DIRO
IFT 2425

EXAMEN INTRA

Max Mignotte

DIRO, Département d'Informatique et de Recherche Opérationnelle, local 2377
Http : //www.iro.umontreal.ca/∼mignotte/ift2425/
E-mail : mignotte@iro.umontreal.ca

Date : *26/02/2008*

I	Mesure d'Incertitude et Amplification d'Erreur (21 pts)
II	Erreur en Arithmétique Flottante (31 pts)
III	Méthode du Point Fixe et de Newton (33 pts)
IV	Factorisation LU (17 pts)
V	Interpolation (16 pts)
Total	118 points.

TOUS DOCUMENTS PERSONNELS, CALCULATRICES ET CALCULATEURS AUTORISÉS

I. Mesure d'Incertitude et Amplification d'Erreur (21 pts)

La loi de Poiseuille relie, par la formule ci-dessous, le débit volumique D (en $m^3 s^{-1}$) d'un fluide (écoulement laminaire) de viscosité ρ (en $Pa\,s$) dans une conduite cylindrique horizontale de longueur l (en m), de rayon intérieur R (en m) avec la différence de pression P (en Pa) créé par ce fluide aux deux extrémités de ce conduit

$$P = \frac{8\,\rho\,l}{\pi R^4}\,D$$

On suppose une valeur exacte pour $\rho = 10^{-3}$ et $\pi = 3.14159$. Pour chacune des autres variables, on a la mesure approximée et l'incertitude suivante,

R	$= 0.05\ m$	avec une erreur relative de 1%
l	$= 10.0$ mètres	tout les chiffres donnés étant considéré comme significatif “exact” (cse)
D	$\in [4.2 \cdot 10^{-3}, 4.6 \cdot 10^{-3}]$	i.e., D est une valeur élément de cette intervalle de confiance

On vous demande de :

1. Exprimer, pour chacune de ces trois variables, l'incertitude que l'on a sur elle en la mettant sous la forme $x = x^\star \pm \Delta x$ où $x^\star$ et Δx seront déterminés.
   <**6 pts**>
2. Donner l'approximation de la différence de pression P, et l'incertitude sur P (i.e., ΔP) par la méthode de propagation d'erreur (utilisant l'approximation de Taylor de la fonction au premier ordre).
   <**5 pts**>
   Souligner les chiffres significatifs “exacts” (cse) de ce résultat. Arrondir cette approximation aux nombres de cse adéquat.
   <**2 pts**>
   Donner finalement l'intervalle de confiance dans lequel se trouverait la vraie valeur de V obtenue par cette méthode (en utilisant les approximations non arrondies).
   <**2 pts**>
3. Peut on utiliser la méthode de la fourchette pour obtenir cette incertitude ? Si oui expliquer pourquoi et utiliser cette méthode (de la fourchette) pour obtenir un intervalle de confiance dans lequel se trouverait la vraie valeur de P et en déduire l'incertitude résultante (i.e., erreur absolue) sur P que l'on obtiendrait par cette méthode.
   <**6 pts**>

Réponse

1.

L'erreur relative de la première variable nous permet de calculer l'erreur absolue que l'on fait sur cette mesure, i.e., $\Delta R = 0.0005$ et donc d'écrire $R = 0.05 \pm 0.0005$. <**2 pts**>

Le fait que le chiffre des dixièmes “0” (de rang -1) soit un chiffre significatif nous permet de trouver une borne supérieure $\Delta l = 0.5 \times 10^{-1}$ de l'erreur absolue d'estimation faîte sur cette mesure et donc d'écrire $l = 10.0 \pm 0.05$. <**2 pts**>

Finalement, la connaissance, pour la mesure D d'un intervalle de confiance, nous permet d'écrire ; d'une part que la valeur approximée pour cette variable sera le centre de cette intervalle et d'autre part que son erreur absolue est la demi largeur de cet intervalle, i.e., $L = 4.4 \cdot 10^{-3} \pm 0.2 \cdot 10^{-3}$. <**2 pts**>

2.

On a pour valeur approchée de P

$$P^* = \frac{8 \times 10^{-3} \times 10}{3.14159 \times 0.05^4} \times 4.4 \times 10^{-3} = 17.93 \quad \text{Pa.s} \qquad < \textbf{2 pts} >$$

Pour calculer l'incertitude de P, on doit calculer la différentielle $\Delta P(R, l, D)$, i.e.,

$$\Delta P(R, l, D) = \left| \frac{-32\, \rho\, l\, D}{\pi\, R^5} \right| \Delta R + \left| \frac{8\, \rho\, l\, D}{\pi\, R^4} \right| \Delta l + \left| \frac{8\, \rho\, l}{\pi\, R^4} \right| \Delta D. \qquad < \textbf{2 pts} >$$

Soit,

$$\begin{aligned}
\Delta P(R, l, D) &= \left| \frac{-32 \times 10^{-3} \times 10 \times 4.4 \times 10^{-3}}{3.14159 \times (0.05)^5} \right| \times 0.0005 + \left| \frac{8 \times 10^{-3} \times 10 \times 4.4 \times 10^{-3}}{3.14159 \times (0.05)^4} \right| \times 0.5 \times 10^{-1} \\
&+ \left| \frac{8 \times 10^{-3} \times 10}{3.14159 \times (0.05)^4} \right| \times 0.2 \times 10^{-3} \\
&= 0.7171 + 0.8964 + 0.8149 \\
&\approx 2.428 \qquad < \textbf{2 pt} >
\end{aligned}$$

On obtient donc $P \approx 17.93 \pm 2.428$. Comme $2.428 < 0.5 \times 10^1$, le rang du dernier chiffre significatif est 1 et on peut écrire $P \approx \underline{1}7.93 \pm 2.4$ <**1 pt**>. Si on arrondi finalement cette estimation aux nombres de cse, adéquat, on obtient

$$P \approx 18 \pm 2.5 \quad (\text{en Pa.s}). \quad < \textbf{1 pt} >$$

$\boxed{\textbf{3.}}$

La méthode de la fourchette peut être utilisée ici car la fonction $P(R, l, D) = l \times L \times h$ est strictement monotone sur chacun des intervalles dans lequel se trouve les variables R, l, et D. Il est donc possible, en utilisant les valeurs minimales pour l et D et la valeur maximale de R de déterminer la borne minimale pour P et inversement. <**2 pts**>

$$\begin{aligned}
P_{\text{min}} &= \frac{8 \times 10^{-3} \times 9.95}{3.14159 \times 0.0505^4} \times 4.2 \times 10^{-3} \\
&= 16.36
\end{aligned}$$

$$\begin{aligned}
P_{\text{max}} &= \frac{8 \times 10^{-3} \times 10.05}{3.14159 \times 0.0495^4} \times 4.6 \times 10^{-3} \\
&= 19.61
\end{aligned}$$

<**2 pts**>

La valeur approximée est donc le centre de cette intervalle de confiance et l'incertitude sur P serait la demi largeur de cette intervalle. Ce qui nous permet d'écrire $P \approx 17.98 \pm 1.625$. <**2 pts**>

II. Erreur en Arithmétique Flottante (27 pts)

1. Soit la série suivante

$$\sum_{n=1}^{\infty} \frac{1}{n}$$

(a) Expliquer pourquoi cette série divergente devient convergente en arithmétique flottante (i.e., identifier et citer le problème numérique associé).
<**3 pts**>.

(b) En supposant que l'on prenne une (vieille) calculatrice qui ferait ces calculs en notation flottante simple précision utilisant

   i. l'arrondissement
   ii. sans double précision pour les calculs intermédiaires
   iii. pour laquelle l'epsilon machine est 2^{-127}

et que l'on programme cette série. Donner la valeur de convergence donnée par cette calculatrice en expliquant ce qui va se passer (vous pouvez exprimer cette valeur a convergence comme une somme finie de termes).
<**3 pts**>.

2. Soit la séquence définie récursivement, pour tout x_0, par

$$x_{n+1} = \ln\Big(\sqrt{x_n^2+1} - x_n\Big)$$

(a) Identifiez, pour quelle valeur de x, et le type de problème numérique majeur auquel sera confronté cette formule itérative.
<**2 pts**>

(b) Arrangez cette formule de telle façon qu'elle évite donc le maximum de perte de précision en arithmétique flottante.
<**3 pts**>

3. Évaluer l'expression suivante (nombre infinie de termes)

$$x = \ln\Bigg(2 + \ln\Bigg(2 + \ln\big(2 + \ln(2 + \dots$$

Pour cela, prenez l'exponentiel des deux membres de cette égalité et en déduire une équation du type $f(x) = 0$ pour laquelle x est solution et utiliser la méthode itérative de Newton en partant de $x_{[0]} = 1$ (Nota : ici l'indice indique l'itération). Calculer les 4 premiers termes de cette méthode itérative.
<**7 pts**>

4. Soit la routine suivante,

ALGO

nrj[0], nrj[1]	2 flottants
Sum	1 flottant
float Energy(int,..)	Fonction renvoyant un flottant positif

- **for** $n = 0$ *to* 1 **do**
  - nrj[n] = exp(−Energy(n, ..))
- Sum ← (nrj[0] + nrj[1])
- **for** $n = 0$ *to* 1 **do**
  - nrj[n] ← nrj[n]/ Sum
- Si (nrj[0] > nrj[1]) Alors Faire <Action 1>
  Autrement Faire <Action 2>

Ce programme plante. L'ordinateur signale une division par zéro. En utilisant un debugger on s'aperçoit que le flottant Sum prend souvent la valeur 0 alors que l'on s'attend a une valeur positive. De plus le fait de mettre devant la division (si Sum $\neq 0$ supprime l'erreur de NaN mais n'arrange pas du tout le fait que le programme n'a pas le comportement désiré). Identifiez le problème numérique qui est l'origine de ce bug (donner par exemple un cas de figure ou le problème ne se pose pas analytiquement mais se pose numériquement) et proposez une solution pour le résoudre en proposant un pseudo-code sans erreur numérique.
<13 pts>

Réponse

$\boxed{\textbf{1.(a)}}$

Pour une certaine valeur de n, le terme $1/n$ va devenir plus petit que l'epsilon machine, i.e., la plus petite valeur représentable dans la machine. Au moment ou il deviendra plus petit que la moitié de l'epsilon machine, l'ordinateur l'arrondira a zéro faisant ainsi une erreur d'*underflow* qu'il fera pour toute les autres termes de cette série. Du coup la série infinie deviendra une série avec un nombre finie de termes et celle-ci deviendra en arithmétique flottante (utilisant un nombre finie de chiffres pour la mantisse utilisé dans ce format) ainsi convergente.
<3 pts>

Nota : Sur mon ordinateur, Cette série converge vers la valeur 15.403682708740234375 lorsqu'on utilise des *floats* et vers la valeur 30.3.... lorsqu'on utilise des *doubles*.

$\boxed{\textbf{1.(b)}}$

Il suffit de calculer la valeur de n qui produira un terme plus petit que la moitié de l'epsilon machine, i.e., la valeur de n pour laquelle $\frac{1}{n} < 2^{-128}$. On trouve facilement $n = 3.403 \times 10^{38}$. La valeur a convergence de cette série est donc $\sum_{n=1}^{n=3.403\times 10^{38}} \frac{1}{n}$ qui sera un nombre finie.
<3 pts>

$\boxed{\textbf{2.}}$

On aura un problème d'annulation de cse (perte de précision) due à la soustraction de deux nombres approximés quasi égaux lorsque x va tendre vers (plus) l'infinie **<2 pt>**. Pour éviter ce problème, on peut écrire cette relation itérative de telle façon que plus aucune soustraction de deux nombres incertains (i.e., imprécis) quasi égaux apparaissent. On utilise le conjugué de l'expression $\sqrt{x^2+1} - x$ que l'on multiplie au numérateur et au dénominateur dans la première expression. En utilisant les propriétés du logarithme, on trouve facilement,

$$x_{n+1} \quad = \quad \ln\Big(\sqrt{x_n^2+1} - x_n\Big) = \ln\Big(\frac{1}{\sqrt{x_n^2+1} + x_n}\Big) = -\ln\Big(\sqrt{x_n^2+1} + x_n\Big)$$

i.e., une expression qui ne fait plus intervenir de soustraction de nombres imprécis quasi-égaux.
<3 pts>

Nota : Sur mon ordinateur, l'utilisation de la relation récursive brute pose problème dès la 46341 ième itération en *floats* et en *doubles*, tandis que la relation itérative ne faisant pas apparaître la soustraction de deux nombres approximés quasi égaux ne pose aucun problème.

$\boxed{\textbf{3}}$

On s'aperçoit facilement que $\exp(x) = 2 + x$ $<$**2 pts**$>$ et donc que x est solution de l'équation $\exp(x) - x - 2 = 0$. La relation itérative de Newton nous permet d'écrire,

$$\begin{aligned} x_{[n+1]} &= x_{[n]} - \frac{f(x_{[n]})}{f'(x_{[n]})}, \qquad k = 0, 1, \ldots \\ &= x_{[n]} - \frac{\exp(x_{[n]}) - x_{[n]} - 2}{\exp(x_{[n]}) - 1} \qquad < \textbf{3 pts} > \end{aligned}$$

ce qui nous permet de trouver,

$$\begin{aligned} x_{[0]} &= 1.0, \\ x_{[1]} &= 1.164, \\ x_{[2]} &= 1.146, \\ x_{[3]} &= 1.146, \\ &\ldots \end{aligned}$$

qui semble converger vers $x = 1.146$ très rapidement.

$<$**2 pts**$>$

$\boxed{\mathbf{4}}$

Pour des valeur de flottant, renvoyé par la fonction FLOAT ENERGY(..) qui serait trop importante, le risque d'*underflow* est grand et le programme risque d'enregistrer *zéro* dans nrj[0] et nrj[1], mettant ainsi un NaN (Not a Number) dans nrj[0] et nrj[1].

Par exemple imaginons que lors de la première boucle, nrj[0] = $\exp(-500)$ et que nrj[1] = $\exp(-501)$. Analytiquement, cela correspond a des valeurs positifs (très petites mais différentes de zéro) dans nrj[0] et nrj[1], numériquement cela correspondra a deux valeurs nulles (car plus petite que la moitié de l'epsilon machine).

Lors de la deuxième boucle, analytiquement, la variable Sum contient la valeur $\exp(-500) + \exp(-501)$ (différente de zéro). Numériquement cela correspondra a zéro.

Lors de la troisième boucle, analytiquement nrj[0] = $\exp(-500)/(\exp(-500) + \exp(-501)) = 0.7311$ et l'autre nrj[1] = $\exp(-501)/(\exp(-500) + \exp(-501)) = 0.2689$ alors que numériquement on aura une forme indéterminée du type 0/0.

$<$**6 pts**$>$

Une façon de résoudre le problème est de proposer le code suivant :

ALGO

nrj[0], nrj[1]	2 flottants
Sum	1 flottant
Min	1 flottant
float Energy(int,..)	Fonction renvoyant un flottant positif

- **for** $n = 0$ *to* 1 **do**
  - nrj[n] = Energy(n, ..)
- Min $\leftarrow$ (nrj[0], nrj[1])
- **for** $n = 0$ *to* 1 **do**
  - nrj[n] = exp(Min $-$ Energy(n, ..))
- Sum $\leftarrow$ (nrj[0] + nrj[1])

etc.

$<$**7 pts**$>$

Nota1 : Une autre possibilité est de considérer le logarithme de l'énergie pour s'affranchir de ce probleme.

Nota2 : Ces quelques lignes de codes sont au coeur d'un des algorithmes d'optimisation stochastique le plus connu qui existe actuellement et qui se nomme le *recuit simulé* ou de l'algorithme de simulation que l'on appelle algorithme de *métropolis*.

III. Méthode du Point Fixe et de Newton (33 pts)

On se propose de trouver numériquement une valeur approchée d'une des racines de la fonction,

$$f(x) = e^x - 3x^2 \tag{1}$$

1. **Méthode dichotomique ou de la bissection**
   (a) Montrer qu'il existe une racine unique r_1 pour cette Eq. (1) dans l'intervalle $J = [-1/2, 0]$.
   <**4 pts**>
   (b) En utilisant 3 itérations de la méthode dichotimique, donner une première approximation de cette racine comprise entre $J = [-1/2, 0]$. Prenez comme valeur estimée a la fin des 3 itérations, le milieu de l'intervalle que vous trouverez et laisser l'approximation de cette racine r sous la forme d'une fraction
   <**5 pts**>
2. **Méthode du point fixe**
   Pour cette question, on supposera que la valeur de la racine trouvée par la méthode dichotomique précédente nous a donné $r \approx -\frac{15}{32}$.
   Pour trouver une meilleure précision numérique de cette racine, on se propose d'étudier les trois suites itératives ou méthodes de point fixes suivant ; $x_{n+1} = \Upsilon_i(x_n)$ pour $i = 1, 2, 3$,
   (a) $\Upsilon_1(x) = -\sqrt{\frac{e^x}{3}}$
   (b) $\Upsilon_2(x) = -\left(\frac{e^x - 3x^2 - 3.4x}{3.4}\right)$
   (c) $\Upsilon_3(x) = e^x - 3x^2 + x$

   En vous aidant aussi de la racine analytiquement trouvée à la question précédente, établissez laquelle ou lesquelles de ces suites itératives sont convergentes si on prend x_0 dans J.
   <**14 pts**>
3. **Méthode de Newton**
   (a) Soit $\Upsilon_4(x)$, la fonction intervenant dans la méthode itérative de Newton pour la résolution de la racine r de l'équation (1) dans J. Donner $\Upsilon_4(x)$ ainsi que la relation itérative $r_{n+1} = \Upsilon_4(r_n)$.
   <**4 pts**>
   (b) En déduire une valeur approchée de r après 3 itérations (i.e., donner r_1, r_2, r_3) (avec $r_0 = -1/2$).
   <**4 pts**>
   (c) Quel est l'ordre de convergence de cette méthode ?
   <**2 pts**>

Réponse

$\boxed{\textbf{1a.}}$

L'étude des variations de la fonction f sur $J = [-1/2, 0]$ montre que la fonction est continue et décroissante sur cet intervalle (donc monotone) ($f'(x) = e^x - 6x$ et $f'(x) > 0 \;\; \forall\, x \in [-1/2, 0]$). <**2 pt**>

De plus, on a $f(-0.5) = -0.14 < 0$ et $f(0) = e^0 - 3 \times 0 = 1 > 0$ (donc $f(-1/2)f(0) < 0$), il existe donc une racine r unique dans cet intervalle. <**2 pts**>

1b.

On utilise donc l'intervalle de départ $J = [-1/2, 0]$. en première itération de la méthode dichotimique, testons la valeurs milieu de cet intervalle, i.e., de $f(1/4) = e^{-0.25} - 3 \times (0.25)^2 \approx 0.59 > 0)$, l'intervalle a considérer (qui encadre de nouveau la racine) est donc $[-1/2, -1/4]$. On recommence en estimant la fonction pour la valeur milieu de cet intervalle, i.e. $f(-3/8) = e^{-3/8} - 3 \times (-3/8)^2 \approx 0.2654 > 0)$ et donc le nouvel intervalle est $[-1/2, -3/8]$. Finalement, on teste $f(-7/16) = e^{-7/16} - 3 \times (-7/16)^2 \approx 0.07143 > 0)$ et on trouve le nouvel intervalle est $[-1/2, -7/16]$.

Prenons comme approximation de cette racine (comme nous le précise l'énoncé) la valeur milieu de cet intervalle et laissons la sous forme fractionnaire, i.e., $r_1 = -\frac{15}{32}$.

<**5 pts**>

2.

On va tout d'abord calculer les dérivées $\Upsilon'_1(x)$, $\Upsilon'_2(x)$ et $\Upsilon'_3(x)$ pour vérifier quelles sont la valeur de leur dérivé au point correspondant à la racine (trouvée analytiquement i.e., en $x = r \approx -\frac{15}{32}$). si $|\Upsilon'_i(x = -\frac{15}{32}| > 1$ alors cela nous dira tout de suite que la méthode diverge (puisqu'on doit trouver un intervalle J dans lequel r et x_0 sont contenu et pour lequel $\forall x \in J, \quad |\Upsilon'_i(x)| < 1$). On trouve,

1. $\Upsilon'_1(x) = -\frac{1}{2\sqrt{3}}\, e^{x/2}$
   <**2 pt**>
   et on a $\Upsilon'_1(r) = -\frac{1}{2\sqrt{3}}\, e^{\frac{-15}{64}} \approx 0.2284 < 1$ donc la suite pourrait converger ... vérifions :
   Sur $J = [-1/2, 0]$, Υ'_1 est décroissante et $\forall x \in J, \Upsilon'_1(x) \in\approx [-0.1751, -0.2887]$ et donc $\forall x \in J$ $|\Upsilon'_1(x)| < 1$ donc la suite converge.
   <**4 pts**>
2. $\Upsilon'_2(x) = -\left(\frac{e^x - 6x - 3.4}{3.4}\right)$
   <**1 pt**>
   et on a $\Upsilon'_2(r) \approx 0.011 < 1$. La suite pourrait converger ... vérifions :
   Sur $J = [-1/2, 0]$, Υ'_2 est croissante et $\forall x \in J, \Upsilon'_2(x) \in\approx [0.06, 0.7059]$ et donc $\forall x \in J$ $|\Upsilon'_2(x)| < 1$ donc la suite converge.
   <**3 pts**>
3. $\Upsilon'_3(x) = e^x - 6x + 1$
   <**1 pt**>
   et on a $\Upsilon'_3(r) = e^{-\frac{15}{32}} - 6 \times (-\frac{15}{32}) + 1 \approx 4.438 > 1$ donc la suite diverge.
   <**3 pts**>

3a.

La fonction $f(x)$ est dérivable sur J et on a,

$$g(x) = x - \frac{f(x)}{f'(x)} = x - \frac{e^x - 3x^2}{e^x - 6x}.$$

On a donc la formule itérative suivante,

$$r_{n+1} = r_n - \frac{e^{r_n} - 3r_n^2}{e^{r_n} - 6r_n}$$

< **4 pts** >

3b.

En partant de $r_0 = -0.5$, on a, $r_n = g_4(r_{n-1})$ et,

$$\begin{array}{rcl} r_0 & = & -0.5 \\ r_1 & = & -0.4602 \\ r_2 & = & -0.459 \\ r_3 & = & -0.459 \quad < \mathbf{4\ pts} > \end{array}$$

Note : Cela semble converger très vite vers la valeur -0.459.

3c.

Ordre 2, convergence quadratique. <**2 pts**>

IV. Factorisation LU (17 pts)

1. Décomposer la matrice A en produit LU (sans permutation) par la méthode de la factorisation directe.
   <**8 pts**>

$$A = \begin{pmatrix} 1 & 4 & 5 \\ 2 & 14 & 58 \\ 3 & 19 & 80 \end{pmatrix}$$

2. Calculer le déterminant de A.
   <**3 pts**>
3. Rappeler quelle est la méthode la plus rapide, vu en cours, pour calculer l'inverse d'une matrice carré de taille n et rappeler sa complexité algorithmique.
   <**6 pts**>

Réponse

1.

Par factorisation directe, on trouve,

$$A = \begin{pmatrix} 1 & 4 & 5 \\ 2 & 14 & 58 \\ 3 & 19 & 80 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 2 & 6 & 0 \\ 3 & 7 & 9 \end{pmatrix}}_{L<\mathbf{4\ pts}>} \underbrace{\begin{pmatrix} 1 & 4 & 5 \\ 0 & 1 & 8 \\ 0 & 0 & 1 \end{pmatrix}}_{U<\mathbf{4\ pts}>}.$$

2.

$$\begin{array}{rcl} \det(\mathrm{A}) & = & \det(\mathrm{L}) \times \det(\mathrm{U}) \\ & = & 1 \times (6 \times 9) \\ & = & 54 \quad < \mathbf{3\ pts} > \end{array}$$

3.

Il s'agit de l'élimination de Gauss pour une matrice n-ement augmentée des vecteurs unitaires (ou augmentée de la matrice identité d'ordre n) suivi de n substitutions arrières, pour une complexité algorithmique totale de $O(\frac{n^3}{3}) + n \times O(\frac{n^2}{2}) = 0(\frac{5\,n^3}{6})$.

<**6 pts**>

V. Interpolation (16 pts)

On considère la fonction suivante,

$$f(x) = y = \text{arctg}x$$

sur l'intervalle $[-1 \;\; 1]$.

1. Appliquer la formule de Lagrange pour trouver un polynôme P de degré deux qui interpole la fonction f aux points $x_0 = -1$, $x_1 = 0$ et $x_2 = 1$.
   <6 pts>
2. Soit la fonction erreur définie par $\varepsilon(x) = \text{arctg}x - P(x)$.
   Montrer que cette erreur (d'interpolation) est maximale sur l'intervalle $[-1 \;\; 1]$ en x_{+e} et $x_{-e} = \pm\sqrt{\frac{4}{\pi} - 1} \approx \pm 0.53$.
   <2 pts>
3. En remarquant que la fonction d'erreur est impaire, trouver le polynôme d'ordre deux Q_+ qui approxime cette fonction d'erreur ε sur l'intervalle $[0 \;\; 1]$, en utilisant les points $x_0 = 0$, $x_1 = x_{+e}$ et $x_2 = 1$, puis le polynôme d'ordre deux Q_- qui approxime cette fonction d'erreur sur l'intervalle $[-1 \;\; 0]$, en utilisant les points $x_0 = -1$, $x_1 = x_{-e}$ et $x_2 = 0$.
   Proposez ensuite une expression pour le polynôme $Q(x)$ d'ordre deux qui approxime cette fonction d'erreur sur l'intervalle $[-1 \;\; 1]$.
   <6 pts>
4. Déduisez en donc une approximation plus précise pour la fonction arctg sur l'intervalle $[-1 \;\; 1]$.
   <2 pts>

Réponse

$\boxed{\textbf{1.}}$

Pour le tableau de points suivant

x	-1	0	1
y	$\pi/4$	0	$\pi/4$

<3 pts>

On obtient le polynôme de Lagrange d'ordre deux suivant

$$P(x) \;\; = \;\; \frac{x(x-1)}{-1 \times -2}\left(-\frac{\pi}{4}\right) + \frac{x(x+1)}{1 \times 2}\left(\frac{\pi}{4}\right) = \frac{\pi}{4}\,x \quad < \textbf{3 pts} >$$

Nota : Ici, le polynôme de collocation d'ordre 2 qui approxime la fonction arctg est d'ordre 1.

$\boxed{\textbf{2.}}$

Erreur maximale en $x_{\pm e}$ tel que $\varepsilon' = \frac{1}{1+x^2} + \frac{\pi}{4} = 0$, i.e., en x_{+e} et $x_{-e} = \pm\sqrt{\frac{4}{\pi} - 1} \approx 0.53$. **<2 pts>**

$\boxed{\textbf{3.}}$

Sur $[0 \;\; 1]$, on a

x	0	0.53	1
y	0	0.071	0

<2 pts> et donc $Q_+(x) = \frac{x(x-1)}{(0.53)(0.53-1)} \times 0.071 \approx 0.285\,x\,(1-x)$.

Sur $[-1 \;\; 0]$, on a

x	-1	-0.53	0
y	0	-0.071	0

et donc $Q_-(x) = \frac{x(x+1)}{(-0.53)(-0.53+1)} \times -0.071 \approx 0.285\, x\,(1+x)$.

<2 pts>

Sur $[-1 \;\; 1]$, on a donc $\quad Q(x) \approx 0.285\, x\,(1-|x|)$

<2 pts>

$\boxed{\textbf{4.}}$

On a donc finalement l'approximation

$$\mathrm{arctg}x \approx \frac{\pi}{4}\, x + 0.285\, x\,(1-|x|)$$

Nota : Cet exercice est inspiré de l'article “Efficient Approximations for the Arctangent Function” de Sreeraman Rajan, Sichun Wang, Robert Inkol and Alain Joyal et publié dans le magazine IEEE Signal Processing magazine, mai 2006 dans la rubrique “dsp TIPS & TRICKS” pages 108-111.
Il nous donne un exemple dans lequel certaines fonctions utilisées dans les systèmes informatiques actuels (ordinateurs et dsp) utilisent non pas la série de Taylor mais quelquefois les méthodes d'interpolation basé sur le polynôme de collocation pour approximer une fonction. Dans le cas de la fonction arc-tangent, la méthode utilisant la série de Taylor est peu efficace car celle ci converge lentement (pour les valeurs proche de 1).