Université des Sciences et de la Technologie d'Oran – Mohamed Boudiaf

Faculté de Génie Mécanique

Département de Génie Mécanique

Analyse Numérique

Recueil d'Exercices Corrigés

Calcul et Programmation

Conformément au programme du module de Math5 de 2ème année LMD ST

Izidi Lahouari

2014

Sommaire

Analyse Numérique

Recueil d'Exercices Corrigés

Calcul et Programmation

Conformément au programme du module de Math5 de 2ème année LMD ST

Ce document propose un recueil d'exercices corrigés d'analyse numérique. Le contenu est adapté au programme du module de math5 de 2ème année LMD ST. Les techniques de calcul avec les différentes méthodes numériques sont présentées ainsi que les formulations et les algorithmes de chaque méthode. Il fournit aussi aux étudiants des programmes de calcul typiques des méthodes sous le langage de programmation Fortran. Ces programmes pourront facilement être implémentés sur ordinateur. Enfin, Des sujets d'examens sont présentés avec les corrigés type. J'espère que les étudiants trouveront dans cet ouvrage un support pour maitriser les fondements des méthodes numériques et de leur programmation.

Izidi Lahouari

Avant propos

Ce document propose un recueil d'exercices corrigés d'analyse numérique. Le contenu est adapté au programme du module de math5 de $2^{ème}$ année LMD ST. Les techniques de calcul avec les différentes méthodes numériques sont présentées ainsi que les formulations et les algorithmes de chaque méthode. Il fournit aussi aux étudiants des programmes de calcul typiques des méthodes sous le langage de programmation Fortran. Ces programmes pourront facilement être implémentés sur ordinateur. Enfin, Des sujets d'examens sont présentés avec les corrigés type. J'espère que les étudiants trouveront dans cet ouvrage un support pour maitriser les fondements des méthodes numériques et de leur programmation.

Izidi Lahouari

Analyse Numérique

Recueil d'Exercices Corrigés

Calcul et Programmation

Conformément au programme du module de Math5 de 2ème année LMD ST

Izidi Lahouari

USTO-MB 2014

Analyse Numérique

Recueil d'Exercices Corrigés

Calcul et Programmation

Conformément au programme du module de Math5 de 2ème année LMD ST

Par

Izidi Lahouari

Maître de conférences à l'USTO-MB

Analyse Numériques

1. Définition de l'analyse numérique

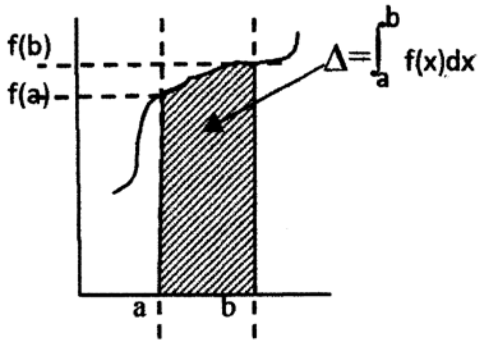
Le domaine de l'analyse numérique est une branche qui regroupe deux grands domaines de la science de l'ingénieur : mathématique et informatique.

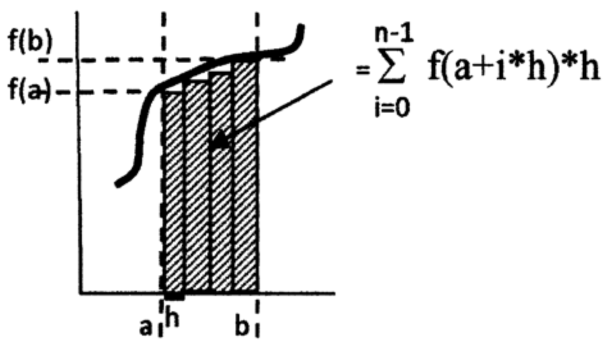
L'aspect mathématique de l'analyse numérique consiste à modéliser une solution à un problème à travers des opérateurs de l'analyse mathématique ($\partial, \int, \Pi, etc.$) ainsi que l'étude des caractéristiques analytiques de ce procédé (convergence, unicité de solution ...etc.).

L'aspect algorithmique de l'analyse numérique consiste à approximer le modèle mathématique par un autre numérique définit seulement au moyen des opérateurs arithmétiques (+, -, /, *, $\sum$, testes, répétitions ... etc.), qui peut être facilement implémenté par la suite sur un ordinateur à travers un langage de programmation.

En bref, l'objectif de l'analyse numérique consiste à trouver des algorithmes informatiques implémentant ou approximant un modèle analytique résolvant un problème scientifique donné.

Exemple : approximer l'opérateur analytique d'intégral $\int$ par un opérateur arithmétique $\sum$ c'est-à-dire approximer la surface Δ par une somme des surfaces des rectangles résultants de la discrétisation du domaine [a , b] en n points.





Une conséquence immédiate de l'approximation d'un modèle mathématique par un autre numérique, est l'écart entre la solution exacte qui résulte du modèle mathématique contre la solution approchée résultant du modèle numérique d'approximation. Cet écart est appelé erreur de troncature.

2. Définition de l'erreur de troncature

C'est l'erreur qui résulte lorsqu'on passe d'un problème continu à un problème discret, générée lorsqu'on remplace une relation exacte par une autre, plus simple ou plus facilement manipulable.

L'objectif de l'analyse numérique et de pouvoir toujours effectuer ce passage en minimisant cette erreur.

1.1. Mesures d'erreur

Soit A la solution exacte d'un problème donné et $\bar{A}$ sa solution approchée.

L'Erreur absolue est définit par A - $\bar{A}$

L'Erreur relative est définit par $\frac{A - \bar{A}}{A}$

Une autre source d'erreur provient de l'outil utilisé pour implémenter les méthodes numériques qui est l'ordinateur. L'ordinateur puissant et rapide ne peut en aucun cas représenter fidèlement l'ensemble continu R des valeurs réelles. Un nombre réel est toujours stocker sur l'ordinateur avec une certaines perte d'information. Cette perte d'information est appelée erreur d'arrondi et tend à s'amplifier avec l'arithmétique de l'ordinateur.

3. Représentation d'un nombre réel :

Un nombre réel est stocker sous une forme exponentielle normalisée $s*m*b^e$ où :

s : signe du nombre

m : mantisse

b : base de représentation

e : exposant

La mantisse et l'exposant sont toujours représentés dans la base b.

Si b = 10

Exemple :

La forme normalisée de 52,2 est $0.52 *10^2$

La forme normalisée de 0.003656 est $0.3656 *10^{-2}$

Si b = 2

Exemple :

La forme normalisée de 110.0110 est $0.110011*2^{11}$ plus exacte : $0.110011_2*10_2{}^{11}$

0.110011_2 est la mantisse

$10_2 = 2_{10}$ est la base

$11_2=3_{10}$ est l'exposant

0.000010111_2 est normalisé comme $0.10111 *10_2{}^{-100}$

3.1. Précision de l'ordinateur :

Le nombre décimal le plus petit en valeur absolue représenté par un ordinateur et lorsqu'il est additionné à 1.0 produit un résultat décimal différent de 1.0 est appelé précision de la machine et est nommé ε (epsilon machine).

3.2. Chiffres significatifs :

La précision d'une valeur se mesure par le nombre de chiffres significatifs qu'il contient.

1. Un chiffre est significatif s'il est non nul
2. Un zéro est significatif s'il est entre 2 chiffres significatifs
3. Le zéro n'est jamais significatif s'il précède les chiffres significatifs non nuls Exemple :

   1,414 => 4 chiffres significatifs, 0.000356 => 3 chiffres significatifs.

La valeur 13.2585 avec 6 chiffres significatifs est plus précise que 13.2500 avec 4 chiffres significatifs.

La normalisation consiste à traduire un nombre réel en un nombre en forme exponentielle normalisé qui ne garde que les chiffres significatifs en mantisse.

4. Erreur d'arrondi :

On peut dire que la précision d'un ordinateur est la précision avec laquelle un nombre décimal est représenté. Toute opération arithmétique sur des nombres décimaux produit une erreur d'au moins ε. Cette erreur est appelée erreur d'arrondi et tend à s'accroitre avec l'arithmétique de l'ordinateur. Le type réel simple précision conserve six chiffres significatifs non affecté par ce type d'erreur (une précision de 10^{-6}) tandis que le type double précision conserve quinze chiffres significatifs non affecté par ce type d'erreur (une précision de 10^{-15}).

Après chaque opération de calcul arithmétique élémentaire, l'ordinateur fait normaliser automatiquement le résultat intermédiaire. L'addition et la soustraction répétées fait parfois amplifier l'erreur d'arrondi.

Exemple :

Supposant une machine qui représente un nombre avec quatre chiffres significatifs et examinons l'opération simple suivante :

6 * 2/3 = (6*2)/3=12/4

Cette même opération peut être remplacée par sommer six fois le nombre 2/3 à lui-même.

- 2/3 ≈ 0.6666 en considérant quatre chiffres significatifs.
- 0.6666 + 0.6666 = 1.3332 le résultat devient 1.333 en considérant quatre chiffres significatifs.
- 1.333 + 0.6666 =1.9996 le résultat devient 1.999 en considérant quatre chiffres significatifs. 1.999 + 0.6666 =2.6656 le résultat devient 2.665 en considérant quatre chiffres significatifs.
- 2.665 + 0.6666=3.33 l 6 le résultat devient 3.331 en considérant quatre chiffres significatifs. 3.331 + 0.6666=3.9976 le résultat devient 3.997 en considérant quatre chiffres significatifs.

L'écart entre 4.000 et 3.997 = 0.003 est appelé erreur d'arrondi sur une machine en considérant quatre chiffres significatifs.

Analyse de l'erreur dans le calcul numérique

1. Représentation des valeurs dans l'ordinateur :

Les valeurs sont d'abord converties en binaire, représentées par les 0 et 1 dans la machine. Pour la représentation d'une valeur, on utilise un nombre de bit déterminé.

1.1 Représentation des entiers

Si on utilise 8 bits (1 octet) pour représenter une valeur entière : (integer*1) alors on aura :
1 bit pour représenter le signe (positif ou négatif), et 7 autres bits pour la valeur. La valeur maximal qui peut être représenté par integer*1 est ±1111111 -> ±127

Si on utilise 16 bits (2 octet) pour représenter une valeur entière : (integer*2) alors on aura :
1 bit pour représenter le signe (positif ou négatif), et 7 autres bits pour la valeur. La valeur maximal qui peut être représenté par integer*2 est ±111111111111111 -> ±32767

Exercice1 : saisissez et compilez le programme suivant, expliquer l'erreur survenue ?

```
Integer*2 X
X= 34767
Print   *,'x= ',X
END
```

1.2 Représentation des réels

Il y a 2 principes de représentation: à point fixe et à point flottant.

Point fixe : partie entière /partie décimale : Cette méthode fixe une position pour le point décimal. La partie à gauche du point est utilisée pour représenter la partie entière, et la partie droite est utilisée pour la partie décimale. Par exemple si on utilise 15 bits pour la partie entière et 16 bits pour la partie décimale, et un autre bit pour le signe, nous aurons le schéma suivant:

| 1 |2 16|17 32|
signe partie entière partie décimale

La valeur maximal d'un réel dans cette représentation est de :
$-2^{15},2^{-16}$ = 111111111111111,1111111111111111 = -32767,9999847412109375. Par exemple, la valeur de 1.5 sera représentée comme "000000000000001.1000000000000000". Cet intervalle est très limité. Les valeurs comme 33667,2569 ne peut pas être représentée dans le modèle du point fixe. (Ce modèle n'est pas utilisé de nos jours)

Point flottant

Ce schéma consiste à représenter une valeur réelle comme $\pm M * 2^E$ où $1 \leq M < 2$ est la *mantisse*, et *E l'exposant*. Par exemple, la valeur 5.5 sera transformée en $0.1011 * 2^{11}$ en code binaire ($=0.6875 * 2^3$). Dans un schéma de représentation à point flottant, la longueur de M et de E on sont fixée comme suit (où $a_1a_2\ldots$ est l'exposant et $b_1b_2\ldots$ est la mantisse):

Précision simple (32 bits)	$\| \pm \|a_1a_2\ldots a_8\|b_1b_2\ldots b_{23}\|$ (8bits pour E, 23bits pour M)
Précision double (64 bits)	$\| \pm \|a_1a_2\ldots a_{11}\|b_1b_2\ldots b_{52}\|$ (11bits pour E, 52bits pour M)

Exercice 2:

Saisissez et compilez le programme suivant (représentation interne en virgule flottante):

```
real x
read(*,*)x
print *,'x=',x
z=fraction(x)
y=exponent(x)
print *,'fraction = ',z,'  exponent= ',y
f=z*2**y
print *,'valeur de x recalculée est  = ',f
```

2. Analyse de l'erreur dans le traitement par ordinateur :

Les erreurs se produisent dans la manipulation des réels. Elles sont généralement de 3 sources : erreur d'arrondi dû à la précision de l'ordinateur, erreur

2.1. Précision de l'ordinateur :

Les ordinateurs ne sauvegardent pas les nombres avec une précision infinie. En fait, les nombres sont sauvegardés de façon approximative sous la forme de bits (binary digits) ou de bytes (groupes de 8 bits). Le problème de la sauvegarde ne se pose pas pour les nombres entiers mais pour les nombres décimaux. Le nombre décimal le plus petit (en valeur absolue) qui, lorsqu'il est additionné au nombre décimal 1.0, produit un résultat décimal différent de 1.0 s'appelle la précision de l'ordinateur ε. (Epsilon)

La fonction *epsilon(x)* donne la valeur d'epsilon dans le type de x. la valeur d'epsilon machine dans le type réel simple précision diffère de valeur d'epsilon machine dans le type réel double précision).

Exercice 3 :

Saisissez le programme suivant, compilez et exécuter le puis commenter les résultats :

```
        Real  x,z,e1,e3
        Double precision  y,e1
        e1=epsilon(x);
        e2=epsilon(f);
        print 100,'epsilon1 = ',e1,'epsilon2 = ',e2
100     format(A11,f32.30,A11,f32.30)
        End
```

2.2. Erreurs et arithmétique de l’ordinateur

2.2.1. Erreur d’arrondi (dû à l’arithmétique de l’ordinateur)

On peut dire que la précision de l’ordinateur est la précision avec laquelle les nombres décimaux sont représentés. En fait, toute opération arithmétique sur des nombres décimaux introduit une erreur supplémentaire d’au moins ε. On appelle ce type d’erreur erreur d’arrondi. L’erreur d’arrondi provient de l’ordinateur.

Exercice 4

Ajoutez ces lignes au programme puis recompilé et commentez les résultats :

```
     X=2356874958663
     Print  10,x
10   format(f30.3)     ! que remarquez vous ?
     z=1.0+e1
     print *,'z =',z , ‘ n’’est pas négligeable ’

     x=1.0+0.00000001
     y=1.0D0 + 0.00000001D0  !     0.00000001< e1 pour simple précision
                             !   mais  0.00000001>e2 pour double précision
     print *,'x =',x , ‘ est négligeable par contre y= ’,y, ‘  est pas négligeable ’
     end
```

On peut connaître le nombre de chiffres significatifs exacts (non affectés par l'erreur d'arrondi de l'ordinateur) sur l'ordinateur pour le type réel simple et le type réel double précision par la fonction précision.

Exercice 5

déterminer par le programme suivant le nombre de chiffres significatifs exactes du réel simple et du réel double

```
      Real x
      Double precision y
      i =precision(x)
      j =precision(y)
      Print  11,i,j
 11   format(I3)     ! que remarquez vous ?
      End
```

2.2.2. Erreur de perte d'information

Ces erreurs peuvent être générées lorsqu'on ne fait pas attention à la conversion que l'ordinateur effectue automatiquement entre les types numériques.

Exercice 6 :

Saisissez le programme suivant, par partie, compilez et exécutez le puis commenter les résultats de chaque partie :

```
      real*4 r1,t
      double precision x,y,z,f1,ff1,f2,ff2,err1,err2
c     première partie : limite de représentation de nombre réel simple précision
      print *,'erreur : nombre réel simple -> double précision '
      r1=365896523145.0           ! conversion automatique
      x=365896523145.0D0     ! conversion explicite
      err1=abs(x-r1)
      print 5,'val réel',r1,'val double',x,'err de représentation',err1
 5    format(3(A22,f30.3,/))
      print *,'erreur : conversion implicite de 5 et 3 dans 5/3 '
      x=5/3
      y=5.0/3.0
      z=dfloat(5)/dfloat(3)
      f1=abs(x-z)
      ff1=abs(y-z)
      print 10,'x=',x,'y=',y,'z=',z,'f1=x-z=',f1,'ff1=y-z=',ff1
c     deuxième partie : propagation d'erreur de conversion automatique dans le calcul
      print *,'propagation de err : de la division à la soustraction'
      x=5/3-5
      y=5.0/3.0-5.0
      z=dfloat(5)/dfloat(3)-dfloat(5)
      f2=abs(x-z)      ! f2>f1 erreur amplifiée
      ff2=abs(y-z)     ! ff2>ff1 erreur amplifiée
      print 10,'x=',x,'y=',y,'z=',z,'f2=x-z=',f2,'ff2=y-z=',ff2
 10   format (/,5(A10,f30.27,/),/)
c  troisième partie : analyse de l'erreur amplifiée
      print *, 'terme de l''erreur amplifiée   '
```

```
      err1=abs(f2-f1)
      err2=abs(ff2-ff1)
      print 20,'err1=f2-f1',err1,'err2=ff2-ff1',err2
20    format (/,2(A13,f30.27,/),/) ! // veut dire sauter deux lignes
      End
```

2.2.3. Erreur de troncature (produite par la méthode de résolution approchée)

L'erreur d'arrondi provient de l'ordinateur. L'autre erreur principale du calcul numérique vient de la discrétisation (passer d'un problème continue à un problème discret) est générée lorsque l'on remplace une relation "exacte" par une autre, plus simple ou plus facilement manipulable. C'est par exemple le cas :

De l'approximation d'une dérivée par une différence finie comme $f'(x) \approx \frac{f(x+h)-f(x)}{h}$

Ou, d'une série de Taylor tronquée comme $e^x \approx 1 + x^2$

L'écart entre la réponse exacte (e^x) et la réponse numérique $(1 + x^2$) s'appelle l'erreur de troncature. L'erreur de troncature persisterait même sur un ordinateur "parfait" sans erreur d'arrondi. L'erreur de troncature est quant à elle sous la responsabilité de celui qui programme. On peut dire sans trop d'exagération qu'une minimisation astucieuse de l'erreur de troncature est l'objet même de l'analyse numérique.

Exercice 7:

Saisissez, compilez et exécuter le programme suivant : calcul exacte et numérique e^x, commentez le résultat. Calculer l'erreur de troncature.

```
      real  x,expo,y
      print *,'donnez la valeur de x (proche de 0) '
      read(*,*)x
      y=exp(x)
      expo=1.0+x**2
      print 10,'valeur exacte =',y,'  valeur numérique =',expo
10    format (A17,f30.23,A22,f30.23)
      end
```

2.2.4. Instabilité d'une méthode numérique

En règle générale il n'y a pas grand-chose que l'on puisse faire pour éviter les erreurs d'arrondi, sinon d'éviter d'utiliser des *algorithmes qui amplifient les erreurs d'arrondi*. La plupart du temps, l'erreur d'arrondi et l'erreur de troncature n'interagissent pas fortement entre elles. Cependant une méthode qui apparaît simple et logique peut être instable et introduit des erreurs. Ceci veut dire qu'une erreur d'arrondi présente au début des calculs peut être amplifiée par la méthode numérique utilisée de telle sorte que la réponse numérique devient rapidement fausse. Il faut toujours étudier et critiquer la stabilité d'une méthode numérique.

Etude de cas : calcul du nombre d'or

Un exemple d'algorithme instable est donné par l'algorithme suivant: supposons que l'on veuille calculer les puissances du nombre d'or, le nombre donné par :

$$\varphi = \frac{\sqrt{5}-1}{2} \approx 0.6180339887$$

Les puissances de φ satisfont la relation de récurrence $\varphi^{n+1} = \varphi^{n-1} - \varphi^{n}$.

Par conséquent, si l'on connaît les deux premières puissances $\varphi^0 = 1$ et $\varphi^1 = 0.6180339887$, on peut calculer toutes les puissances de φ uniquement avec des soustractions.

Saisissez le programme suivant, exécutez-le et commentez les résultats. Ce programme permet de calculer les puissances de φ par la méthode exacte et par la méthode numérique donnée par la relation de récurrence. Calculer en suite l'erreur absolue (valeur exacte – valeur approchée) et l'erreur relative (erreur absolue / valeur exacte).

```
      double precision or_exact,or_approche
      double precision or_0,or_1,or_actuel,or_prochain,or_anterieur
```

c lecture de l'ordre de puissance

```
      print *,'donnez la puissance '
      read(*,*) n
```

c calcul par la méthode exacte

```
      or_exact=((sqrt(5.0D0)-1.0D0)/2.0D0)**n
      print *, 'la valeur exacte =',or_exact
```

c calcul par la méthode de récurrence

```
      or_0=1.0D0
      or_1= 0.6180339887D0

      or_actuel=or_1
```

```
or_anterieur=or_0

do i=2,n          ! commencer par la puiss 2 parce que les puiss 0,1
                  ! sont déja calculées

or_prochain=or_anterieur-or_actuel;

or_anterieur=or_actuel
or_actuel=or_prochain

enddo

or_approche=or_prochain
print *,' la valeur approchée =',or_approche
end
```

On peut remarquer que la méthode de calcul par différence est instable. Dès la puissance 18, l'écart entre la valeur exacte et approchée devient grand. Cela revient à la soustraction des nombres φ^{n+1} et φ^{n-1} qui deviennent sensiblement proches quand n est grand. On dit que la récurrence est instable. De nombreuses méthodes numériques qu'on va étudier sont itératives (récurrentes), donc source d'erreur. Pour ce type de méthode, deux questions essentielles se posent:

1. Le processus converge-t-il ?
2. Quand s'arrête-t-on pour obtenir une réponse plus ou moins exacte ?

La réponse à ces deux questions n'est pas toujours facile et demeure l'une des préoccupations essentielles du numéricien. Et la préoccupation majeure de ce cours (méthodes numériques appliquées).

Recherche des zéros d'une fonction à une variable

Exercice 1 :

Soit la fonction $f\left(x\right)=e^{x}-x-2$

1. Séparer graphiquement les racines de *f*
2. Déterminer graphiquement deux intervalles d'existence des racines.
3. Résolution de *f(x)=0* par la méthode de dichotomie :
    a. Existe-t-il de racine sur l'intervalle [0,1.5]
    b. pour atteindre une précision ε=0.016, combien faut-il d'itérations n pour trouver la racine sur l'intervalle [0,1.5]
    c. calculer la racine pour l'intervalle [0,1.5] et ε=0.016.
4. Résolution de *f(x)=0* par la méthode de newton :
    a- quelle est la condition suffisante pour la convergence de la méthode de newton ?
    b- est- elle vérifiée pour x_0=1 ? sinon peut-on trancher sur la convergence ?
    c- trouver la racine pour x_0=1, pour ε=0.016
5. Comparer le résultat des deux méthodes.
6. Refaire les questions 3, 4 et 5 pour l'intervalle [1,2] pour ε= 0.0003, x_0=1.5, utiliser le critère d'arrêt pour la méthode de newton $|x_i - x_{i-1}| / |x_i|$

Exercice 2 :

Ecrire le programme fortran qui résout l'équation $f(x)=xe^{x}+3e^{x}-1=0$ par la méthode de dichotomie (bissection).

1. Il faut lire la précision epsilon.
2. Il faut proposer l'intervalle initial [a, b] par une lecture de a et b et tester si elles sont bonnes ou non.
3. Programmer la méthode en affichant la solution à la fin.
4. Utiliser un sous programme du type function pour définir la fonction *f(x)*.
5. Application : prendre la précision 10^{-3} et l'intervalle initial [0, 2].

Exercice 3 :

Ecrire le programme fortran qui résout l'équation $f(x) = \cos x - x^3 = 0$ par la méthode de Newton.

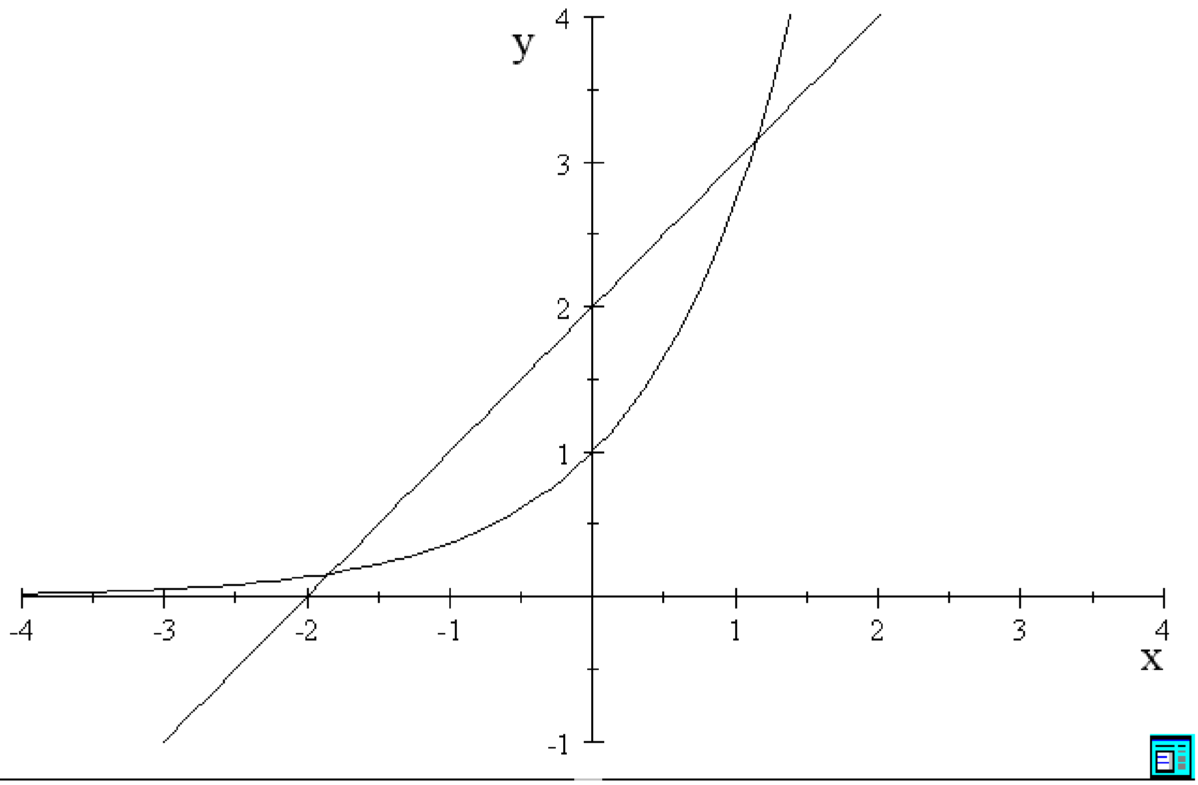
1. Il faut lire la précision epsilon.
2. Il faut lire la valeur initiale x_0
3. Programmer la méthode en affichant le nombre d'itérations et la solution à la fin.
4. Utiliser un sous programme du type function pour définir la fonction $f(x)$

5- Application : prendre la précision de 10^{-7} et $x_0 = 0$.

Corrigé de l'exercice 1

1- Soit f(x) = $f(x) = e^x - x - 2 = 0$ qui peut s'écrire : $f(x) = e^x - (x+2) = 0$ où $g(x) = e^x$ et $h(x) = x + 2$

$f(x) = g(x) - h(x)$ =0 d'où $g(x) = h(x)$ les racines de *f* sont tous les points x d'intersection de la courbe de $g(x)$ avec celle de $h(x)$



Dichotomie :

1- vérification sur [0,1.5] : *f*(a)=*f*(0)= -1

f(b)=*f*(1.5)=0.9816 . *f*(a)**f*(b)<0 ∃ au moins une racine.

2- pour atteindre une précision ε=0.016, une solution x_n à l'itération n est supposée racine approchée si $\frac{(b0-a0)}{2^{n+1}} \le \varepsilon \Rightarrow \quad n \ge \frac{\ln\left(\frac{b0-a0}{2\varepsilon}\right)}{ln 2}$

$$n_{min} = \frac{\ln\left(\frac{b0-a0}{2\varepsilon}\right)}{ln 2} = 5.55 \approx 6 \text{ itérations}$$

i	a_i	b_i	x_i	$f(x_i)$	ε_i
0	0	1.5	0.75	-0.633	0.75
1	0.75	1.5	1.125	-0.00447	0.375
2	1.125	1.5	1.3125	0.4029	0.1875
3	1.125	1.3125	1.21875	0.16420	0.09375
4	1.125	1.21875	1.1718	0.0559	0.0468
5	1.125	1.1718	1.14843	0.0048	0.0234
6	1.125	1.14843	1.13671	-0.02	0.01171

Tout l'intervalle final [1.13671,1.14843] est solution du problème. On peut choisir la milieu de cet intervalle comme solution, donc la solution est : *x=(a+b)/2=1.1425*

Algorithme de la méthode de dichotomie (bissection):

1. $i = 0$
2. $a_i = 0$; $b_i = 1.5$
3. $Calculer\ x_i = (a_i + b_i)/2\ \ calculer\ f(x_i)$
4. $si\ f(a).f(xi) < 0\ alors$
   $b_i = x_i\ et\ a\ reste\ le\ $même
   sinon
   $a_i = x_i\ \ et\ b\ reste\ le\ $même
   $fin\ si$
5. $Si\ abs(a_i - b_i) \geq \varepsilon\ \ i = i + 1$
   $aller$ à 3 , $sol = (b_n + a_n)/2$
6. fin

Méthode de Newton :

1-Si les conditions suivantes sont vérifiées :

$f'(x_0)$ et $f''(x_0)$ existent et $f(x_0)*f''(x_0)>0$ cela et suffisant pour dire que la méthode de newton converge vers une racine.

2- pour $f(x) = e^x - x - 2 = 0$ $f'(x) = e^x - 1$ existe, $f''(x) = e^x$ existe

Pour $x_0=1$: $f(1)*f''(1) = (e^1 - 1 - 2)\ e^1 = -0.76579 < 0$ condition non vérifiée. On ne peut rien dire sur la convergence de la méthode pour $x_0=1$ car la condition est suffisante mais non nécessaire.

3- $f(x) = e^x - x - 2$ $f'(x) = e^x - 1$ la formule de newton s'écrit

$$x_n = x_{n-1} - \frac{e^{x_{n-1}} - x_{n-1} - 2}{e^{x_{n-1}} - 1}$$

i	x_i	$\lvert x_i - x_{i-1} \rvert$
0	0	
1	1.16395	0.16
2	1.14642	0.017
3	1.146193	0.00033

Algorithme de la méthode de Newton :

1. $x_0 = 1$
2. $i = 1$
3. $x_i = x_{i-1} - \frac{(e^{x_{i-1}} - x_{i-1} - 2)}{(e^{x_{i-1}} - 1)}$
4. $si\ \ |x_i - x_{i-1}| \geq \varepsilon\ alors\ x_{i-1} = x_i$
   $i = i + 1\quad aller$ à 3
   $si\ non$
5. fin

Comparaison entre la méthode de dichotomie et la méthode de Newton :

La méthode de newton est plus rapide (convergence quadratique) que celle de la dichotomie (convergence linéaire) elle est aussi plus précise

Corrigé de l'exercice 2

```
                  Program DICHOTOMIE BISSECTION
      EPS=0.001
      write(*,*) ' donner le domaine de la racine'
      read(*,*) a, b
         if(f(a)*f(b).lt.0.0) then
           write(*,*) 'bonnes valeurs initiales'
           else
           write(*,*) 'changer les valeurs '
           goto 20
         endif
10    xm=(a+b)/2.
         if((f(xm)*f(a).lt.0.0) then
           b=xm
           else
           a=xm
         endif
      err=abs(a-b)
         if(err.le.eps) then
           xm=(a+b)/2.
           write(*,*) 'La solution est :',xm
           else
           goto 10
         endif
20    end
      function f(x)
        f=x*exp(x)+3*exp(x)-1
        return
      end
```

Corrigé de l'exercice 3

```
              Program NEWTON RAPHSON
       EPS=1e-7
       write(*,*) 'donner la valeur initiale x0'
       read(*,*) x0
       i=1
       x=x0
10     continue
       xi=x-(f(x)/ff(x))
       err=abs(xi-x)
         if (err.ge.eps) then
           write(*,*) i,xi
           x=xi
           i=i+1
           goto 10
           else
           write(*,*) 'solution converge'
           write(*,*) i,xi
         endif
       end
       function f(x)
         f=cos(x)-x**3
         return
       end
       function ff(x)
         ff=-sin(x)-3*x**2
         return
       end
```

Dérivation et Intégration Numérique

Exercice 1 :

Soit la fonction $f(x) = x^3 - 2e^x + 1$:

1- Calculer les valeurs de la dérivée de f aux points : -5, -2, 0.25, 2.25, 3.5, 4, pour h=0.0015 en utilisant :

a- La méthode des différences excentrées en avant.

b- La méthode des différences excentrées en arrière.

c- La méthode des différences centrées.

2- Comparer les résultats avec les valeurs obtenues par la formule analytique de la fonction dérivée *f'*, commenter (critère de comparaison : erreur absolue en moyenne).

3- Calculer la valeur de la dérivée de *f* au point 3.5 par les différences centrées pour les différentes valeurs de h : 0.00015, 0.0015, 0.015, 0.5.

4- Comparer les résultats avec la valeur f' (3.5) obtenue par la formule analytique de la fonction dérivée *f'*, commenter.

5- Calculer *f''* au point -3 avec h=0.015, comparer le résultat avec celui obtenu en utilisant la formule analytique de *f''*.

Exercice 2 :

Soit à déterminer l'intégrale de $f(x) = \frac{1}{x}$ sur l'intervalle [1,4] en utilisant :

a- Méthode des trapèzes, n=8.

b- Méthode de Simpson, n=8.

c- Formule analytique Ln (logarithme).

d- Comparer les résultats sur la base de l'erreur de troncature absolue.

e- Ecrire les algorithmes des méthodes : trapèze et Simpson

Exercice 3 :

Soit la fonction *f* définit par l'ensemble de points $\{x_i, y_i\}$:

X	-0.3	0	0.3	0.6	0.9	1.2	1.5	1.8	2.1	2.4	2.7
Y	0.373	1.0	1.627	2.416	3.529	5.128	7.375	10.433	14.461	19.624	26.083

1- Calculer les valeurs de la dérivée de *f* dans tous les points Xi par la méthode des différences centrée. Utiliser le schéma excentré (en avant ou en arrière) dans les bornes de l'intervalle de définition de *f*.

2- sur la base du tableau résultant (valeurs de f' sur l'intervalle [-0.3, 2.7]), calculer, par la méthode de trapèze, les intégrales suivantes :

$$\int_0^{1.5} f'(x) \quad ; \quad \int_0^{2.1} f'(x) \quad ; \quad \int_{0.6}^{1.8} f'(x) \quad ; \quad \int_{1.8}^{2.7} f'(x)$$

3- comparer les résultats obtenus en 2 par les valeurs exactes des différents intégrales sachant que $\int_a^b f'(x)$ = *f(b)-f(a)*, en utilisant comme critère.

- En termes d'erreur absolue entre la valeur exacte et la valeur approchée.
- Erreur obtenue dans chaque trapèze (expliquer pourquoi l'erreur est relativement importante dans le calcul du dernier intégral).

Corrigé de l'exercice 1 :

$f(x) = x^3 - 2e^x + 1$ Approximer une dérivée par une différence finie :

1-Schéma excentré avant : $f'(x0) = \frac{f(x0+h)-f(x0)}{h}$

2-Schéma excentré arrière : $f'(x0) = \frac{f(x0)-f(x0-h)}{h}$

3-Schéma centré $f'(x0) = \frac{f(x0+h)-f(x0-h)}{2h}$

Dérivée analytique : $f'(x) = 3x^2 - 2e^x$

1/ calculer f'(x0) dans les différents points pour h=0.0015 par les différents schémas :

X	-5.0	-2.0	0.25	2.25	3.5	4.0
Schéma ex_avant	74.9664	11.7203	-2.3813	-3.7921	-29.5130	-61.2640
Erreur absolue	0.020	0.0089	0.00077	0.0041	0.0321	0.0677
Schéma ex_arri	75.0172	11.7378	-2.3797	-3.7835	-29.4456	-61.1292
Erreur absolue	0.030	0.0084	0.00081	0.0044	0.0352	0.0670
Schéma centré	74.9918	11.7290	-2.38053	-3.7878	-29.4793	-61.1966
Erreur absolue	0.0053	0.0002	0.00002	0.00013	0.0015	0.0003
F' analytique	74.9865	11.7293	-2.38055	-3.7879	-29.4809	-61.1963

2/ On remarque que le schéma centré est celui qui donne une meilleur approximation de la dérivée : quantité d'erreur absolue minimale par rapport aux 2 autres schéma.

En moyenne : erreur absolue moyenne pour les 3 schéma sur l'échantillon de 6 points =

Erreur absolue moyenne par le schéma excentré avant = 0.0223

Erreur absolue moyenne par le schéma excentré arrière = 0.0244

Erreur absolue moyenne par le schéma centré = 0.0012

3/ Calcul de $f'(3.5)$ en utilisant le schéma centré pour les différents valeurs de h :

$f'_{analytique}(3.5) = -29.4809$

Schéma centré	-29.4685	-29.4793	-29.4834	-29.7070	-32.025
h	0.00015	0.0015	0.015	0.15	0.5
Erreur absolue	0.0123	0.00155	0.0025	0.2261	2.544

4/ L'écart entre la valeur exacte et la valeur approchée de la dérivée dans un point donnée tend à s'accroitre quant h est trop grand ou même relativement trop petit. La valeur de h doit être bien ajustée car elle influe sur la qualité du résultat.

5/ $f''(x0) = \frac{f(x0-h)-2f(x0)+f(x0+h)}{h^2}$; $f''_{analytique} = 6x - 2e^x$

Pour h=0.015 et x_0=3.0 : $f''_{analytique}$ $(3.0) = -18.0986$; $f''_{approc\ hée}$ $(3.0) = -18.0995$; err_abs=0.00095.

L'approximation de la dérivée par des différences (centrées ou excentrées) est trop sensible d'une part au paramètre de discrétisation h et d'autre part à l'allure de la fonction elle-même. Si l'écart entre f(x0) et la valeur de f dans les points adjacents x0+h ou x0-h (f(x0+h) et f(x0-h)) est important, les 3 schémas risquent d'engendrer des erreurs d'approximation importantes. Il faut adapter le h selon cet écart.

<u>Corrigé de l'exercice 2 :</u>

Méthode exacte :

En sachant la formule analytique de la fonction qui résulte de l'intégral de *1/x* qui est *ln*

$\int_1^4 \frac{1}{x} = \ln(4) - \ln(1) = 1.38629$

Formule de la méthode des Trapèzes :

$\int_a^b f(x) = \sum_{i=0}^{n-1} \frac{X_{i+1}-\mathrm{X_i}}{2} (f(\mathrm{X_i}) + f(\mathrm{X_{i+1}}))$

Pour i=0 : x_i=a ; pour i=n-1 : x_{i+1}=b c'est les borne elles sont présentent une fois dans la somme ; chaque xi (hors a et b) est présent 2 fois (une fois sous la forme de x_i et une fois sous la forme de x_{i+1} ce qui justifie que le facteur ½ ne ce multiplie qu'avec a et b)

$\int_a^b f(x) = \frac{h}{2}(f(a) + f(b)) + h\sum_{i=1}^{n-1} f(xi)$ h=x_{i+1} – x_i =(b-a) / n avec n nombre de trapèzes et h donc est la largeur de chaque trapèze :

$$\int_a^b f(x) = \frac{h}{2}\big(f(a) + f(b)\big) + h\sum_{i=1}^{n-1} f(a + i * h)$$

Par application numérique : f(x)=1/x ; [a,b]=[1,4] ; n=8 ⇒h=(4-1)/8=0.375

$\int_1^4 1/x = \frac{0.375}{2}\big(f(1) + f(4)\big) + 0.375\sum_{i=1}^{n-1} f(1 + i * h)$=1.3971

Erreur absolue = |$1.38629 - 1.397126$| =0.0108

Formule de la méthode de Simpson

$\int_a^b f(x) = \frac{h}{3}\sum_{i=0}^{n-2}(f(x_i) + 4f(x_{i+1}) + f(x_{i+2}))$ i vari avec un pas de 2

Par application directe :

$$\int_1^4 f(x) = \frac{0.375}{3}\sum_{i=0,2}^{8-2}\left(\frac{1}{Xi} + 4\frac{1}{x_{i+1}} + \frac{1}{x_{i+2}}\right)$$

$=\frac{0.375}{3}$ (f(x0)+4f(x1)+f(x2) +f(x2)+4f(x3)+f(x4) +f(x4)+4f(x5)+f(x6) +f(x6)+4f(x7)+f(x8))

$=\frac{0.375}{3}$(f(a) + 4*(f(x1)+f(x3)+f(x5)+f(x7) + 2 *(f(x2)+f(x4)+f(x6)) + f(b))

$=\frac{0.375}{3}$(f(a)+f(b)+ 4* $\sum_{i=1,2}^{n-1} f(xi)$ + 2*$\sum_{i=2,2}^{n-2} f(xi)$) =1.386805

Erreur absolue = $|1.38629 - 1.386805|$ = 0.000510

Meilleur approximation donnée par la méthode de Simpson

Algoritheme de la méthode des Trapèzes

```
Début

      Lire (a,b,n)
      h←(b-a)/n
      s←0

      pour i←1,n-1,1

            s←s+f(a+i*h)

      finpour

      s←s*h + (f(a)+f(b))*(h/2)
      ecrire(s)
fin
```

Algorithme de la méthode de Simpson

```
Début
        Lire (a,b,n)
        h←(b-a)/n
        s_pair←0
        s_imp←0
s←0

        pour i←1,n-1,2
                s_imp←s_imp+f(a+i*h)
        finpour
        pour i←2,n-2,2
                s_pair←s_pair+f(a+i*h)
        finpour


        s←(f(a)+f(b) + 4*s_imp+ 2*s_pair)*(h/3)
        ecrire(s)
fin
```

Corrigé de l'exercice 3

1- h=0.3

Schéma centré $f'(x0) = \frac{f(x0+h)-f(x0-h)}{2h}$

Pour x=-0.3 en utilise le schéma **excentré en avant**

Pour x=2.7 en utilise le schéma **excentré en arrière**

X	-0.3	0	0.3	0.6	0.9	1.2	1.5	1.8	2.1	2.4	2.7
Y	2.1	2.09	2.36	3.17	4.52	6.41	8.839	11.81	15.32	19.37	21.54

$\int_0^{1.5} f'(x) =$ exacte : f(1.5)-f(0) = 6.375

Trapèze =6.57735

Erreur absolue =**0.20235**

Il existe **5** trapèzes :

erreur par trapèze = 0.20235/**5** = **0.040**

$\int_0^{2.1} f'(x) =$ exacte : f(2.1)-f(0) = 13.461

Trapèze =13.7442

Erreur absolue =**0.2832**

Il existe **7** trapèzes

erreur par trapèze = 0.20235/**7**= **0.040**

$\int_{0.6}^{1.8} f'(x)$ = exacte : f(1.8)-f(0.6) = 8.016

Trapèze =8.1777

Erreur absolue =**0.16**

Il existe **4** trapèzes

erreur par trapèze = 0.20235/**7**= **0.040**

$\int_{1.8}^{2.7} f'(x)$ = exacte : f(2.7)-f(1.8) = 15.651

Trapèze =15.4095

Erreur absolue =**0.2415**

Il existe **3** trapèzes

erreur par trapèze = 0.20235/**3**= **0.08**

Conclusion

On remarque que l'erreur d'approximation est stable (0.04) dans chaque trapèze cela est logique puisque *h*=0.3 est fixe. Néanmoins dans le calcul du dernier intégral l'erreur par trapèze est relativement doublée. Cela est dû à l'erreur dans le calcul initial de *f'*(2.7) calculé par le schéma des différences excentrées en arrière et pas le schéma centré(effet de bord).

Programmes de calcul:

Les programmes sont écrits en langage de programmation Fortran.

Programme de dérivation numérique :

```
                         Program derivation

        write(*,*) 'donner la valeur de x'
        read(*,*) x

        write(*,*) 'donner le pas h'
        read(*,*) h

        ana1=3*x**2-2*exp(x)
        davant=(f(x+h)-f(x))/h
        darrir=(f(x)-f(x-h))/h
        dcentr=(f(x+h)-f(x-h))/(2*h)

        erravant=abs(davant-ana1)
        errarrir=abs(darrir-ana1)
        errcentr=abs(dcentr-ana1)

        write (*,*)  '        ana       avant       arriere      centree '
        write (*,*)            ana1,     davant,    darrir,    dcentr
        write (*,*)                          erravant , errarrir , errcentr

        ana2=6*x-2*exp(x)
        dsecon=(f(x-h)-2*f(x)+f(x+h))/h**2
        errsecon=abs(dsecon-ana2)

        write (*,*)  '          ana       seconde        erreur     '
        write (*,*)            ana2 ,     dsecon ,    errsecon
         end

        function f(x)
        f=x**3-2*exp(x)+1
        return
        end
```

Programmes d'intégration numérique par la méthode des trapèzes et la Méthode de Simpson :

```
                        Program integration

      write(*,*) ' donner les bornes de l integrale'
      read(*,*) a,b

      write(*,*) ' donner le nombre de segments'
      read(*,*) n

      h=(b-a)/n


c   ************************************
c        calcul par la methode des trapèzes
c   ************************************

      TRA=h*(f(a)+f(b))/2

      do i=1,n-1
      TRA=TRA+(h*f(a+i*h))
      enddo
c     write(*,*) 'methode du trapeze', TRA



c   ************************************
c        calcul par la methode de Simpson
c   ************************************


      S0=f(a)+f(b)

      S1=0.
      do i=1,n-1,2
      S1=S1+f(a+i*h)
      enddo

        S2=0.
      do i=2,n-2,2
      S2=S2+f(a+i*h)
      enddo

      SIM=(h/3.)*(S0+4*S1+2*S2)

c     write(*,*) 'methode de Simpson ', SIM


c   ************************************
c        calcul par la methode analytique
c   ************************************

       ANA=alog(b)-alog(a)

c     write(*,*) 'solution analytique ', ANA



c   ************************************
c        calcul d erreur
c   ************************************

      err1=abs(ana-TRA)
      err2=abs(ana-SIM)

      write(*,*) 'solution Trapeze    ',    TRA, err1
        write(*,*) 'solution Simpson    ',    SIM, err2

      write(*,*) 'program termined'
      END

      function f(x)
      f=1/x
      return
      end
```

Interpolation Numérique

Exercice 1 :

Soit la fonction $f(x)$ =ln(x).

1. Déterminer le polynôme d'interpolation de Lagrange qui interpole f par les deux points d'appui d'abscisses : 0.5, 0.7. Quelle est l'ordre de cette interpolation
2. Calculer $f(0.6)$. Comparer le résultat obtenue avec la valeur exacte de ln(0.6).
3. Peut-on utiliser ce polynôme pour calculer $f(2)$? Pourquoi ?
4. Déterminer le polynôme d'interpolation de Lagrange qui interpole f par 3 points d'appui d'abscisses 0.4, 0.5, 0.7. Quelle est l'ordre de cette interpolation ?
5. Calculer $f(0.6)$. Comparer le résultat obtenue à la valeur exact ln(0.6) puis au résultat obtenue par l'interpolation précédente

Exercice 2 :

Avec quelle précision peut-on calculer la valeur $\sqrt{115}$ à l'aide de l'interpolation de Lagrange qui repose sur les points d'appui d'abscisses : 100, 121, 144

Exercice 3 :

Construire le polynôme d'interpolation de Lagrange de la fonction f donnée par les tableaux suivants :

x	0	1	3	4
f(x)	-1	1	3	5

x	-1	0	2	3	5
f(x)	0	1	0	-1	0

x	-2	-1	0	1	2	3
f(x)	-3	-2	-2	0	1	2

Corrigé de l'exercice 1 :

Rappelons que le polynôme de Lagrange basé sur les points d'appui d'abscisses X_0, X_1, X_2, X_n est d'ordre **n** et il s'écrit :

$$P_n(x) = \sum_{i=0}^{n} f(x_i) L_i(x)$$

avec

$$L_i(x) = \prod_{j=0, j\neq i}^{n} \frac{(x-x_j)}{(x_i-x_j)}$$

1- *f(x)*=ln(x) ; on cherche le polynôme d'interpolation de f avec 2 points d'appui : $(x_0, f(x_0))$= (0.5,-0.693) ; $(x_1, f(x_1))$ = (0.7,-0.357)

$L_0(x) = \frac{(x-x_1)}{(x_0-x_1)} = \frac{(x-0.7)}{(0.5-0.7)} = -5x + \frac{7}{2}$

$L_1(x) = \frac{(x-x_0)}{(x_1-x_0)} = \frac{(x-0.5)}{(0.7-0.5)} = 5x - \frac{5}{2}$

$$f(x) \cong P_1(x) = f(x_0)\, L_0(x) + f(x_1)\, L_1(x)$$
$$= -0.693\, L_0(x) - 0.357\, L_1(x)$$

$$\boldsymbol{f(x) \cong P_1(x) = 1.68x - 1.5325}$$

n+1=2 $\Rightarrow$ n=1 $\Rightarrow$ Cette interpolation est d'ordre 1 elle est dite linéaire.

2- $\text{Ln}(0.6)_{\text{approché}} = f(0.6) \cong P_1(0.6) = 1.68 * 0.6 - 1.5325 = -0.5245$

$\text{Ln}(0.6)_{\text{exacte}}$ = - 0.5108

Err_abs = 0.0137

3- On ne peut pas utiliser ce polynôme pour calculer f(2) car le contexte de l'interpolation exige que l'approximation de f dans tout point d'abscisse *x* par un polynôme est valide si et seulement si $x \in [x_0 , x_n]$ et dans ce cas $2 \notin [0.5,0.7]$ (dans le cas où $x < x_0$ ou $x > x_n$ c'est une extrapolation).

4- On cherche le polynôme d'interpolation de f avec 3 points d'appui : $(x_0, f(x_0))$= (0.4,-0.916) ; $(x_1, f(x_1))$ = (0.5,-0.693) ; $(x_2, f(x_2))$ = (0.7,-0.357) cette interpolation est dite quadratique

$L_0(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} = \frac{(x-0.5)(x-0.7)}{(0.4-0.5)(0.4-0.7)}$

$L_1(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} = \frac{(x-0.4)(x-0.7)}{(0.5-0.4)(0.5-0.7)}$

$\mathbf{L_0(x)}=\frac{100}{3}\boldsymbol{x}^2-\frac{120}{3}\boldsymbol{x}+\frac{35}{3}$

$\mathbf{L_1(x)}=-\frac{100}{2}\boldsymbol{x}^2+\frac{110}{32}\boldsymbol{x}-\frac{28}{32}$

$L_2(x) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} = \frac{(x-0.4)(x-0.5)}{(0.7-0.4)(0.7-0.5)}$

$\mathbf{L_2(x)}=\frac{50}{3}\boldsymbol{x}^2-\mathbf{15}\boldsymbol{x}+\frac{10}{3}$

$\boldsymbol{f(x)\cong P_2(x)=-0.916L_0(x)-0.693L_1(x)-0.357L_2}(\mathbf{x})$

$\boldsymbol{f(x)\cong P_2(x)=-1.813x^2+3.865x-2.17}$

5- $Ln(0.6)_{approché} = f(0.6)\cong P_2(0.6) = -1.813*0.6^2+3.865*0.6-2.17 = -0.50368$

$Ln(0.6)_{exacte}$ = - 0.5108

Err_abs = 0.00714

6- Plus le nombre de points d'appui augmente est plus l'interpolation est meilleure.

<u>Corrigé de l'exercice 2 :</u>

On veut calculer $\sqrt{115}$ par interpolation et le comparer avec sa valeur exacte : l'écart entre la valeur approchée et la valeur exacte représente la précision de l'interpolation.

$f(x) = \sqrt{x}$; points d'appui $(x_0, f(x_0))$ = (100,10); $(x_1, f(x_1))$=(121,11) ; $(x_2, f(x_2))$=(144,12) ; n+1=3 $\Rightarrow$ interpolation d'ordre 2 (quadratique)

On nous demande pas de déterminer le polynôme mais plutôt calcul directe de $\sqrt{115}$ par application numérique dans les différents Li et par la suite la valeur final de $\sqrt{115}$

$L_0(115) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} = \frac{(115-121)(115-144)}{(100-121)(100-144)} = 0.18831$

$L_1(115) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} = \frac{(115-100)(115-144)}{(121-100)(121-144)} = 0.90062$

$L_2(115) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} = \frac{(115-100)(115-121)}{(144-100)(144-121)} = -8.8933 \times 10^{-2}$

$\sqrt{115} = \boldsymbol{f(115) \cong P_2(115) = 10 * L_1(115) + 11 * L_1(115) + 12 * L_2(115)}$

$\sqrt{115} = \boldsymbol{f(115) \cong P_2(115) = 10 * 0.18831 + 11 * 0.90062 + 12 * -8.8933} \times 10^{-2}$

$\sqrt{115}_{\text{approchée}} = 10.72276$

$\sqrt{115}_{\text{exacte}} = 10.72381$

On peut obtenir une Précision = 1.0499×10^{-3} avec une interpolation quadratique basée sur les 3 points d'appui : $(x_0, f(x_0))$ = (100,10); $(x_1, f(x_1))$=(121,11) ; $(x_2, f(x_2))$=(144,12)

Corrigé de l'exercice 3 :

x	0	1	3	4
f(x)	-1	1	3	5

$L_0(x) = \frac{(x-1)(x-3)(x-4)}{(0-1)(0-3)(0-4)} = -\frac{1}{12}(x-1)(x-3)(x-4)$

$L_1(x) = \frac{(x-0)(x-3)(x-4)}{(1-0)(1-3)(1-4)} = \frac{1}{12}x(x-3)(x-4)$

$L_2(x) = \frac{(x-0)(x-1)(x-4)}{(3-0)(3-1)(3-4)} = -\frac{1}{6}x(x-1)(x-4)$

$L_3(x) = \frac{(x-0)(x-1)(x-3)}{(4-0)(4-1)(4-3)} = \frac{1}{12}x(x-1)(x-3)$

$f(x) \cong P_1(x) = f(x_0)L_0(x) + f(x_1)L_1(x) + f(x_2)L_2(x) + f(x_3)L_3(x)$

Calcul à compléter.

x	-1	0	2	3	5
f(x)	0	1	0	-1	0

$L_0(x) = \frac{(x-0)(x-2)(x-3)(x-5)}{(-1-0)(-1-2)(-1-3)(-1-5)} = \frac{1}{72}x(x-2)(x-3)(x-5)$

$L_1(x) = \frac{(x+1)(x-2)(x-3)(x-5)}{(0+1)(0-2)(0-3)(0-5)} = -\frac{1}{30}(x+1)(x-2)(x-3)(x-5)$

$L_2(x) = \frac{(x+1)(x-0)(x-3)(x-5)}{(2+1)(2-0)(2-3)(2-5)} = -\frac{1}{18}x(x+1)(x-3)(x-5)$

$L_3(x) = \frac{(x+1)(x-0)(x-2)(x-5)}{(3+1)(3-0)(3-2)(3-5)} = -\frac{1}{48}x(x+1)(x-2)(x-5)$

$L_4(x) = \frac{(x+1)(x-0)(x-2)(x-3)}{(5+1)(5-0)(5-2)(5-3)} = -\frac{1}{180}x(x+1)(x-2)(x-3)$

$f(x) \cong P_1(x) = f(x_0)L_0(x) + f(x_1)L_1(x) + f(x_2)L_2(x) + f(x_3)L_3(x) + f(x_4)L_4(x)$

Calcul à compléter.

x	-2	-1	0	1	2	3
f(x)	-3	-2	-2	0	1	2

$L_0(x) = \frac{(x+1)(x-0)(x-1)(x-2)(x-3)}{(-2+1)(-2-0)(-2-1)(-2-2)(-2-3)} = -\frac{1}{120}x(x+1)(x-1)(x-2)(x-3)$

$L_1(x) = \frac{(x+2)(x-0)(x-1)(x-2)(x-3)}{(-1+2)(-1-0)(-1-1)(-1-2)(-1-3)} = \frac{1}{24}x(x+2)(x-1)(x-2)(x-3)$

$L_2(x) = \frac{(x+2)(x+1)(x-1)(x-2)(x-3)}{(0+2)(0+1)(0-1)(0-2)(0-3)} = -\frac{1}{12}(x+2)(x+1)(x-1)(x-2)(x-3)$

$L_3(x) = \frac{(x+2)(x+1)(x-0)(x-2)(x-3)}{(1+2)(1+1)(1-0)(1-2)(1-3)} = \frac{1}{12}x(x+2)(x+1)(x-2)(x-3)$

$L_4(x) = \frac{(x+2)(x+1)(x-0)(x-1)(x-3)}{(2+2)(2+1)(2-0)(2-1)(2-3)} = -\frac{1}{24}x(x+2)(x+1)(x-1)(x-3)$

$L_5(x) = \frac{(x+2)(x+1)(x-0)(x-1)(x-2)}{(3+2)(3+1)(3-0)(3-1)(3-2)} = \frac{1}{120}x(x+2)(x+1)(x-1)(x-2)$

Calcul à compléter.

Résolution des équations différentielles par les méthodes numériques

Exercice 1 :

Soit le problème de Cauchy : $\begin{cases} y' = \dfrac{y-x}{y+x} \\ y(0) = 1 \end{cases}$

Résoudre l'équation différentielle en utilisant la méthode à un seul pas d'Euler sur l'intervalle [0-1] en utilisant un pas $h = 0.1$

Résoudre l'équation différentielle en utilisant la méthode à un seul pas d'Euler sur l'intervalle [0-1] en utilisant un pas $h = 0.2$

Comparer les deux solutions aux points $x = 0.6$ et $x = 1$

Quelle est la solution la plus précise ?

Exercice 2 :

Soit le problème de Cauchy : $\begin{cases} y' = y - \dfrac{2x}{y} \\ y(0) = 1 \end{cases}$

En utilisant un pas constant $h = 0.2$, calculer la solution au point x=1 en utilisant :

1. La méthode à un seul pas d'Euler (ordre 1)
2. La méthode à un seul pas de Taylor (ordre 2)
3. La méthode à un seul pas de Runge Kutta (ordre 2)
4. La méthode à un seul pas de Runge Kutta (ordre 4)

Comparer les résultats numériques avec ceux de la solution analytique donnés par :

$$y = \sqrt{2x+1}$$

Exercice 3 :

Soit le problème de Cauchy : $\begin{cases} y' = y + x \\ y(0) = 1 \end{cases}$

Programmer les méthodes de résolution de l'équation différentielle donnée dans le problème de Cauchy (méthode d'Euler, méthode de Taylor, méthode de Runge Kutta d'ordre 2 et d'ordre 4)

Il faut lire le point initial et le point final où la solution finale doit être calculée.
Il faut lire le nombre d'intervalles
Programmer la méthode d'Euler
Programmer la méthode de Taylor
Programmer la méthode de Runge Kutta d'ordre 2
Programmer la méthode Runge Kutta d'ordre 4
Calculer la solution analytique donnée par $y(x) = -2e^{x} - x - 1$
Comparer l'erreur des quatre méthodes
Calculer la constante *k* figurant dans les équations de calcul des erreurs

Il faut généraliser le programme par l'utilisation des sous programmes de type function pour la définition de la fonction *f(x)=y+x*

Corrigé de l'exercice 1 :

Soit le problème de Cauchy : $\begin{cases} y' = \dfrac{y-x}{y+x} \\ y(0) = 1 \end{cases}$

La méthode à un seul pas d'Euler sur l'intervalle [0,1] en utilisant un pas h = 0.1

Le processus de résolution d'Euler se résume en :

$$\begin{cases} \text{Étant donnée } y(0) = 1 \\ \quad \text{Sachant que } y'(x_i) = \frac{y(x_i)-x_i}{y(x_i)+x_i} \\ \quad \text{Alors : } y(x_i + h) = y(x_i) + \mathrm{h} * y'(x_i) \end{cases}$$

1- L'application de ce processus a donné le résultat suivant :

Sur l'intervalle [0,1] en utilisant un pas =0.1, les x_i où on doit approximer y seront : 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0

x	y
0.000000E+00	1.000000
1.000000E-01	1.100000
2.000000E-01	1.183333
3.000000E-01	1.254418
4.000000E-01	1.315818
5.000000E-01	1.369193
6.000000E-01	1.415694
7.000000E-01	1.456161
8.000001E-01	1.491231
9.000001E-01	1.521399
1.000000	1.547062

1- L'application de ce processus a donnée le résultat suivant :

Sur l'intervalle [0,1] en utilisant un pas h= 0.2, les x_i où on doit approximer y seront : 0, 0.2, 0.4, 0.6, 0.8, 1.0.

x	y
0.000000E+00	1.000000
2.000000E-01	1.200000
4.000000E-01	1.342857
6.000000E-01	1.451054
8.000000E-01	1.534041
1.000000	1.596940

Comparaison de la solution aux points x = 0.6 et x =1 :

x	h=0.1	h=0.2
6.000E-01	1.415694	1.451054
1.000000	1.547062	1.596940

Il est évident que la solution avec le pas h=0.1 soit plus précise que celle avec le pas h=0.2

Corrigé de l'exercice 2 :

Soit le problème de Cauchy : $\begin{cases} y' = y - \dfrac{2x}{y} \\ y(0) = 1 \end{cases}$

Processus d'Euler :

$$\begin{cases} \text{Étant donnée } y(0) = 1 \\ \text{Sachant que } y'(x_i) = y(x_i) - \frac{2x_i}{y(x_i)} = \text{F}(x_i, y(x_i)) \\ \text{Alors } y(x_i + h) = y(x_i) + \text{h} * y'(x_i) \end{cases}$$

Pour la méthode de Taylor :

Le développement limité de Taylor en point $x_i + h$ tronqué à l'ordre 2 donne :

$$y(x_i + h) \cong y(x_i) + h * y'(x_i) + \frac{h^2}{2} y''(xi)$$

Avec $y''(x) = F'(x, y(x)) = (\frac{\partial F}{\partial x} + \frac{\partial F}{\partial y} * \frac{\partial y}{\partial x})_{(xi,yi)} = (\frac{\partial F}{\partial x} + \frac{\partial F}{\partial y} * F)_{(xi,yi)}$

Donc le processus de résolution des équations différentielles (EDPs) par Taylor est donné par :

$$\begin{cases} \text{Étant donnée } y(0) = 1 \\ \text{Sachant que } y'(x_i) = y(x_i) - \frac{2x_i}{y(x_i)} = \text{F}(x_i, y(x_i)) \\ \text{Alors } y(x_i + h) = y(x_i) + \text{h} * \text{y}'(x_i) + \frac{h^2}{2} F'(xi, y(x_i)) \\ \text{Avec } F'(x, y(x)) = -2/y + \left(1 + \frac{2x}{y^2}\right) * (y - \frac{2x}{y}) \end{cases}$$

Processus de Runge-Kutta d'ordre 2 :

$$\begin{cases} \text{Étant donnée } y(0) = 1 \\ \text{Sachant que } y'(x_i) = y(x_i) - \frac{2x_i}{y(x_i)} = F(x_i, y(x_i)) \\ \text{on calcul } \quad m_1 = y'(x_i) = F(x_i, y_i) \\ m_2 = y'(x_i + h) = F(x_i + h, y(x_i) + h * m_1) \\ \text{Puis } y(x_i + h) = y(x_i) + h * (\frac{m_1 + m_2}{2}) \end{cases}$$

Processus de Runge-Kutta d'ordre 4 :

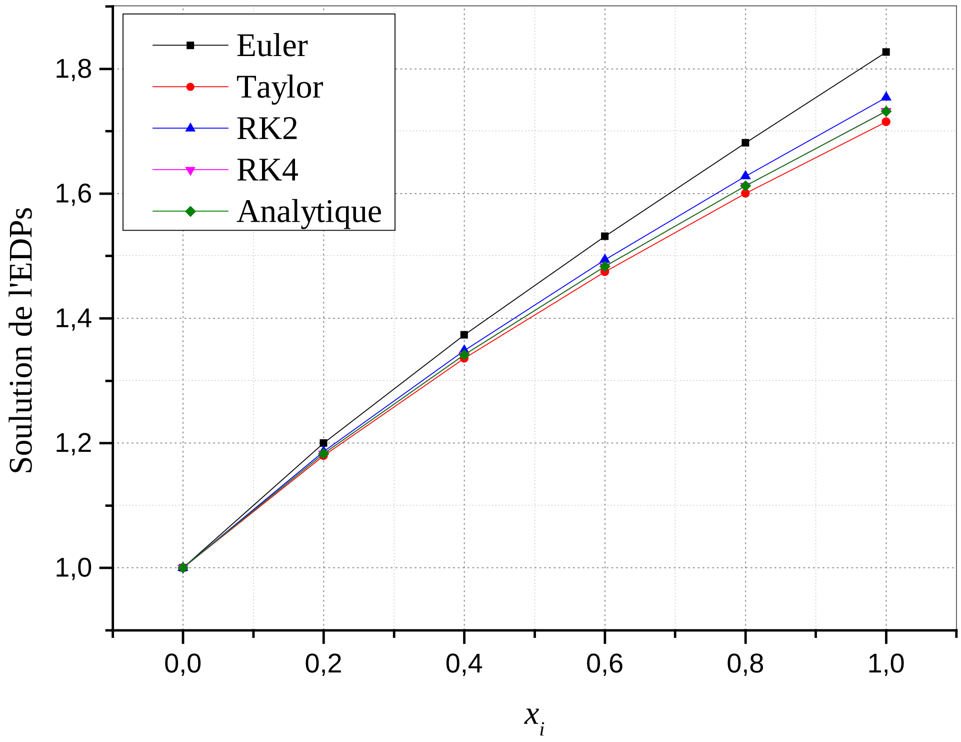
$$\begin{cases} \text{Étant donnée } y(0) = 1 \\ \text{Sachant que } y'(x_i) = y(x_i) - \frac{2x_i}{y(x_i)} = F(x_i, y(x_i)) \\ \text{on calcul } \quad m_1 = y'(x_i) = F(x_i, y_i) \\ m_2 = y'(x_i + \frac{h}{2}) = F(x_i + \frac{h}{2}, \ y(x_i) + \frac{h}{2} * m_1) \\ m_3 = y'(x_i + \frac{h}{2}) = F(x_i + \frac{h}{2}, \ y(x_i) + \frac{h}{2} * m_2) \\ m_4 = y'(x_i + h) = F(x_i + h, \ y(x_i) + h * m_3) \\ \text{Puis } y(x_i + h) = y(x_i) + \mathrm{h} * (\frac{m_1 + 2 * m_2 + 2 * m_3 + m_4}{6}) \end{cases}$$

La solution analytique est donnée par : $y = \sqrt{2x+1}$

L'application des différents processus a donné les résultats suivants sur l'intervalle [0,1] avec h=0.2 (tous les résultats – même intermédiaires – doivent êtres calculés avec une précision de l'ordre de 10^{-6})

x_i	Euler	Taylor	RK2	RK4	Analytique
0.0	1.000000	1.000000	1.000000	1.000000	1.000000
0.2	1.200000	1.180000	1.186667	1.183229	1.183216
0.4	1.373333	1.335958	1.348312	1.341667	1.341641
0.6	1.531495	1.474795	1.493704	1.483281	1.483240
0.8	1.681085	1.600414	1.627861	1.612514	1.612452
1.0	1.826949	1.715073	1.754205	1.732142	1.732051

La figure ci-dessous illustre la représentation graphique de la solution de l'équation différentielle par les différentes méthodes comparée à la solution analytique.



Il est clair que la méthode de Runge Kutta d'ordre 4 est la plus précise pour la résolution des équations différentielles.

Programme de calcul

```
program differentiel
         REAL EUL,TAY,RG2,RG4,tt,a,b,y,ana
         INTEGER n
         write (*,*)'donner la solution: Yo='
         read (*,*) y
         write (*,*)'donner le point initial: a'
         read (*,*) a
         write (*,*)'donner le piont final: b'
         read (*,*)b
         write (*,*)'donner le nbr d interval: n'
         read (*,*)n
c           initialisation
      EUL=Y
         TAY=Y
         RG2=Y
         RG4=Y
         tt=a
         h=(b-a)/n
         write (*,*)'        EULER            TAYLOR        KUTTA 2      KUTTA 4'
         do i=1,n,1
         EUL=EUL+h*f(EUL,t)
         TAY=TAY+h*f(TAY,t)+((h**2)/2)*(f1(TAY,t)+(f2(TAY,t)*f(TAY,t)))

         p11=f(t,RG2)
         p22=f(t+h,RG2+h*p11)
         RG2=RG2+(h/2)*(p11+p22)

         p1=f(t,RG4)
         p2=f(t+(h/2),RG4+(h/2)*p1)
         p3=f(t+(h/2),RG4+(h/2)*p2)
         p4=f(t+h,RG4+h*p3)
         RG4=RG4+(h/6)*(p1+2*p2+2*p3+p4)
         t=t+h
         write (*,*)EUL,TAY,RG2,RG4
         enddo

         ana=2*exp(b)-b-1
         write (*,*)        '                                                   '
         write(*,*)  'la solution analytique=',ana

         ereul=abs(ana-eul)
         ertay=abs(ana-tay)
         errg2=abs(ana-rg2)
         errg4=abs(ana-rg4)

         akeul=ereul/h
         aktay=ertay/h**2
         akrg2=errg2/h**2
         akrg4=errg4/h**4
         write (*,*)        '                        les erreurs                            '
      write(*,23)  EREUL,ERTAY,ERRG2,ERRG4
      write(*,23)  akEUL,akTAY,akRG2,akRG4
23       format(8x, F8.6,8X,F8.6,8X,F8.6,8X,F8.6)

      end
         function f(y,t)
         f=y+t
         return
         end
      function f1(y,t)
         z=y+t
         f1=1.
         return
         end
      function f2(y,t)
         z=t+y
         f2=1.
         return
         end
```

Résolution des systèmes d'équations linéaires

1. Introduction

Une équation est dite linéaire si chacun de ses termes a au plus une variable et chaque variable apparait à la puissance d'ordre un.
Exemple :

$2x + y - 6z = 4$ est linéaire

$2xy + y - 7 = 2$ est non linéaire

$2x^2 + y - 7 = 2$ est non linéaire

Un système d'équations linéaires est un système de N équations linéaires. Une solution à un système d'équations linéaires consiste en des valeurs des N variables qui vérifient simultanément les N équations. Le système est souvent noté : **A.X=B**

2. Méthodes de résolution des systèmes d'équations linéaires

En général, il existe 2 types de méthodes numériques pour résoudre un système d'équations linéaires. Les méthodes dites directes et les méthodes indirectes qui sont théoriquement des processus *infinis*.

Les *méthodes directes* (telle que la méthode de gauss) sont des processus *finis*. Une méthode directe conduit à une solution en un nombre fini d'étapes. Cette solution si elle existe est la solution du système.

Une *méthode indirecte* (telle que la méthode de Seidel) fait passer d'un estimé X_n à un autre estimé X_{n+1} de cette solution. Ce processus demande un nombre infini d'opérations arithmétiques pour produire une solution exacte. Cela est en pratique impossible. En fait en doit toujours limiter un processus infini par un nombre fini d'itérations et cela génère ce qu'on appel l'erreur de troncature.

Les méthodes itératives sont rarement utilisées pour la résolution de systèmes à matrice pleine et de faible dimension et les méthodes itératives sont généralement préférées pour résoudre les systèmes de grande taille.

2.1. Méthodes directe : méthode de gauss

La méthode de gauss consiste à traduire un système A.X=B en un autre système A'X=B' avec A' matrice triangulaire supérieure. Cette phase est appelée **Triangularisation**.

Le système d'équations à résoudre est de la forme :

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & a_{24} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & a_{34} & \dots & a_{3n} \\ a_{41} & a_{42} & a_{43} & a_{44} & \dots & a_{4n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & a_{n4} & \dots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ \vdots \\ b_n \end{pmatrix}$$

Après triangularisation, le système prend la forme :

$$\begin{pmatrix} a'_{11} & a'_{21} & a'_{11} & a'_{11} & \dots & a'_{11} \\ 0 & a'_{22} & a'_{23} & a'_{24} & \dots & a'_{2n} \\ 0 & 0 & a'_{33} & a'_{34} & \dots & a'_{3n} \\ 0 & 0 & 0 & a'_{44} & \dots & a'_{4n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & a'_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ \vdots \\ b'_n \end{pmatrix}$$

La solution X (x1, x2, x3, …, xn) sera par la suite aisément calculée. Il suffit de calculer $x_n = \frac{b'_n}{a'_{nn}}$ puis de remplacer la valeur de x_n dans le calcul de x_{n-1} .

$x_{n-1} = \frac{(b'_{n-1} - .\, a'_{nn}\, x_n)}{a_{n,n-1}}$ et ainsi de suite on remonte dans le calcul de x_n, x_{n-1}, x_{n-2}, x_{n-3} jusqu'à calculer x_1. Cette phase est appelée phase de **résolution.**

L'algorithme de gauss sera composé donc de deux principales phases :

1. 1ère phases : Triangularisation
2. 2ème phase : Résolution.

L'algorithme de gauss et présenté ci-dessous.

```
Initialisation :   Lecture de la Taille N, de la matrice A et du vecteur B.
Phase 1 : triangularisation :
pour k=1 à N-1
     pour i=k+1 à N
           w=a(i,k)/a(k,k)
            pour   j=1 à N
                  si (j<i) alors
                                a(i,j)=0
                  sinon
                          a(i,j)=a(i,j) - w*a(k,j)
                  finsi
           fin pour
            b(i)=b(i)-w*b(k)
      fin pour
fin pour
Phase 2 : Résolution :  calcul du vecteur X
Pour    i =N   à    1
            S=0
            Pour j= i+1  à  N
            S= S + a(i,j) *X(j)
            fin pour
            X(i)= (b(i)-s)/a(i,i)
   fin pour
```

Exercice

Écrire le programme Fortran qui implémente la méthode de Gauss décrite précédemment.

2.2. Méthodes indirecte : méthode de Gauss - Seidel

Une méthode itérative met le vecteur X en relation récurrente. X à l'instant *t* dépend de X à l'instant *t*-1. Pour cela le système aura besoin en entrée de la matrice A, du vecteur B et d'une valeur initiale de X (X_0). Après un certain nombre d'itérations, la suite créée : X_0, X_1, X_2, X_3,…, X_n converge vers une solution approché du système. La procédure s'arrête quand $Max(|X_n - X_{n+1}|) \le \varepsilon$

La matrice A est décomposé comme suit :

$$A = (D - L) - U$$

$$A\,X = B \quad \text{devient} \quad [(D - L) - U]X = B$$

$$(D - L)X^{k+1} = UX^{k} + B$$

$$DX^{k+1} = LX^{k+1} + UX^{k} + B$$

$$X^{k+1} = D^{-1}L\,X^{k+1} + D^{-1}UX^{k} + D^{-1}B$$

avec

D(i,j)=a(i,i) si i = j et 0 sinon. On obtient ainsi la matrice D.

$$D = \begin{pmatrix} a_{11} & 0 & 0 & 0 & \cdots & 0 \\ 0 & a_{22} & 0 & 0 & \cdots & 0 \\ 0 & 0 & a_{33} & 0 & \cdots & 0 \\ 0 & 0 & 0 & a_{44} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 0 \end{pmatrix}$$

L(i,j)=-a(i,j) si i<j et 0 sinon. On obtient ainsi la matrice L.

$$L = \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots & 0 \\ -a_{21} & 0 & 0 & 0 & \cdots & 0 \\ -a_{31} & -a_{32} & 0 & 0 & \cdots & 0 \\ -a_{41} & -a_{42} & -a_{43} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -a_{n1} & -a_{n2} & -a_{n3} & -a_{n4} & \cdots & 0 \end{pmatrix}$$

D(i,j)=-a(i,i) si i>j et 0 sinon. On obtient ainsi la matrice D.

$$U = \begin{pmatrix} 0 & -a_{12} & -a_{13} & -a_{14} & \cdots & -a_{1n} \\ 0 & 0 & -a_{23} & -a_{24} & \cdots & -a_{2n} \\ 0 & 0 & 0 & -a_{34} & \cdots & -a_{3n} \\ 0 & 0 & 0 & 0 & \cdots & -a_{4n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 0 \end{pmatrix}$$

$D_{i,i}^{-1} = {}^{1}/_{a_{i,i}}$ est la matrice inverse de D.

X se calcul alors comme suit : X à l'instant *k+1* :

$$X_i^{k+1} = {\left(-\sum_{j=1}^{i-1} a_{i,j}\,.\,X_j^{k+1} - \sum_{j=i+1}^{n} a_{i,j}\,.\,X_j^{k}\right)}\Big/{a_{i,i}}$$

Les éléments $a_{i,i}$ de la matrice A sont appelés des Pivots. La condition suffisante pour que le processus de gauss converge vers une solution est : $a_{i,i} \geq \sum_{j=1}^{n} a_{i,j}$

On peut faire un réarrangement de la matrice A (permutation de lignes) de telle sorte à ce que cette condition sera vérifiée.

Programme de la méthode de Gauss :

```
! Méthode de Gauss :

      real a(1000,1001),m(1000,1000)
          real b(1000),bb(1000),x(1000),w,s

   10 print *,'donner la taille N du système :'
          read(*,*)N
          if(N.gt.1000)then
                      print *,'N doit être <=1000'
                      goto  10
      endif
          print *,' introduisez les valeurs de la matrice  A'
          read(*,*)((m(i,j),j=1,N),i=1,N)
          print *,' introduisez les valeurs du vecteur  B'
          read(*,*)(b(i),i=1,N)

! construction de la matrice augmentée

          do i=1,N
            do j=1,N
        a(i,j)=m(i,j)
            enddo
           a(i,N+1)=b(i)
      enddo

! affichage de la matrice augmentée

      print *,' la matrice augmentee :'
          do i=1,N
          write(*,*)(a(i,j),j=1,N+1)
      enddo
          do k=1,N-1
             do i=k+1,N
            w=a(i,k)/a(k,k)
            do j=1,N+1
                if(j.le.k)then
                                a(i,j)=0.0
            else
                     s=w*a(k,j)
                 a(i,j)=a(i,j)-s
            endif
                enddo
             enddo
      enddo

! affichage de la matrice triangulaire résultat
      print *,' la matrice triangulaire superieur:'
          do i=1,N
          write(*,*)(a(i,j),j=1,N+1)
      enddo
          do j=1,N
          x(j)=0.0
      enddo

! calcul du vecteur résultat X

          !x(N)=a(N,N+1)/a(N,N)
      do i=N,1,-1
                 s=0.0
                 do j=1,N
                 s=s+(a(i,j))*x(j)
                 enddo
             x(i)=(a(i,N+1)-s)/a(i,i)
      enddo

! affichage du vecteur solution
      print *,' le vecteur solution:'
          write(*,*)(x(i),i=1,N)

! test du résultat '

      do i=1,N
             s=0.0
             do j=1,N
                s=s+m(i,j)*x(j)
             enddo
             bb(i)=s
          enddo

! affichage de b et b'

      print *,'ancien b :'
      write(*,*)(b(i),i=1,N)
      print *,'nouveau b :'
      write(*,*)(bb(i),i=1,N)
          end
```

Programme de la méthode de Gauss-Seidel

```
! essey : 1 2 4 5 4 18 , eps 0.001 iter 300 puis 800 puis 5000 puis 10000 50000
      double precision a(1000,1000),b(1000),bb(1000),temp
      double precision s,eps,normeR,big,tt,x(1000)
   10 print *,'donner la taille N du système :'
          read(*,*)N
          if(N.gt.1000)then
                      print *,'N doit être <=1000'
                      goto  10
      endif
          print *,' introduisez les valeurs de la matrice  A'
          read(*,*)((a(i,j),j=1,N),i=1,N)
          print *,' introduisez les valeurs du vecteur  B'
          read(*,*)(b(i),i=1,N)
          print *,' introduisez epsilon'
          read(*,*)eps
          print *,'donner la valeur maximal d"iteration:'
          read(*,*)kmax

! réarrangement
          do k=1,N
           s=0.0
           do j=1,N
           if(j.ne.k)s=s+a(k,j)
       enddo
           if(s.gt.a(k,k))then
            ! rechercher une ligne i qui sera permutée avec la ligne K
                do i=K+1,N
                    s=0.0
                    do j=1,N
                    if(j.ne.k)s=s+a(i,j)
                enddo
                               if(s.le.a(i,k))then
                                tt=b(i)
                                                                       b(i)=b(k)
                                                                       b(k)=tt
                                         do j=1,N
                                   tt=a(k,j)
                                   a(k,j)=a(i,j)
                                   a(i,j)=tt
                                   enddo
                                   goto 100
                    endif
           enddo
100     endif
          enddo
                    print *,'systeme rearange'
          do i=1,N
           write(*,*)(a(i,j),j=1,N)
           write(*,*)b(i)
      enddo
! boucle principale
          do  i=1,N
          x(i)=0.0
          enddo
          do k=1,kmax
             big=0.0
         do i=1,N
             s=0.0
             if(i.gt.1)then
                do j=1,i-1
                s=s+a(i,j)*x(j)
                enddo
                if(j.eq.N-1)goto 30
         endif
             do j=i+1,N
             s=s+a(i,j)*x(j)
             enddo
  30     temp=(1.0/a(i,i))*(b(i)-s)
             if(abs(temp-x(i)).gt.big)big=abs(temp-x(i))
             x(i)=temp
             enddo
        if(big.le.eps)then
                              print *,' système convergent'
                              goto 40
            endif
            if(k.gt.kmax)then
                         print *,'système non convergent '
                                          goto 40
        endif
      enddo ! fin de la boucle itération K

  40  print *,' le vecteur solution:'
          write(*,*)(x(i),i=1,N)
! test du résultat '
      do i=1,N
             s=0.0
             do j=1,N
                s=s+a(i,j)*x(j)
             enddo
             bb(i)=s
          enddo
! affichage de b et b'
      print *,'ancien b :'
      write(*,*)(b(i),i=1,N)
      print *,'nouveau b :'
      write(*,*)(bb(i),i=1,N)
      end
```

Exercices

Méthode de Gauss :

1- Exécutez le programme de Gauss pour le système d'équation suivant :

$$A = \begin{matrix} 2 & 5 & 8 \\ 6 & 7 & 3 \\ 9 & 2 & 4 \end{matrix} \qquad B = \begin{matrix} 1 \\ 2 \\ 3 \end{matrix}$$

2- Calculez l'écart entre le vecteur B_initial et le vecteur B_recalculé après résolution

3- Exécutez le programme de Gauss pour le système

$$A = \begin{matrix} 0 & 5 & 8 \\ 6 & 1 & 3 \\ 9 & 2 & 4 \end{matrix} \qquad B = \begin{matrix} 1 \\ 2 \\ 3 \end{matrix}$$

Expliquez le résultat, que doit vérifier la matrice A pour éviter cette erreur.

Méthode de Gauss – Seidel :

1- Exécutez le programme de Gauss Seidel pour le même système :

$$A = \begin{matrix} 2 & 5 & 8 \\ 6 & 7 & 3 \\ 9 & 2 & 4 \end{matrix} \qquad B = \begin{matrix} 1 \\ 2 \\ 3 \end{matrix}$$

Avec $\varepsilon = 0.0001$ et Nbr_iter = 200000 (récupérer le résultat : X, B ancien et B nouveau)

2- Calculer l'écart entre B ancien et B nouveau. Qu'appelle –t-on cet écart : erreur d'arrondi ou de troncature ?

3- Comparer ce résultat avec celui de Gauss.

4- Exécutez le programme de Gauss Seidel pour ε vaut respectivement 0.01, 0.001 , 0.0001, 0.00001 et iter= 200 000. Puis ré-exécutez le même programme pour iter vaut respectivement 100, 200, 1000, 20000, 100000 et $\varepsilon = 0.00001$ (porter sur la feuille les valeurs de X, B_ancien et B_nouveau).

5- Que peut on déduire sur l'évolution de l'écart entre B_ancien et B_nouveau par rapport à ε et à iter.

6- reprenant maintenant le programme source de la méthode de Gauss Seidel

a/ dresser l'organigramme de cet partie (réarrangement de la matrice A)

b/ qu'elle est l'utilité de la partie réarrangement dans ce programme. Expliquer brièvement son principe (d'après le programme).

Exécution du programme de Gauss :

```
donner la taille N du systÞme :
3
  introduisez les valeurs de la matrice  A
2
5
8
6
7
3
9
2
4
  introduisez les valeurs du vecteur  B
1
2
3
  la matrice augmentee :
      2.00000         5.00000         8.00000         1.00000
      6.00000         7.00000         3.00000         2.00000
      9.00000         2.00000         4.00000         3.00000
  la matrice triangulaire superieur:
      2.00000         5.00000         8.00000         1.00000
      0.00000        -8.00000        -21.0000        -1.00000
      0.00000         0.00000         21.8125         1.06250
  le vecteur solution:
     0.312321       -2.865329E-03    4.871060E-02
 ancien b :
      1.00000         2.00000         3.00000
 nouveau b :
      1.00000         2.00000         3.00000

Press RETURN to close window...
```

Exécution du programme de Gauss-Seidel :

```
donner la taille N du système :
3
  introduisez les valeurs de la matrice  A
2
5
8
6
7
3
9
2
4
  introduisez les valeurs du vecteur  B
1
2
3
  introduisez epsilon
0.0001
 donner la valeur maximal d"iteration:
200000
 système réarrangé
           9.00000000000              2.00000000000
4.00000000000

           3.00000000000
           6.00000000000              7.00000000000
3.00000000000

           2.00000000000
           2.00000000000              5.00000000000
8.00000000000

           1.00000000000
  système convergent
  le vecteur solution:
          0.313564383229          1.694481437490E-02
3.601839520837E-02

 ancien b :
           3.00000000000              2.00000000000
1.00000000000

 nouveau b :
           3.00004265865              2.10805518563
1.00000000000


Press RETURN to close window...
```

Contrôle continu

En date du mardi le 31 mai 2011

Sujet N°01

Exercice 2 :

Soit la fonction f(x) = x^2

1- En prenant h=0.5, calculer numériquement les intégrales suivantes par la méthode des trapèzes.

A1=$\int_0^1 f(x)dx$; A2=$\int_0^2 f(x)dx$; A3=$\int_0^3 f(x)dx$

2- Comparer ces résultats numériques aux valeurs analytiques exactes correspondantes.

3- Remplir le tableau suivant :

x	1	2	3
G(x)	A1	A2	A3

4- Calculer numériquement g'' (2).

5- Que représente la valeur g'' (2) ? Comparer le résultat précédent avec la valeur analytique exacte.

Exercice 2 :

Pourquoi la méthode de Simpson est généralement plus exacte que la méthode des trapèzes ?

Ecrire l'algorithme de la méthode des trapèzes.

Corrigé du contrôle continu

Sujet N°01

Corrigé de l'exercice 1

h=0.5 ; $f(x) = x^2$; $I(x) = \frac{1}{3}x^3$ avec I(x) la primitive de *f(x)*.

1.1 Calcul numérique des intégrales A1, A2, A3 par la méthode des trapèzes :

Formule des trapèzes : $\int_a^b f(x) = \frac{h}{2}(f(a) + f(b)) + h\sum_{i=1}^{n-1} f(xi)$

A1=$\int_0^1 f(x) = \frac{0.5}{2}\big(f(0) + f(1)\big) + 0.5(f(0.5))$ = **0.375**

A2=$\int_0^2 f(x) = \frac{0.5}{2}\big(f(0) + f(2)\big) + 0.5(f(0.5) + f(1) + f(1.5))$ =**2.75**

A3=$\int_0^3 f(x) = \frac{0.5}{2}\big(f(0) + f(3)\big) + 0.5\big(f(0.5) + f(1) + f(1.5) + f(2) + f(2.5)\big)$ = **9.125**

1.2 Calcul analytique exacte de A1, A2, A3

Noter que I(0)=0 $\Rightarrow$ $\int_0^b f(x)$ =I(b)

A1 =$\int_0^1 f(x)$ = I(1) – I(0) = **I(1) = 0.333**

A2=$\int_0^2 f(x)$ = I(2) – I(0) = **I(2) =2.666**

A3=$\int_0^3 f(x)$ = I(3) – I(0) = **I(3) = 9.000**

2. Comparaison

En terme d'erreur absolue :

err_A1= 0.042

err_A2=0.084

err_A3=0.125

Erreur commise dans chaque trapèze : err_par_trapèze_A1 = $\frac{0.042}{2} = 0.021$

err_par_trapèze_A2 = $\frac{0.084}{4} = 0.021$

err_par_trapèze_A3 = $\frac{0.125}{6} = 0.0208$

Erreur dans chaque trapèze est stable vu que les valeurs de f(x) et I(x) sont exactes

3. Calcul numérique de g''(2)

x	1	2	3
G(x)	0.375	2.75	9.125

$$g''(x0) = \frac{f(x0-h)-2f(x0)+f(x0+h)}{h^2} \Rightarrow g''(2) = \frac{f(2-1)-2f(2)+f(2+1)}{1^2}$$

$$= \frac{0.375-2*2.75+9.125}{1}$$

$g''(2)_{num} = 4.0$

La fonction g(x) est la fonction I(x) définit par les points 1, 2 et 3

Donc g'(x)=f(x) dans les points 1, 2, 3

Et g''(x)=f'(x) dans les points 1, 2, 3

Donc la valeur analytique de g''(2)=f'(2)=2*2=4.0 $g''(2)_{analy} = 4.0$

Corrigé de l'exercice 2

1- La méthode de Simpson est généralement plus exacte que la méthode des trapèzes puisque elle fait intervenir 3 points (x_i-h, x_i, x_i+h) dans le calcul de chaque aire partielle Ai en approximant la fonction à intégrer entre l'intervalle [x_i-h, x_i+h] par une parabole (interpolation quadratique).

Par contre, la méthode des trapèzes fait intervenir juste 2 points (x_i *et* x_i+h) dans le calcul de chaque aire partielle Ai en approximant la fonction à intégrer entre l'intervalle [x_i, x_i+h] par une droite (interpolation linéaire)

2- l'algorithme de la méthode des trapèzes

```
début
        Lire(a,b,n)
         h=(b-a)/n
        som←h/2*(f(a)+f(b))
        s←0.0

        pour i←1,n-1,1
         s←s+f(a+i*h)
        FinPour

        som←som+h*s
        écrire('l″intégrale de f entre', a , ' et ', b , ' est = ', som)
fin

fonction  f(x)
----  corps de la fonction f
Fin
```

Sujet N°02

Exercice 1 :

Soit à résoudre l'équation suivante $x^2 - 4 = 0$ par la méthode de newton avec x_0=1 et ε = 0.1.

1. Pour x_0=1, la convergence de la méthode de Newton est – elle garantie ? Justifier.
2. Calculer la valeur approchée x_n de la racine.
3. Compléter le tableau suivant :

x	1	2.0	2.05	2.5
F(x)	*f(1)*	*f(2.0)*	*f(2.05)*	*f(2.5)*

4. Calculer les valeurs numériques f'(1), f'(2), f' (2.05), f' (2.5), avec le schéma excentré en avant (utiliser le schéma excentré en arrière pour 2.5).
5. Calculer l'erreur absolue commise dans le calcul de chaque dérivée.
6. A quel xi correspond la plus grande valeur d'erreur absolue ? Pourquoi ?

Exercice 2 :

1. Quelle est l'influence du pas de discrétisation h dans la précision d'une méthode de dérivation ?
2. Ecrire l'algorithme de la méthode de newton.

Corrigé du contrôle continu

Sujet N°02

Corrigé de l'exercice 1

f(x)= $x^2 - 4$; en prenant x_0=1 et ε = 0.1

1- La condition suffisante de convergence de la méthode de newton exige que *f(x_0)f''(x_0)>0.* Pour x_0=1 : *f(1)f''(1)*= (-3)*(2) =-6<0. On ne peut rien dire. Le processus de résolution peut converger comme il peut diverger.

2-

Processus itératif de newton
$$\begin{cases} x_0=1 \\ x_{i+1}= x_i - \frac{f(x_i)}{f'(x_i)} \\ \Rightarrow x_{i+1}=x_i - \frac{x_i{}^2 - 4}{2*x_i} \\ \text{Teste d'arrêt : } |x_{i+1}-x_i| \leq \varepsilon \quad (\varepsilon = 0.1) \end{cases}$$

Par application on obtient le tableau suivant avec comme racine approché $\mathbf{x_n}$**=2.0006**

i	x_i	$f(x_i)$	f'(xi)	$\|x_{i+1}-x_i\|$
0	1	-3	2	
1	2.5	2.25	5	1.5
2	2.05	0.2025	4.1	0.45
3	2.0006	0.0024	4.0012	0.04

3-

x	1	2.0	2.05	2.5
f(x)	-3	0	0.2025	2.25

4- Calcul numérique de la drivée :

par schéma excentré en avant : $f'(x0) = \frac{f(x0+h)-f(x0)}{(x0+h)-x0}$

Pour f'(1) : le point en avant de 1 est 2 $\Rightarrow$ h=2-1=1

$$f'(1) = \frac{f(1+1)-f(1)}{1} = \frac{f(2)-f(1)}{1} = \frac{0-(-3)}{1} = 3$$

Pour f'(2) : le point en avant de 2 est 2.05 $\Rightarrow$ h=2.05-2=0.05

$$f'(2) = \frac{f(2+0.05)-f(2)}{0.05} = \frac{f(2.05)-f(2)}{0.05} = \frac{0.2025-(0)}{0.05} = 4.05$$

Pour f'(2.05) : le point en avant de 2.05 est 2.5 $\Rightarrow$ h=2.5-2.05=0.45

$$f'(2) = \frac{f(2.05+0.45)-f(2.05)}{0.45} = \frac{f(2.5)-f(2.05)}{0.45} = \frac{2.25-0.2025}{0.45} = 4.55$$

Par schéma excentré en arrière : $f'(x0) = \frac{f(x0)-f(x0-h)}{x0-(x0-h)}$

Pour f'(2.5) : le point en arrière de 2.5 est 2.05 $\Rightarrow$ h=2.5-2.05=0.45

$$f'(2.5) = \frac{f(2.5)-f(2.5-0.45)}{0.45} = \frac{f(2.5)-f(2.05)}{0.45} = \frac{2.25-0.2025}{0.45} = 4.55$$

Le calcul exact des valeurs des dérivées : f'(1)=2 ; f'(2)=4 ; f'(2.05)=4.1 ; f'(2.5)=5

5- erreur absolue commise dans le calcul de :

f'(1) : err_abs= |2-3|=1 ; f'(2) : err_abs= |4-4.05|=0.05 ; f'(2.05) : err_abs= |4.1-4.55|=0.45

f'(2.5) : err_abs= |5-4.55|=0.45

6- La plus grande erreur absolue a été commise lors du calcul de f'(1) parce que la distance qui sépare le point en question (1) et son point adjacent (2) est =1 et est la plus grande distance (h le plus élevé) comparé aux autres résultats

Corrigé de l'exercice 2

Le choix du pas de discrétisation h est très critique. Son influence est directe et importante dans la précision des résultats obtenus par les différents schémas de calcul des dérivées numériques

- Si h est trop grand, les résultats s'éloignent des résultats exacts
- Si h est trop petit, l'erreur d'approximation de f' s'accroit par la présence des erreurs d'arrondi (dans le calcul manuel ou par ordinateur à simple précision).

2- l'algorithme de la méthode de Newton :

```
début
        Lire(x0,epsilon)
        Xanc←x0
5        Xnouv ← Xanc – f(Xanc)/ff(Xanc)
         Si (abs(Xnouv-Xanc)≤epsilon)alors
           Ecrire('la racine de f est =', Xnouv)
         Sinon
                Xanc←Xnouv
                Aller à 5
         finsi
fin
fonction  f(x)
----  corps de la fonction f
Fin
fonction ff(x)
-----corps de la fonction f'
fin
```

Sujet de l'examen Final (juin 2011) proposé aux étudiants du département de Génie Mécanique USTO-MB

Durée de l'examen: 2h00

Exercice 1 :

Soit la fonction $f(x) = \frac{1}{x}$

1. Déterminer le polynôme de Lagrange **P(x)** qui interpole ***f*** par les 3 points d'appuis d'abscisses 1, 2, 3.

2. Calculer **P**(1,5) ; P (2.5)

3. Déterminer l'erreur d'interpolation commise dans de calcul de P(1.5) et P(2.5).

4. Compléter le tableau ci-dessous puis calculer $\int_1^3 P(x)$ par la méthode des trapèzes.

X	1	1.5	2	2.5	3
P(x)					

5. Comparer le résultat obtenu aux résultats analytiques.

Exercice 2 :

Soit le problème de Cauchy suivant :

$$\begin{cases} y'(x) = -y^2(x) \\ y(1) = 1 \end{cases}$$

1. Montrer que $y(x) = \frac{1}{x}$ vérifie le problème de Cauchy posé.

2. Approximer la solution $y(x)$ de l'équation différentielle par la méthode de Runge Kutta d'ordre 2 sur l'intervalle [1,1.6] avec un pas 0.2.

3. Comparer la valeur y (1.6) à la valeur analytique exacte (erreur absolue).

4. En s'appuyant sur l'approximation de y(x), calculer $y'(1.4)$ par le schéma centré.

5. Comparer le résultat numérique de e '(1.4)obtenu au résultat analytique exact.

6. En s'appuyant sur l'approximation de y(x), calculer la valeur numérique de $y''(1.4)$.

7. Comparer le résultat obtenu à la valeur analytique exacte.

Corrigé de l'Examen

Corrigé de l'exercice 1 :

Soit $f(x) = \frac{1}{x}$

Interpolation de Lagrange :

1- $P_n(x) = \sum_{i=0}^{n} f(x_i) L_i(x)$

avec

$$L_i(x) = \prod_{j=0, j\neq i}^{n} \frac{(x - x_j)}{(x_i - x_j)}$$

Nombre de points d'appui =3 → polynôme d'ordre 2 (interpolation quadratique). Les points d'appui sont : (1,1) ; (2,0.5) ; (3,0.333

$L_0 = \frac{1}{2}(x-2)(x-3)$; $L_1 = -1(x-1)(x-3)$; $L_2 = \frac{1}{2}(x-1)(x-2)$

$P_2(x) = f(x_0)l_0 + f(x_1)l_1 + f(x_2)l_2$; $\boldsymbol{P_2(x) = \frac{1}{6}x^2 - x + \frac{11}{6}}$

2- $P_2(1.5) = 0.70833$; $P_2(2.5) = 0.375$

3- Err_abs_1= $|f(1.5)- P_2(1.5)| = 0.041$; Err_abs_1= $|f(2.5)- P_2(2.5)| = 0.025$

4- Intégration numérique :

X	1	1.5	2	2.5	3
P(x)	1	0.70833	0.5	0.375	0.333

n= 4 et h =0.5

$$\int_a^b f(x) = \frac{h}{2}\big(f(a) + f(b)\big) + h\sum_{i=1}^{n-1} f(a + i * h) \quad \Rightarrow$$

$$\int_1^3 P_2(x) = \frac{0.5}{2}(P_2(1) + P_2(3)) + 0.5\sum_{i=1}^{3} f(1 + i * 0.5)$$

$$\frac{0.5}{2}(P_2(1) + P_2(3)) + 0.5(P_2(1.5) + P_2(2) + P_2(2.5))$$

$$\int_1^3 \boldsymbol{P_2(x)} = \mathbf{1.12499}$$

5- 2 résultats analytiques : un tiré de $\int_1^3 f(x)$ et un autre tiré de $\int_1^3 P(x)$

$$\int_1^3 f(x) = \ln(3) - \ln(1) = 1.09861 - 0 = \mathbf{1.09861}$$

$$\int_1^3 P_2(x) = PrimP2(3) - PrimP2(1) \quad avec \quad PrimP2(x) = \frac{1}{18}x^3 - \frac{1}{2}x^2 + \frac{11}{6}x$$

$$\int_1^3 P_2(x) = 2.5 - 1.38889 = \mathbf{1.1111}$$

La valeur exacte de l'intégrale est celle calculée à partir de la primitive de *f* (1.09861).

La valeur analytique de l'intégrale tirée de la primitive du polynôme P (qui est une approximation de *f*) est égale à 1.1111.

La valeur numérique de l'intégrale de $P_2(x)$ (1.12499) calculée par la méthode des Trapèzes est proche de $\int_1^3 P_2(x)\ \ que \quad \int_1^3 f(x)$ (résultat logique)

Corrigé de l'exercice 2 :

Equation différentielle

$$\begin{cases} y'(x) = -y^2(x) \\ y(1) = 1 \end{cases}$$

1- si $y(x) = \frac{1}{x}$ alors $y'(x) = -\frac{1}{x^2}$ = - $(\frac{1}{x})^2 = -y^2(x)$ donc y(x)= $\frac{1}{x}$ vérifie l'équation différentielle ci-dessus

2- Processus de Runge-Kutta d'ordre 2 :

$$\begin{cases} \text{Étant donnée } y(1) = 1 \\ \text{Sachant que } y'(x_i) = -y^2(x) = \text{F}(x_i, y(x_i)) \\ \text{on calcul} \quad m_1 = y'(x_i) = F(x_i, y_i) \\ m_2 = y'(x_i + h) = F(x_i + h, y(x_i) + h * m_1) \end{cases}$$

Puis $y(x_i + h) = y(x_i) + \mathrm{h} * (\frac{m_1+m_2}{2})$

En appliquant cette méthode, on obtient sur [1.0 ,1.6] avec pas 0.2 :

x_i	1.0	1.2	1.4	1.6
$y(x_i)$	1.0	0.836	0.71764	0.62836

3- $y_{exact}(1.6) = 0.62499$; $y_{num}(1.6) = 0.62836$; err_abs(1.6)= 0.00337

4- Dérivation numérique :

par le schéma centré $y'(1.4)= \frac{y(1.6)- y(1.2)}{1.6-1.2} = \frac{0.62836-0.836}{0.4} = -0.5191$

Valeur exacte $y'(1.4)= -(\frac{1}{1.4})^2$ = -0.5102

5- $y''(1.4) =\frac{y(1.2)-2y(1.4)+y(1.6)}{0.2^2} = \frac{0.836-2*0.71764+0.62836}{0.04} = 0.727$

$y''(1.4)_{\text{exact}}= \frac{2*1.4}{1.4^4} = 0.72886$; err_abs=|0.72886-0.727| = 0.00186

Examen de Rattrapage (septembre 2011) proposé aux étudiants du département de Génie Mécanique USTO-MB

Durée : 2h

Question de Cours :

1- Quelle est la différence primordiale entre la méthode d'Euler et la méthode de Runge Kutta (d'ordre 2 ou d'ordre 4).
2- Pourquoi la méthode de Simpson est généralement plus exacte que la méthode des Trapèzes

Exercice 1 :

Soit la fonction *f*(x)=x^2

1- Calculer par la méthode de Simpson l'intégrale suivante **A1=**$\int_{0.4}^{0.6} f(x)dx$ avec n=4.
2- Comparer le résultat obtenu à la valeur analytique exacte.

Exercice 2 :

Soit la fonction *f*(x)= Ln(x)

1- Déterminer le Polynôme de Lagrange P(x) qui interpole *f* par les deux points d'appui : (0.5,-0.693) et (0.7, -0.357).
2- Calculer P(0.6), f(0.6) ainsi que l'erreur d'interpolation dans le point 0.6

Exercice 3 :

Soit à résoudre le système de Cauchy suivant :

$$\begin{cases} y' = -2xy^2 \\ y(1) = 1 \end{cases}$$

1- Vérifier que $y(x) = \frac{1}{x^2}$ est la solution exacte au problème posé.
2- Calculer les estimations de *y* sur l'intervalle [1 , 1.75] avec *h*=0.25 par la méthode d'Euler.
3- Comparer les estimations obtenues aux valeurs analytiques exactes.