)$ (on ne demande pas d'explication rigoureuse, juste une intuition fondée).
2.a. On suppose maintenant que g est deux fois dérivable sur un intervalle ouvert I contenant a, et l'on note $M_2 = \sup_I |g'|$. On introduit :

$$K(h) = g(a+h) - (g(a) + g'(a)h).$$

Déterminer K' et K''.
2.b. Montrer que :

$$|K''(h)| \leq M_2,$$

puis en déduire que :

$$|K(h)| \leq \frac{M_2}{2} h^2$$

(une démonstration dans le cas où $h \geq 0$ est suffisante).
2.c. Expliquer pourquoi ce que l'on vient de faire rend rigoureux le résultat de la question **1**.
3. On veut déterminer, sans calculatrice, une valeur approchée de 101. La formule du **1** avec $a = 1$ dit que si x est proche de 1, alors $\sqrt{x}$ est proche de $1 + \frac{x-1}{2}$.
3.a. Vous paraît-il raisonnable de faire ce calcul avec $x = 100$? Pourquoi ?
3.b. On peut aussi remarquer que $\sqrt{101} = 10\sqrt{1,01}$, et approcher $\sqrt{1,01}$ par la formule précédente avec $x = 1,01$. Cela vous paraît-il plus raisonnable (pourquoi) ? Déterminer la valeur approchée ainsi obtenue, et un majorant de l'erreur commise (pour ce dernier point, on prendra $M_2 = \frac{1}{2}$).

5.5 Interrogation de janvier 2007.

Exercice 1. On considère x_1, x_2, x_3 trois réels distincts, $(y_1, y_2, y_3, z_1, z_2, z_3) \in \mathbb{R}^6$. On considère l'ensemble $\mathcal{P}$ des polynômes P tels que :

$$\forall i \in \{1,2,3\}, \quad P(x_i) = y_i, \quad P'(x_i) = z_i.$$

1. On rappelle que dans $\mathcal{P}$, il y a un unique élément de degré minimal, que l'on note ici P_0. Que peut-on dire du degré de P_0 ?
2. Soit $P \in \mathcal{P}$. Que peut-on dire de $P - P_0$?
3. Ecrire une fonction SCILAB dont la première ligne est :
`function c=P0(x1,x2,x3,y1,y2,y3,z1,z2,z3)`
retournant dans `c` les coefficients de P_0.

Exercice 2. Soit $\sigma \in]0;1]$. On note P_0 le polynôme d'interpolation (de degré au plus 1) d'une fonction f aux points σ et $-\sigma$, et l'approximation de $\int_{-1}^{1} f(x)dx$

par $\int_{-1}^{1} P_0(x)dx$ aboutit à une formule du type :

$$\int_{-1}^{1} f(x)dx \approx c_1 f(-\sigma) + c_2 f(\sigma).$$

1.
1.a. Vérifier que :

$$P_0(x) = f(-\sigma) + \frac{f(\sigma) - f(-\sigma)}{2\sigma}(x + \sigma).$$

1.b. Calculer $\int_{-1}^{1} P_0(x)dx$.
1.c. En déduire que $c_1 = c_2 = 1$.

2. Désormais, la formule d'approximation prend donc la forme :

$$\int_{-1}^{1} f(x)dx \approx f(-\sigma) + f(\sigma),$$

On rappelle que la méthode est d'ordre q si et seulement si, pour tout $k \leq q$, on a :

$$\int_{-1}^{1} x^k dx \approx (-\sigma)^k + \sigma^k.$$

2.a. Sachant que la méthode est basée sur 2 points, quel encadrement peut-on donner de l'ordre q de la méthode ?
2.b. Déterminer σ pour que la formule soit (au moins) d'ordre 2.
2.c. Vérifier que la formule ainsi obtenue est d'ordre 3.
2.d. Si f est de classe C^4 sur $[-1; 1]$, en notant Q le polynôme de degré au plus 3 tel que $Q(x_i) = f(x_i)$ et $Q'(x_i) = f'(x_i)$, aboutir à une formule d'erreur faisant intervenir $M_4 = \sup |f^{(4)}|$ et $\int_{-1}^{1}(x^2 - \sigma^2)^2 dx$.

5.6 Partiel de janvier 2007.

Petite question. Quelles sont, selon vous, les avantages et les limites de l'utilisation d'un logiciel tel SCILAB pour résoudre des problèmes numériques ? Comment remédier à certaines limites ?

Exercice 1. On considère x_1, x_2, x_3, x_4 quatre réels distincts, $(y_1, y_2, y_3, y_4, z_1, z_2, z_3, z_4) \in \mathbb{R}^8$. On considère l'ensemble $\mathcal{P}$ des polynômes P tels que :

$$\forall i \in \{1, 2, 3, 4\}, \quad P(x_i) = y_i, \quad P'(x_i) = z_i.$$

1. On rappelle que dans $\mathcal{P}$, il y a un unique élément de degré minimal, que l'on note ici P_0. Que peut-on dire du degré de P_0 ?

2. Soit $P \in \mathcal{P}$. Que peut-on dire de $P - P_0$?
3. On considère maintenant une fonction f dérivable pour laquelle on a : $y_i = f(x_i)$ et $z_i = f'(x_i)$ pour tout i.
3.a. On considère la suite formée par les itérées issues de la méthode de Newton pour la suite $(u_n)_n$: $u_{n+1} = g(u_n)$. Rappeler l'expression de g en fonction de f.
3.b. En déduire les valeurs z_1, z_2, z_3, z_4 en fonction de x_1, x_2, x_3, x_4 et de y_1, y_2, y_3, y_4, de sorte que si $u_0 = x_1$, on a : $u_1 = x_2$, $u_2 = x_3$, $u_3 = x_4$, puis $u_4 = x_1$.
3.c. Que remarquez vous pour la suite $(u_n)_n$?
3.d. Déduire de **1** et de **3.b.** comment fabriquer une fonction f de sorte que la suite des itérées de Newton partant de x_1 soit $x_1, x_2, x_3, x_4, x_1, x_2, x_3, x_4, \ldots$.

Exercice 2. On veut calculer les intégrales :

$$I_A = \int_0^A e^{-x} x^{5/2} dx,$$

pour $A = 1$ et $A = +\infty$. Les questions sont indépendantes entre elles (mais ce n'est pas le cas d'une sous-question à l'autre).
1. On suppose ici que $A = +\infty$. Expliquer comment, à l'aide de lois exponentielles et de la méthode de Monte-Carlo, comment procéder. Donner la (ou les) ligne(s) de commande SCILAB que vous allez entrer. Quelles semblent être les limites de la méthode de Monte-Carlo. Comment y remédier ?
2. On suppose ici que $A = 1$. On va se servir d'un développement en série.
2.a. Justifier que

$$e^{-x} x^{5/2} = \sum_{n=0}^{+\infty} \frac{(-1)^n x^{5/2+n}}{n!}.$$

2.b. On admet que l'on peut intervertir la série et l'intégrale. Montrer qu'alors :

$$I_1 = \sum_{n=0}^{+\infty} \frac{(-1)^n}{n!(7/2+n)}.$$

2.c. On approche I_1 par une somme partielle S_N de la série. Que peut-on dire de l'erreur ? En déduire comment, dans SCILAB, trouver un encadrement de I_1 d'amplitude 10^{-5} (on écrira la ligne de commande).
3. On suppose toujours que $A = 1$. On envisage d'appliquer comme méthode composée la formule des rectangles à gauche. A quelle erreur aboutit-on ? Ecrire un fichier script SCILAB (ou une ligne de commande) permettant d'avoir I_1 à 10^{-3} près.

5.7 Interrogation de janvier 2008.

Exercice 1.
On veut calculer de manière approchée les séries (convergentes) :

$$S_1 = \sum_{k=1}^{+\infty} \frac{1}{k^3}$$

et

$$S_2 = \sum_{k=1}^{+\infty} \frac{(-1)^k}{k^3}.$$

1. Montrer que :

$$(a \leq b \leq c) \Longleftrightarrow \left(\left| b - \frac{a+c}{2} \right| \leq \frac{c-a}{2} \right).$$

2. Notons $R_n^{(1)}$ le reste pour la première série et $U_n^{(1)}$ la somme partielle. Montrer que :

$$\frac{1}{2(n+1)^2} \leq R_n^{(1)} \leq \frac{1}{2n^2}$$

et en déduire que :

$$\left| S_1 - \left(U_n^{(1)} + \frac{2n^2+2n+1}{4n^2(n+1)^2} \right) \right| \leq \frac{2n+1}{4n^2(n+1)^2}.$$

On obtient donc un encadrement d'amplitude approximative $1/n^3$.
3. Rappeler comment on obtient un encadrement de S_2. Quelle est l'amplitude de l'encadrement obtenu ?
4. Programmer le calcul approché de la première méthode dans SCILAB. Le programme devra prendre en entrée n et fournir l'encadrement de S_1 obtenu.

Exercice 2. Soit $f : I \to \mathbb{R}$ une fonction deux fois dérivable, où I est un intervalle ouvert non vide de $\mathbb{R}$. On se donne trois points distincts de I, x_0, x_1, x_2.
On définit par récurrence :

$$f[x_i] = f(x_i), \quad i = 0, \ldots, 2,$$

$$f[x_i, x_j] = \frac{f[x_j] - f[x_i]}{x_j - x_i}, \quad i, j = 0, \ldots, 2 \quad \text{distincts},$$

$$f[x_i, x_j, x_k] = \frac{f[x_j, x_k] - f[x_i, x_k]}{x_j - x_i}, \quad i, j, k = 0, \ldots, 2 \quad \text{distincts}.$$

1. Soit i, j deux indices distincts. Démontrer que :

$$f[x_i, x_j] = f[x_j, x_i],$$

On admettra que $f[x_i, x_j, x_k]$ ne dépend pas de l'ordre de x_i, x_j, x_k.
2. On note P_2 le polynôme d'interpolation de Lagrange de f aux points x_0, x_1, x_2. Que peut-on dire de son degré ? Démontrez que :

$$P_2(X) = f[x_0] + (X - x_0)f[x_0, x_1] + (X - x_0)(X - x_1)f[x_0, x_1, x_2].$$

3. On considère maintenant le cas particulier de $f : x \mapsto x^4$, avec $x_0 = 1$, $x_1 = 1 + h$, $x_2 = 1 + 2h$, où h est un réel strictement positif.
3.a. Donner l'expression du polynôme de Taylor de degré 2 au point x_0. On note T_2 ce polynôme.
3.b. Déterminer les coefficients $f[x_0]$, $f[x_0, x_1]$ et $f[x_0, x_1, x_2]$ de P_2 (on ne demande pas de développer ces coefficients). En déduire une expression de P_2.
3.c. En négligeant les termes d'ordre strictement supérieur à 1 en h, donner des approximations de $f[x_0]$, $f[x_0, x_1]$ et $f[x_0, x_1, x_2]$ lorsque h tend vers 0. Vérifier qu'alors P_2 s'approche par une expression, lorsque h est proche de 0 :

$$P_2(X) \approx Q_2(X) + hR_2(X),$$

où Q_2 et R_2 sont deux polynômes de degré au plus 2 et ne dépendant pas de h. Reconnaissez-vous Q_2 ?
4. Ecrire une fonction SCILAB prenant en entrée le vecteur $(x_0, x_1, x_2, f(x_0), f(x_1), f(x_2))$ et donnant en sortie le vecteur $(f[x_0], f[x_0, x_1], f[x_0, x_1, x_2])$.

5.8 Interrogation de janvier 2008.

Exercice 1. On s'intéresse à l'ensemble $\mathcal{F}$ des polynômes P satisfaisant les conditions suivantes :

$$P(1) = 1, \quad P'(1) = 1, \quad P(2) = 0.$$

1. On note P_0 l'unique polynôme de degré inférieur ou égal à 2 satisfaisant ces conditions. Comment feriez-vous dans SCILAB pour déterminer ce polynôme ?
2. Décrire l'ensemble $\mathcal{F}$ à partir de P_0. Pour f suffisament dérivable vérifiant $f(1) = 1, \quad f'(1) = 1, \quad f(2) = 0$, majorer $|f(t) - P_0(t)|$ lorsque $t \in [1; 2]$.
3. Calculez P_0.

Exercice 2. On s'intéresse à déterminer une valeur approchée de $I = \int_0^1 t^p dt$, avec p entier au moins égale à 2. Bien entendu, cette intégrale est connue, mais nous allons ignorer cet aspect des choses dans la question 1. Pour cela, nous allons appliquer la méthode composée basée sur la formule des rectangles à gauche, mais sans nécessairement faire un découpage régulier.
1. Soit $0 = x_0 < x_1 < \ldots < x_{n-1} < x_n = 1$ les points du découpage, on note pour $i = 0, \ldots, n-1$, $h_i = x_{i+1} - x_i$.
1.a. Que vaut :

$$\sum_{i=0}^{n-1} h_i \quad ?$$

1.b. On applique la méthode élémentaire des rectangles à gauche sur $[x_i, x_{i+1}]$. Montrer que l'on peut aboutir à une majoration du type :

$$\left| \int_{x_i}^{x_{i+1}} t^p dt - h_i x_i^p \right| \leq \frac{p x_{i+1}^{p-1} h_i^2}{2}.$$

En déduire une majoration de l'erreur dans la méthode composée. On exprimera ce majorant uniquement en fonction de p et des h_i, sans les x_i.
1.c. On cherche à choisir les h_i de sorte à minimiser le majorant. Ecrire ce problème comme un problème d'optimisation en $(h_1, \ldots, h_{n-1})$.
2. Calculer I explicitement et écrire la formule des rectangles à gauche avec un découpage régulier. En déduire la limite de la suite $\left(\frac{1}{n^{p+1}} \sum_{k=0}^{n-1} k^p\right)_n$.

Exercice 3. Proposer une méthode de calcul approché de $I = \int_1^{+\infty} \frac{e^{-t}}{t} dt$ dans SCILAB (on fera attention au fait que l'intégrale va jusqu'à $+\infty$). La méthode, expliquée mathématiquement d'abord et suivie d'un programme, devra pouvoir, à partir d'un seuil $\varepsilon > 0$, fournir une valeur I_ε satisfaisant :

$$|I - I_\varepsilon| \leq \varepsilon.$$

5.9 Interrogation de janvier 2007.

SCILAB. Il est possible de montrer que pour tout $n \in \mathbb{N}^*$, il existe un polynôme T_n de degré n tel que, pour tout $\theta \in \mathbb{R}$:

$$\cos(n\theta) = T_n(\cos(\theta)).$$

Ainsi $\cos(\theta) = \cos(\theta)$ donc $T_1(X) = X$, $\cos(2\theta) = 2\cos^2(\theta) - 1$ donc $T_2(X) = 2X^2 - 1$, etc. Expliquer comment, à l'aide de SCILAB (et sans utiliser les formules de trigonométrie de lycée), trouver les six coefficients du polynôme T_5.

Exercice. On rappelle que pour tout réel x, on a :

$$e^x = \sum_{n=0}^{+\infty} \frac{x^n}{n!}$$

et que plus précisément :

$$\forall N \in \mathbb{N}^*, \quad \forall x \in \mathbb{R}, \quad \left| e^x - \sum_{n=0}^{N} \frac{x^n}{n!} \right| \leq \frac{\max\{1, e^x\} |x|^{N+1}}{(N+1)!}.$$

On définit la fonction g sur $\mathbb{R}^*$ par $g(x) = \frac{e^x - 1}{x}$. On admettra que g se prolonge en une fonction C^∞ sur $\mathbb{R}$ qui est somme d'une série entière.

1. Comment feriez-vous pour calculer une valeur approchée de $g'(1)$ (sans calculer la dérivée) ?
2.
2.a. Calculer le DSE de g, puis celui de g'.
2.b. En déduire que

$$g'(1) = \sum_{p=1}^{+\infty} \frac{p}{(p+1)!}.$$

2.c. Soit $N \geq 1$. Démontrer que :

$$\left| g'(1) - \sum_{p=1}^{N} \frac{p}{(p+1)!} \right| \leq \frac{e}{(N+1)!}.$$

2.d. En déduire comment, $\varepsilon > 0$ étant donné, donner une valeur approchée de $g'(1)$ à ε près.

5.10 Partiel de janvier 2009.

Exercice. On considère la fonction g définie sur $\mathbb{R}^*$ par l'expression :

$$\forall x \in \mathbb{R}^*, \quad g(x) = \frac{e^x - 1}{x}.$$

Elle est prolongeable en une fonction de classe C^∞ sur $\mathbb{R}$, développable en série entière. Ce développement permet de montrer que toutes les dérivées sont positives sur $\mathbb{R}^+$. Le but de l'exercice est de proposer plusieurs méthodes de calcul de l'intégrale :

$$I = \int_0^1 g(t)dt.$$

Partie A. Dans cette partie, on envisage d'utiliser une méthode de Monte-Carlo en utilisant une loi uniforme sur $[0; 1]$. On rappelle que la loi uniforme sur $[0; 1]$ a pour moyenne $1/2$ et variance $1/12$.
1. Rappeler le principe de la méthode de Monte-Carlo. Comment la met-on en pratique ?
2. Construire des intervalles de confiance à 95%.
3. Ecrire le programme dans SCILAB.

Partie B. Dans cette partie, on envisage d'utiliser une méthode composée de Simpson.
1. Rappeler le principe de la méthode élémentaire de Simpson.
2. La formule d'erreur pour la méthode composée donne une majoration en $\frac{M_4}{2880n^4}$, où $M_4 = \sup_{x\in[0;1]} |g^{(4)}(x)|$. Il faut donc estimer tout d'abord M_4 (ce qui est l'objet des questions 2 et 3). Montrer que $M_4 = g^{(4)}(1)$.

3. L'objet de cette question est de fournir une approximation de $g^{(4)}(1)$. On pose $f = g''$, de sorte que $g^{(4)} = f''$.
3.a. Soit $h > 0$ petit. Proposer une approximation de $f''(1)$ faisant intervenir $f(1+h)$, $f(1)$, $f(1-h)$ et h.
3.b. En déduire l'approximation suivante pour $g^{(4)}(1)$:

$$g^{(4)}(1) \approx \frac{g(1+2h) - 4g(1+h) + 6g(1) - 4g(1-h) + g(1-2h)}{h^4}.$$

3.c. On a tapé dans SCILAB les commandes suivantes :

```
deff('y=g(x)','y=(exp(x)-1)./x')
h=1/10;
for i=1:5
(g(1+2*h)-4*g(1+h)+6*g(1)-4*g(1-h)+g(1-2*h))/h^4
h=h/10;
end
```

3.c.i. Que fait le programme ?
3.c.ii. Les réponses fournies sont successivement 0.4651113, 0.4645423, 0.4651834, -6.6613381, -22204.46. Proposez un majorant raisonnable (i.e. certain mais pas trop fort) pour M_4.
4. A l'aide de tous ces éléments, comment feriez vous dans SCILAB pour trouver un nombre J tel que $|I - J| \leq \varepsilon$, $\varepsilon > 0$ étant donné ?

Partie C. Dans cette partie, nous envisageons d'utiliser un calcul par une série.
1. Montrer que :

$$I = \sum_{p=0}^{+\infty} \frac{1}{(p+1)(p+1)!}.$$

On note :

$$R_N = \sum_{p \geq N+1} \frac{1}{(p+1)(p+1)!}.$$

2. Montrer que :

$$0 \leq R_N \leq \frac{1}{N+2} \sum_{p \geq N+1} \frac{1}{(p+1)!}$$

et en déduire :

$$0 \leq R_N \leq \frac{e}{(N+2)!}.$$

3. A l'aide de tous ces éléments, comment feriez vous dans SCILAB pour trouver un nombre J tel que $|I - J| \leq \varepsilon$, $\varepsilon > 0$ étant donné ?

Partie D. Dans cette partie, nous reprenons les notations du C, et nous allons étudier plus précisément comment choisir un entier N de sorte que $R_N \leq \varepsilon$.

1. Démontrer que pour tout $j \geq 2$,

$$\ln(j) \geq \int_{j-1}^{j} \ln(t)dt.$$

De cette majoration, il est possible de déduire que :

$$\ln((N+2)!) \geq \int_{1}^{N+2} \ln(t)dt.$$

2. Déduire de la question précédente que si :

$$(N+2)(\ln(N+2)-1) \geq -\ln(\varepsilon),$$

alors $R_N \leq \varepsilon$ (on utilisera C.2.).
3. Soit K la fonction définie sur $J = [1; +\infty[$ par :

$$K(x) = x(\ln(x) - 1).$$

3.a. Montrer que K est convexe sur J, et qu'elle est une bijection strictement croissante de J sur son image $K(J)$ que l'on déterminera.
3.b. On prend $\varepsilon \in]0; \exp(-e^2)[$, et l'on veut résoudre de manière approchée :

$$K(x) = A_\varepsilon,$$

avec $A_\varepsilon = -\ln(\varepsilon)$ par la méthode de dichotomie. Rappeler le principe de la méthode de dichotomie. Quels sont ses avantages et inconvénients ? Quel intervalle de départ choisissez vous ? Combien d'itérations faudra t-il pour avoir un intervalle de longueur inférieure où égale à 1/2 ? Proposer un programme dans SCILAB. La méthode fournit un intervalle $[a, b]$. Comment choisissez vous le N de sorte que $R_N \leq \varepsilon$?
3.c. On souhaite résoudre l'équation précédente par la méthode de Newton. Rappeler le principe de la méthode de Newton. Expliquer pourquoi la méthode sera nécessairement convergente.

5.11 Partiel de juin 2011.

Cet examen porte autour du calcul approché de π. Les trois parties sont indépendantes.

I. Utilisation d'une série (sur 8). Pour cette partie, nous admettrons que :

$$\sum_{n=1}^{+\infty} \frac{1}{n^4} = \frac{\pi^4}{90}.$$

On note $S_N = \sum_{n=1}^{N} \frac{1}{n^4}$ la somme de la série, et $R_N = \sum_{n=N+1}^{+\infty} \frac{1}{n^4}$ son reste.
1. Démontrer l'encadrement

$$0 \leq R_N \leq \frac{1}{3N^3}.$$

2. Notons $\alpha = \frac{\pi^4}{90}$, et ϵ_α^a l'erreur absolue obtenue en remplaçant α par S_N.
2.a. Donner un encadrement de ϵ_α^a.
2.b. Ayant approché α par S_N, donner une expression T_N (ne faisant intervenir que S_N) permettant d'approcher π.
2.c. Estimer l'erreur absolue ϵ_π^a commise en approchant π par T_N en fonction de ϵ_α^a.
3. À l'aide des questions précédentes, une précision $\varepsilon > 0$ sur π étant requise, comment vous y prendriez vous pour approcher π ? Ecrire le programme SCILAB.

II. Une technique aléatoire (sur 8).
Considérons une suite de tirages (X_i, Y_i), $1 \leq i \leq N$ où les X_i et Y_j sont tous indépendants et suivent une loi uniforme sur $[0; 1]$. Lorsqu'un point (X_i, Y_i) tombe dans le quart de cercle :

$$C = \{(x, y) \in (\mathbb{R}^+)^2, \quad x^2 + y^2 \leq 1\}$$

on le compte (on ajoute 1 à un compteur C_i valant initialement 0), sinon on le rejète. On s'intéresse au pourcentage de points arrivant dans le cercle, c'est-à-dire à C_N/N.
1. Intuitivement, quelle est la limite de C_N/N ?
2. Le but de cette question est de justifier l'intuition.
2.a. On note $Z_i = 1_{X_i^2+Y_i^2\leq 1}$. Quelles sont les valeurs que peut prendre la v.a. Z_i ? En déduire que Z_i suit une loi de Bernoulli d'un paramètre p que l'on précisera.
2.b. Comparer C_N à $Z_1 + \ldots + Z_N$.
2.c. On admet que les Z_i sont indépendantes. Justifier que :

$$\frac{Z_1 + \ldots + Z_N}{N} \to \frac{\pi}{4}$$

et conclure.
3. Justifiez que :

$$\sqrt{n}\left(\frac{Z_1 + \ldots + Z_N}{N} - \frac{\pi}{4}\right) \to \mathcal{N}(0, \sigma^2)$$

avec un σ à préciser. En déduire des intervalles de confiance à 95%.

III. Autres (sur 6). Proposez une autre méthode de calcul approché de π, non basée sur un calcul approché de série ni sur une technique aléatoire. Vous exposerez les grandes lignes de votre méthode, sans donner tous les détails.

5.12 Interrogation de novembre 2011.

Si E_1 (resp. E_2, resp. S) désigne votre note dans l'exercice 1 (resp. l'exercice 2, resp. scilab), votre note sera égale à :

$$S + \min\{E_1 + E_2, 16\}.$$

SCILAB (sur 4). Ecrire une ligne SCILAB permettant de calculer :

$$\sum_{j=1}^{25} \frac{1-j^2}{1+j^3}.$$

Exercice 1 (sur 10). On considère un ensemble de 121 points (x_i, y_i) $(1 \leq i \leq 121)$ du plan. On s'intéresse à les approcher par une fonction.
1. Rappeler le principe de la méthode d'interpolation (de Lagrange). On expliquera en particulier le principe de la méthode, sous quelle forme on cherche la (ou les solutions) du problème d'un point de vue théorique puis comment on procède en pratique.
2. En fait les points semblent approximativement se répartir selon une courbe $y = ax^3 + bx^2 + cx + d$. Expliquer le principe de la méthode permettant d'obtenir l'équation de la courbe.
3. Comment feriez vous pour choisir entre les deux méthodes ?

Exercice 2 (sur 10).
1. Justifier rapidement que pour tout réel $y > 0$, il existe $c \in]-y, 0[$ tel que :

$$e^{-y} - (1-y) = \frac{e^c}{2} y^2.$$

En déduire que la valeur absolue de l'erreur absolue commise en remplaçant e^{-y} par $1-y$ est encadrée entre 0 et $y^2/2$.
2. Soit $x > 0$. On se propose d'approcher $e^{-x} = (e^{-x/n})^n$ par :

$$u_n(x) = \left(1 - \frac{x}{n}\right)^n.$$

2.a. Donner un majorant de la valeur absolue de l'erreur absolue commise en remplaçant e^{-x} par $u_n(x)$.
2.b. Comment faudrait-il choisir n pour avoir l'erreur la plus faible ? Répondre théoriquement puis ensuite dans le cadre d'un logiciel à précision limitée.

Remarques de correction.
Partie SCILAB (sur 4) Faire une boucle n'est pas dans l'esprit SCILAB et évite

beaucoup les risques d'erreurs de syntaxe. Par conséquent, la boucle est notée sur 2 seulement. Si l'on fait une boucle, l'oubli de l'initialisation est désastreuse, l'oubli du end gênante et le non contrôle des affichages pourra être désagréable au lecteur ; points qui n'ont pas échappé au correcteur. Ceux qui se lancent dans la vectorisation récoltent deux points d'office. Le point avant la division a souvent été oublié ; en revanche devant un signe + ou un signe −, il n'est pas nécessaire du fait que l'opération vectorisée ou non est la même (ce dernier point n'a pas été sanctionné).

Exercice 1.
1. (sur 4). En premier lieu, il s'agissait d'adapter son cours à la situation décrite dans l'exercice et non le recopier, d'autant plus qu'une feuille recto-verso était autorisée. Recopier son cours, le td ou le poly a fait mauvaise impression. Il s'agissait ici d'expliquer que l'on cherche les polynômes passant par les 121 points. On sait qu'il en existe un et un seul, P_0 de degré **inférieur ou** égal à 120, on connaît tous les autres à partir de celui-ci. Le calcul pratique fait appel à la résolution d'un système linéaire plutôt qu'aux polynômes L_i, qui aboutirait à un calcul pénible au final (essayez avec 4 points par exemple !). Les y_i n'étant pas issus d'une fonction, on ne peut pas parler de l'erreur $f(x) - P_0(x)$. La méthode d'ailleurs ne s'appliquant ici qu'aux x_i, elle a une erreur nulle.
2. (sur 4). Là encore, il s'agissait d'adapter son cours à la situation décrite dans l'exercice. Il s'agissait d'expliquer qu'ici on ne cherche pas à passer par tous les points, mais par y passer au mieux avec un polynôme de degré plus petit. On écrit le problème sous la forme $Y = Xb + u$ (il fallait donner les expresssions au moins de X et b) où u est le vecteur des erreurs dont on cherche à minimiser la norme euclidienne. Dire ensuite qu'il y a une unique solution ssi la matrice X est de rang 4 (ce qui exigeait ici qu'au moins quatre x_i soient distincts, mais je n'en demandais pas tant) et qu'alors la solution du problème des moindres carrés est donnée par $({}^tXX)^{-1}{}^tXY$ (qui n'est pas égal à $X^{-1}Y$ du fait que X n'est pas carrée).
3. (sur 2). Il s'agit d'une question plus ouverte donc des réponses pertinentes ont pu apporter des points. En fait, l'idée est qu'un polynôme de degré 3 est bien mieux manipulable qu'un polynôme de degré 120 (car la première méthode a toutes les chances de donner un polynôme de degré 120 exactement) et que donc on risque de préférer la méthode 2, sous réserve qu'elle ne donne pas une erreur relative trop grande. Tout le problème serait d'ailleurs comment mesurer cette erreur relative, mais je n'en demandais pas tant.

Exercice 2.
1. (sur 3). A ma grande surprise, le début de la question n'a eu que très peu de succés, il s'agit pourtant tout simplement de la formule de Taylor avec reste de Lagrange (ou si vous préférez de la formule d'erreur dans l'interpolation, cas de Taylor). La suite était immédiate avec ce qui précède, si ce n'est une coquille d'énoncé qui ne semble pas vous avoir troublé.

2. (sur 3). Il s'agissait d'utiliser ce qui précède pour avoir l'erreur obtenue en remplaçant $e^{-x/n}$ par $(1 - x/n)$ puis d'utiliser la formule de composition avec $f : z \mapsto z^n$.
3. (sur 4). En théorie il faudrait prendre $n = \infty$ pour minimiser l'erreur, d'ailleurs certains ont bien rappelé (ou redémontré) que $(1 - x/n)^n \to e^{-x}$. En pratique, si x/n est trop petit, le logiciel remplacera $1 - x/n$ par 1 ce qui limite la possibilité du choix du n à des valeurs qui ne rentrent pas dans cette zone.

5.13 Partiel de Juin 2012.

Exercice 1. Soit θ un nombre réel strictement positif. On s'intéresse aux solutions réelles de l'équation (E_θ) :

$$x^3 + \theta x - 1 = 0.$$

On note $I = [0; 1]$.
1. Montrer que l'équation a une unique solution réelle x_θ, puis que $x_\theta \in I$.
2. On définit la fonction $g_\theta : I \to I$ par l'expression :

$$g_\theta(x) = \frac{1 - x^3}{\theta}.$$

2.a. Montrer que $g_\theta(I) \subset I$ et que x_θ est l'unique point fixe de g_θ.
2.b. Montrer que pour $\theta > \theta_0$ (avec θ_0 à préciser), la fonction g_θ est une contraction.
2.c. Lorsque $\theta = 10$, écrire un script scilab permettant d'obtenir une valeur approchée de x_θ.
3. Maintenant, on s'intéresse à ce qui se passe lorsque θ devient très grand.
3.a. Justifier que $\theta x_\theta \in I$ et en déduire $\lim_{\theta \to +\infty} x_\theta$.
3.b. Expliquer heuristiquement pourquoi :

$$x_\theta \sim \frac{1}{\theta}.$$

3.c. On suppose que l'on dispose d'une fonction scilab `raci` qui prend θ en entrée et donne x_θ en sortie. Comment feriez-vous pour vérifier dans SCILAB ce que l'on a présenti en **3.b** ?

Exercice 2.
1. Discuter la qualité du générateur congruentiel suivant pour produire des nombres pseudo-aléatoires:

$$Z_0 = 1, \quad Z_i = 11\, Z_{i-1} \mod 16.$$

(On pourra par exemple calculer les quelques premières valeurs des Z_i.)
2. On suppose que l'on dispose d'un générateur d'une variable U uniforme sur

$[0,1]$. Expliquer comment l'on peut simuler une variable X suivant une loi:
2.a. géométrique, donc de distribution $P(X=n)=(1-p)^{n-1}p$, avec $p\in[0,1]$.
2.b. exponentielle, donc de densité $f_X(x)=\lambda e^{-\lambda x}$, avec $\lambda>0$.

Exercice 3.
1. Écrire un programme Scilab permettant de générer un échantillon aléatoire $(Y_1,\dots,Y_{50})$ de taille $n=50$ où chaque variable aléatoire Y_i suit une loi normale de paramètres $\mu=1$ et $\sigma^2=2$.
2. Donner la vraisemblance de l'échantillon aléatoire $(Y_1,\dots,Y_{50})$. Rappel : si $X\sim\mathcal{N}(\mu,\sigma^2)$, sa densité est donnée par : $f_X(x)=\frac{1}{\sqrt{2\pi\sigma^2}}\exp(-\frac{1}{2\sigma^2}(x-\mu)^2)$, $\forall x\in\mathbb{R}$.
3. Démontrer que l'estimateur du maximum de vraisemblance de μ est donné par : $\hat{\mu}=\frac{1}{N}\sum_{i=1}^{N}Y_i$.
4. Quel est l'estimateur du maximum de vraisemblance de σ^2 ?

5.14 Interrogation de décembre 2012.

Partie Mathématique.

Exercice 1. On dispose de $n>4$ points (x_i,y_i) qui semblent se répartir selon une courbe d'équation $y=ax^3+bx^2+cx+d$. Comment feriez vous pour trouver les coefficients (a,b,c,d) ?

Exercice 2.
1. Démontrer que pour tout réel positif x :

$$x-\frac{x^3}{6}\leq\sin(x)\leq x$$

(on pourra commencer par l'inégalité de droite).
2. On définit la fonction f par :

$$\forall x\in\mathbb{R},\quad f(x)=x-\sin(x).$$

Trouver un $\alpha_0>0$ de sorte que si $\alpha>\alpha_0$, la série :

$$\sum_{n=1}^{+\infty}f\left(1/n^{\alpha}\right)$$

converge.
3. Soit $\alpha>\alpha_0$. Comment feriez-vous pour calculer la somme de la série avec une précision requise ?

Partie Informatique.

Les deux questions sont indépendantes.

Question 1. Calculer à l'aide de SCILAB :

$$\sum_{n=1}^{10} n^{-n}.$$

Question 2. Quelle sera la réponse au lancer du script suivant :

```
N=1;S=0
while S<=10
S=S+N;
N=N+1;
end
S
```

5.15 Partiel de janvier 2013.

Partie Mathématique (sur 16).

Exercice 1 (sur 5). 1. Démontrer que pour tout réel positif x :

$$1 - \frac{x^2}{2} \leq \cos(x) \leq 1.$$

2. On définit la fonction f par :

$$\forall x \in \mathbb{R}, \quad f(x) = 1 - \cos(x).$$

Trouver un $\alpha_0 > 0$ de sorte que si $\alpha > \alpha_0$, la série :

$$\sum_{n=1}^{+\infty} f\left(1/n^{\alpha}\right)$$

converge.
3. Soit $\alpha > \alpha_0$. Comment feriez-vous pour calculer la somme de la série avec une précision requise ?

Exercice 2 (sur 3). Proposer une méthode de calcul approchée de l'intégrale :

$$I = \int_{-\infty}^{+\infty} e^{-x^2} \sqrt{1 + x^4} dx.$$

On expliquera le principe de la méthode, sa mise en oeuvre et l'on indiquera la vitesse de convergence de la méthode.

Exercice 3 (sur 3). On donne les valeurs approchées suivantes :

$$x_1 = \sqrt{2} \approx 1,414, \quad x_2 = \pi \approx 3,142, \quad x_3 = \sqrt{3} \approx 1,732.$$

1. Indiquer, n'ayant que ces informations, un majorant des erreurs absolues et relatives sur chacun de ces nombres (il n'est pas demandé d'effectuer les opérations pour le calcul des erreurs relatives).
2. On note ϵ_i^r l'erreur relative sur x_i. On note $x = x_1x_2x_3$. Justifier que l'erreur relative sur x peut s'approcher par :

$$\sum_{i=1}^{3} \epsilon_i^r.$$

Exercice 4 (sur 5). Soit $\alpha \in]0;1[$. On introduit la fonction $f : x \mapsto x - \alpha$.
1. Justifier que la fonction f a une unique racine sur $I_0 = [0;1]$ que l'on déterminera.
2. Rappeler le principe de la méthode de dichotomie, ainsi que ses avantages et inconvénients.
3. On applique la méthode de dichotomie à f en partant de :

$$[a_0, b_0] = I;$$

On note $[a_n, b_n]$ l'intervalle obtenu après n itérations.
3.a. Justifier que pour tout n, $2^n a_n$ et $2^n b_n$ sont des entiers positifs.
3.b. En déduire que :

$$\left\{ \frac{p}{2^q}, (p,q) \in \mathbb{N}^2 \right\} \cap I$$

est dense dans I.

Partie Informatique (sur 4).
On note $1_{p,q}$ la matrice à p lignes et q colonnes composée uniquement de 1.
1. Soit X un vecteur colonne de taille $n \times 1$. Calculer le produit matriciel $X1_{1,n}$.
2. On se donne un vecteur colonne $X = {}^t(\lambda_1, \ldots, \lambda_{30})$. Donner une ligne permettant dans SCILAB à partir de X de fabriquer la matrice de Van-der-Monde :

$$M = \left(\lambda_i^{j-1}\right)_{1 \leq i,j \leq 30}$$

(une réponse sans boucle sera appréciée).

5.16 Partiel de juin 2013.

Question 1 (sur 4). Présenter une méthode de résolution de $f(x) = 0$, en discutant ses avantages et inconvénients.

Question 2 (sur 4). Donner une ligne de commande SCILAB permettant de calculer

$$\sum_{n=1}^{100} \frac{n}{n^2+1}.$$

Question 3 (sur 5). Toute matrice 2×2 satisfait la relation :

$$A^2 - \mathrm{tr}(A)A + \det(A)I_2 = 0.$$

Comment feriez vous à l'aide de SCILAB pour vérifier cette propriété ?

Question 4 (sur 7). Soit $f \in C^2(\mathbb{R}, \mathbb{R})$, $x_0 \in \mathbb{R}$.
1. Ecrire le développement limité d'ordre 2 de f au voisinage de x_0.
2. En déduire :

$$\frac{f(x_0+\alpha h) - f(x_0+\beta h)}{h} = (\alpha-\beta)f'(x_0) + \frac{\alpha^2-\beta^2}{2}f''(x_0)h + o(h).$$

3. On veut donner une formule de calcul approché de $f'(x_0)$ en fonction de valeurs uniquement prises par f. Pour ce faire, on fixe $h > 0$ petit, et l'on cherche les valeurs α et β (indépendantes de f et h).
3.a. Quelle relation vous semble t-il raisonnable de poser ?
3.b. Quel choix de α et β vous semble optimal (on justifiera).

5.17 Interrogation de novembre 2013 (1h20).

SCILAB (sur 3). Proposer une ligne de commande pour calculer :

$$\sum_{n=1}^{500} \frac{n+1}{n^3-n+8}.$$

Exercice (sur 17). On note :

$$S = \sum_{n=1}^{+\infty} \frac{1}{n^6}, \quad S_N = \sum_{n=1}^{N} \frac{1}{n^6}, \quad R_N = S - S_N = \sum_{n=N+1}^{+\infty} \frac{1}{n^6}.$$

1. Rappeler pourquoi la somme S est convergente. On admettra que :

$$S = \frac{\pi^6}{945}.$$

2.
2.a. Justifier que pour tout entier $n \geq 2$:

$$\int_n^{n+1} \frac{dt}{t^6} \leq \frac{1}{n^6} \leq \int_{n-1}^{n} \frac{dt}{t^6}.$$

2.b. En déduire que pour tout N (assez grand) :

$$T_{N+1} \leq R_N \leq T_N,$$

avec $T_k = 1/(5k^5)$.
3.
3.a. Fixons $\varepsilon > 0$. Proposer un $N(\varepsilon) \in \mathbb{N}$ tel que $T_{N(\varepsilon)} \leq \varepsilon$.
3.b. En déduire comment donner une valeur approchée de S à ε près.
4. On approche S par la valeur approchée S' proposée dans **3.b.** (qu'il n'est pas nécessaire de calculer). On commet donc une erreur absolue majorée par ε. On approche alors :

$$\pi = (945S)^{1/6}$$

par $(945S')^{1/6}$. Que dire de l'erreur commise ?
5. On revient sur l'encadrement de l'erreur fait en **3**.
5.a. Déterminer une constante $C > 0$ telle que :

$$\left| R_N - \frac{T_N + T_{N+1}}{2} \right| \leq \frac{C}{N^6}.$$

5.b. Reprendre la question **3.b** à la lumière de cette majoration.
5.c. Entre celle vue ici et celle vue dans la question **3**, quelle méthode est la plus rapide (justifier) ?

5.18 Partiel de janvier 2014 (1h20).

SCILAB (sur 3)
Écrire une ligne de commande SCILAB courte pour calculer :

$$\sum_{n=1}^{10} \frac{n^3 - n + 1}{n^4 + \sin(n) + 1}.$$

(Une réponse dans tout autre logiciel ou en pseudo-langage rapportera 0).

Questions (sur 6). Les questions sont indépendantes
1. Quelles sont selon vous les avantages et inconvénients à l'utilisation d'un logiciel type SCILAB dans la résolution de problèmes mathématiques ?
2. Expliquer dans les grandes lignes la méthode des moindres carrés et ses avantages et inconvénients par rapport à une interpolation de Lagrange.

Exercice 1. (sur 8) On veut calculer une valeur approchée de $\sqrt{2}$. Pour cela, on va procéder géométriquement. On va fabriquer une suite de rectangles R_n d'aire totale 2 dont on appelle L_n la longueur et ℓ_n la largeur. On notera que lorsque le rectangle est un carré, son côté est de longueur $\sqrt{2}$.
1. (sur 1). Donner la relation qu'il y a entre L_n et ℓ_n. On en déduit qu'il suffit

de construire la suite $(L_n)_n$, ce sur quoi nous allons nous concentrer.
2. On propose la récurrence suivante :

$$L_{n+1} = \frac{1}{2}\left(L_n + \frac{2}{L_n}\right).$$

2.a. (sur 2). Interprêter ce choix.
2.b. (sur 2). Supposant par exemple que $L_0 > \sqrt{2}$, démontrer (par récurrence) que la suite $(L_n)_n$ est décroissante et que $L_n > \sqrt{2}$ pour tout n.
2.c. (sur 1). En déduire qu'elle converge. Quelle est sa limite ?
3. (sur 2). On se propose de calculer une valeur approchée de $\sqrt{2}$ par la méthode de Newton appliquée à la fonction $f : x \mapsto x^2 - 2$. A quelle suite aboutit-on ?

Exercice 2. (sur 6). On rappelle la formule d'approximation suivante pour le calcul d'une dérivée seconde :

$$f''(x) \sim \frac{f(x+h) + f(x-h) - 2f(x)}{h^2}$$

avec $h > 0$ suffisamment petit. On se propose de donner un majorant de l'erreur commise :

$$\left| f''(x) - \frac{f(x+h) + f(x-h) - 2f(x)}{h^2} \right|$$

pour une fonction suffisamment dérivable. Fixons x et $h > 0$, et définissons :

$$K(s) = f(x+s) + f(x-s) - 2f(x) - s^2 f''(x)$$

pour $s \in [0; h]$.
1. (sur 2). Calculer les trois premières dérivées de K, et en déduire :

$$|K'''(s)| \leq 2sM_4,$$

où M_4 est un majorant de $|f''''|$ sur $[x-h, x+h]$.
2. (sur 3). En déduire :

$$|K(s)| \leq M_4 \frac{s^4}{12},$$

puis la réponse à la question initiale.
3. (sur 1). On applique la formule d'approximation à la fonction :

$$f(x) = x^3 - 12x^2 + 17x - 16.$$

Que se passe t-il ?

5.19 Partiel de juin 2014 (1h30).

SCILAB (sur 6). Calculer dans SCILAB :

$$\sum_{n=1}^{100} \frac{2n + \sin(n)}{n^2 + 2}.$$

Question (sur 6). Présenter la méthode de Newton en discutant les avantages et inconvénients de cette méthode.

Exercice (sur 8). On introduit la suite $(u_n)_n$ définie par $u_0 = 1$ et la récurrence :

$$u_{n+1} = u_n + \frac{1}{u_n}.$$

1. Montrer que pour tout entier $n \geq 1$, $u_n > u_{n-1}$ (ce qui démontre que u_n ne s'annule jamais et donc que la suite est correctement définie).
2. Justifier que $\lim_{n\to+\infty} u_n = +\infty$.
3. Calculer u_{n+1}^2. En déduire que pour tout n :

$$u_{n+1}^2 \geq u_n^2 + 2,$$

puis que :

$$u_n^2 \geq 1 + 2n.$$

4. Justifier que pour tout n :

$$u_{n+1}^2 \leq u_n^2 + 2 + \frac{1}{2n+1},$$

et en déduire :

$$u_n^2 \leq 2n + 1 + V_n,$$

où :

$$V_n = \sum_{j=0}^{n-1} \frac{1}{2j+1}.$$

5. Justifier que :

$$V_n \leq 1 + \frac{\ln(2n+1)}{2},$$

puis en déduire un équivalent de u_n.

5.20 Enoncé de l'interrogation de novembre 2014 (1h).

Exercice 1. On travaille avec un logiciel qui fonctionne à 5 chiffres significatifs. On approche $x = \pi$ par sa représentation x' dans le logiciel.
1. Donner un majorant de $|x - x'|$.
2. Soit $h : t \to t^t$. Montrer que $h'(t) = (\ln(t) + 1)t^t$.
3. On calcule une approximation de x^x par $x'^{x'}$. Donner un majorant de l'erreur commise.

Exercice 2. On cherche à trouver le polynôme P de plus petit degré tel que $P(-1) = -1$, $P(1) = 1$, $P'(-1) = P'(1) = 1/2$.
1. Justifier rapidement en utilisant un résultat du cours qu'il existe un polynôme de degré au plus 3 et un seul solution de ce problème. On le note P_0.
2. Désormais, on suppose que P_0 est en fait l'interpolé d'une fonction $f \in C^4([-1, 1])$ telle que $f(1) = -f(-1) = 1$ et $f'(1) = f'(-1) = 1/2$. Soit $x \in]-1, 1[$.
2.a. Soit x distinct de -1 et 1. Déterminer A_x en fonction de x de sorte que la fonction $g : t \mapsto f(t) - P_0(t) - A_x(t + 1)^2(t - 1)^2$ satisfasse $g(x) = 0$.
2.b. Montrer que g' s'annule (au moins) 4 fois sur $[-1, 1]$. Montrer qu'il existe un ξ_x tel que $g^{(4)}(\xi_x) = 0$.
2.c. En déduire :

$$\forall x \in]-1; 1[, \quad \exists \xi_x \in]-1, 1[, \quad f(x) - P_0(x) = \frac{f^{(4)}(\xi_x)}{4!}(x - 1)^2(x + 1)^2.$$

5.21 Corrigé de l'interrogation de novembre 2014.

Exercice 1.
1. Puisque π est compris entre 1 et 10, son cinquième chiffre significatif est sa quatrième décimale. On a donc :

$$|x - x'| \leq 5 \times 10^{-5}.$$

2. Calcul immédiat en écrivant $h(t) = \exp(t \ln(t))$ puis en utilisant $(e^u)' = e^u u$.
3. Posons $y = x^x$. Puisque $y = h(x)$, la formule du cours dit que :

$$\epsilon_y^a \approx h(x)\epsilon_x^a \approx h'(x')\epsilon_x^a.$$

Avec l'abus habituel, on obtient donc la majoration :

$$|\epsilon_y^a| \leq 5.10^{-5}(\ln(x') + 1)(x')^{x'}.$$

Exercice 2.
1. On utilise le théorème du cours en constatant qu'il y a 4 conditions. On a affaire à une interpolation d'Hermite, et le degré du polynôme P_0 est inférieur ou

égal à $4 - 1 = 3$.
2.
2.a. Ecrivant $g(x) = 0$, on a :

$$A_x = \frac{f(x) - P_0(x)}{(x+1)^2(x-1)^2},$$

(on notera que le dénominateur est non nul).
2.b. Puisque g s'annule en trois points (-1, x et 1), le théorème de Rolle donne au moins deux zéros à g', un dans $]-1, x[$ et l'autre dans $]x, 1[$. On trouve aussi que g' est nulle en 1 et en -1 puisque $f = P_0$ en ces points, donc g' s'annule au moins 4 fois sur $[-1, 1]$. En appliquant successivement Rolle, on trouve (au moins) 3 zéros à g'', deux à g''' et donc un à $g^{(4)}$. On ne notera ξ_x.
2.c. On calcule immédiatement :

$$g^{(4)}(t) = f^{(4)}(t) - 4!A_x.$$

On a alors par le point précédent :

$$A_x = \frac{f^{(4)}(\xi_x)}{4!}.$$

En reportant dans le point **2.a**, on obtient la formule proposée.

5.22 Enoncé de l'interrogation de décembre 2014 (1h).

SCILAB. Calculer de la manière la plus courte possible dans SCILAB :

$$\sum_{n=1}^{1000} \frac{2n + \sin(n)}{n^2 + 1}.$$

Exercice. On souhaite résoudre de manière approchée :

$$x^3 + x - 1 = 0.$$

1. Montrer que cette équation a une unique racine réelle $\bar{x}$. Montrer que $\bar{x} \in [0; 1]$. On se propose d'appliquer la méthode de Newton pour résoudre numériquement le problème, en partant de $u_0 = 1$.
2. Montrer que l'on aboutit à la suite $(u_n)_n$ définie par :

$$u_{n+1} = \frac{2u_n^3 + 1}{3u_n^2 + 1}.$$

3. On pose $\varepsilon_n = u_n - \bar{x}$.
3.a. Calculer ε_{n+1} en fonction de ε_n et de $\bar{x}$.

3.b. Montrer par récurrence que $\varepsilon_n > 0$ pour tout n.
3.c. Montrer que la suite $(u_n)_n$ est strictement décroissante.
3.d. Montrer que $u_n \to \bar{x}$.
3.e. Trouver un $C > 0$ (dépendant de $\bar{x}$ uniquement) tel que :

$$\forall n \in \mathbb{N}, \quad \varepsilon_{n+1} \leq C\varepsilon_n^2.$$

Quelle est l'interprétation de cette formule ?

5.23 Corrigé de l'exercice de l'interrogation de décembre 2014.

1. La fonction $f : x \mapsto x^3 + x - 1$ est continue et strictement croissante. Compte tenu des limites aux bornes, elle réalise donc une bijection de $\mathbb{R}$ sur lui-même, d'où l'existence et l'unicité de $\bar{x}$. On notera que $f(0) < 0 < f(1)$, donc $\bar{x} \in]0; 1[$.
2 La méthode de Newton consiste à produire l'itération :

$$u_{n+1} = u_n - \frac{f(u_n)}{f'(u_n)}.$$

Compte tenu de $f'(x) = 3x^2 + 1$, on trouve bien la formule proposée.
3.
3.a. On a :

$$\varepsilon_{n+1} = u_{n+1} - \bar{x} = \frac{(2(\varepsilon_n + \bar{x})^3 + 1) - \bar{x}(3(\varepsilon_n + \bar{x})^2 + 1)}{3u_n^2 + 1}.$$

Le développement du dénominateur, en tenant compte de $\bar{x}^3 + \bar{x} - 1 = 0$, fournit :

$$\varepsilon_{n+1} = \varepsilon_n^2 \ \frac{3\bar{x} + 2\varepsilon_n}{3(\varepsilon_n + \bar{x})^2 + 1}.$$

3.b. La formule précédente montre que lorsque $\varepsilon_n > 0$, il en est de même de ε_{n+1}. Puisque c'est vrai au rang 0, on conclut.
3.c. On procède par récurrence. On a $u_1 = 3/4$ donc $u_1 < u_0$. L'initialisation est donc bonne. Si l'hypothèse est vraie au rang n :

$$u_{n+1} - u_n = -\frac{f(u_n)}{f'(u_n)} < 0$$

puisque $f'(u_n) > 0$ et $u_n > \bar{x}$ implique $f(u_n) > 0$. Ainsi $u_{n+1} < u_n$.
3.d. La suite est décroissante minorée, elle converge donc vers un réel ℓ tel que :

$$\ell = \frac{2\ell^2 + 1}{3\ell^2 + 1}.$$

On trouve $\ell^3 + \ell - 1 = 0$, et donc $\ell = \bar{x}$.
3.e. On a $3u_n^2 + 1 \geq 3\bar{x}^2 + 1$ et

$$3\bar{x} + 2\varepsilon_n = 2u_n + \bar{x} \leq 2 + \bar{x}.$$

Ainsi, en utilisant le résultat de **3.a** :

$$\varepsilon_{n+1} \leq C\varepsilon_n^2$$

avec $C = \frac{2+\bar{x}}{3\bar{x}^2+1}$. Cette formule montre qu'approximativement et asymptotiquement, le nombre de décimales exactes est multiplié par 2 à chaque étape.