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 : *24/02/2009*

I	Mesure d'Incertitude et Amplification d'Erreur (29 pts)
II	Erreur en Arithmétique Flottante (29 pts)
III	Méthode du Point Fixe et de Newton (25 pts)
IV	Factorisation LU (17 pts)
V	Interpolation (17 pts)
Total	117 points.

Tous documents personnels, calculatrices et calculateurs autorisés

I. Mesure d'Incertitude et Amplification d'Erreur (29 pts)

On aimerait faire un programme informatique qui résoud automatiquement les équations du second degré du style $ax^2 + bx + c = 0$. A cette fin, le programme demandera donc à l'utilisateur de rentrer au clavier les valeurs a, b et c et affichera ensuite le résultat, c'est à dire les deux racines (x_1 et x_2) de cette équation, en utilisant la formule classique suivante

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \tag{1}$$

$$x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a} \tag{2}$$

On suppose que l'on rentre les valeurs suivantes $a = 1.0$, $b = 4.00$ et $c = 2.0$. On supposera aussi que la valeur de a est connue sans approximation et que les deux autres valeurs sont connues avec une erreur relative de 1%.

On vous demande de :

1. Soulignez les chiffres significatifs "exact" (cse) existant dans la valeur approximée de b et c.
   <**4 pts**>
2. Donner la valeur approximée de x_1 et x_2 donnée par ce programme (i.e., x_1^* et x_2^*).
   <**2 pts**>
3. Donner la borne supérieure de l'erreur absolue ou incertitude totale de x_1 et x_2 (i.e., Δx_1 et Δx_2) par la méthode de propagation d'erreur (utilisant l'approximation de Taylor de la fonction au premier ordre).
   <**10 pts**>
4. En déduire l'erreur relative que fait ce programme dans l'estimation numérique de x_1 et x_2.
   <**3 pts**>
5. Montrer qu'une autre possibilité serait d'exprimer x_1 sous cette forme

$$x_1 = \frac{-2c}{b + \sqrt{b^2 - 4ac}} \tag{3}$$

   <**2 pts**>
6. Donner l' incertitude totale de x_1 (i.e., Δx_1) par la méthode de propagation d'erreur utilisant l'approximation de Taylor de la fonction au premier ordre pour l'expression (3).
   <**5 pts**>
7. Des deux expressions pour x_1 (Equations (1) et (3)) laquelle peut conduire à des problèmes d'erreurs numériques lorsque b sera très grand par rapport à a et c. Justifier votre réponse.
   <**3 pts**>

Réponse

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

On a respectivement pour b et c une erreur absolue de $\Delta b = 0.04 < 0.5 \times 10^{-1}$ et $\Delta c = 0.02 < 0.5 \times 10^{-1}$. Ce qui nous donne respectivement, $b^\star = \underline{4.0}0$, $c^\star = \underline{2.0}$.
<**4 pts**>

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

On a pour valeur approchée

$$x_1^* = \frac{-4+\sqrt{4^2-4\times1\times2}}{2} = -2+\sqrt{2} \approx -0.58578 \qquad \text{et} \qquad x_2^* = \frac{-4-\sqrt{4^2-4\times1\times2}}{2} = -2-\sqrt{2} \approx -3.41421$$

<2 pts>

$\boxed{\mathbf{3.}}$

Pour calculer l'incertitude (ou la borne supérieure de l'erreur absolue) de x_1 et x_2, on doit calculer la différentielle Δx_1 et Δx_2 (fonction des deux variables approximées b et c et de la valeur précise a), i.e., (en posant $\Delta = b^2 - 4ac$)

$$\begin{aligned} \Delta x_1(b,c) &= \left|\frac{1}{2a}\Big(-1+\frac{b}{\sqrt{\Delta}}\Big)\right|\Delta b + \left|\frac{-1}{\sqrt{\Delta}}\right|\Delta c \\ \Delta x_2(b,c) &= \left|\frac{1}{2a}\Big(-1-\frac{b}{\sqrt{\Delta}}\Big)\right|\Delta b + \left|\frac{1}{\sqrt{\Delta}}\right|\Delta c \end{aligned}$$

<8 pts>

Avec $\Delta b = 0.04$ et $\Delta c = 0.02$, soit puisque $\sqrt{\Delta} = \sqrt{4^2 - 4\times 2} \approx 2.828427$

$$\begin{aligned} \Delta x_1 &= \left|\frac{1}{2}\Big(-1+\frac{4}{2.828427}\Big)\right|\times 0.04 + \left|\frac{-1}{2.828427}\right|\times 0.02 \approx 0.01535 \\ \Delta x_2 &= \left|\frac{1}{2}\Big(-1-\frac{4}{2.828427}\Big)\right|\times 0.04 + \left|\frac{1}{2.828427}\right|\times 0.02 \approx 0.05535 \end{aligned}$$

<2 pts>

$\boxed{\mathbf{4.}}$

On obtient donc une erreur relative de

$$\Delta x_1/x_1^* \approx 0.01535/0.58578 = 2.62\,\% \quad \text{et} \quad \Delta x_2/x_2^* = 0.05535/3.41421 \approx 1.62\,\%$$

<3 pts>

$\boxed{\mathbf{5.}}$

$$\frac{-b+\sqrt{b^2-4ac}}{2a} = \frac{-b+\sqrt{b^2-4ac}}{2a}\cdot\frac{b+\sqrt{b^2-4ac}}{b+\sqrt{b^2-4ac}} = \frac{-b^2+b^2-4ac}{2a(b+\sqrt{b^2-4ac})} = \frac{-2c}{b+\sqrt{b^2-4ac}}$$

<2 pts>

$\boxed{\mathbf{6.}}$

$$\begin{aligned} \Delta x_1(b,c) &= \left|-2c\frac{\Big(-1+\frac{b}{\sqrt{\Delta}}\Big)}{(b+\sqrt{\Delta})^2}\right|\Delta b + \left|\frac{-2(b+\sqrt{\Delta})-\frac{4c}{\sqrt{\Delta}}}{(b+\sqrt{\Delta})^2}\right|\Delta c \\ &\approx 0.03539 + 0.0064644 \approx 0.0418544 \end{aligned}$$

<5 pts>

$\boxed{\mathbf{7.}}$

Lorsque b sera très grand par rapport à a et c, l'expression de x_1 donnée par l'équation (1) aura une annulation de cse due à la soustraction de deux nombres quasi égaux approximées.

<3 pts>

II. Erreur en Arithmétique Flottante (29 pts)

1. Soit l'expresion numérique suivante

$$f(x) = \ln(x + \sqrt{x^2 + 1})$$

   (a) Identifiez, pour quelle valeur de x, et le type de problème numérique majeur auquel sera confrontée le calcul de cette expression.
   <3 pts>

   (b) Calculer $f(x) + f(-x)$.
   <2 pts>

   (c) En s'aidant de la question précédente, trouver donc une façon de calculer $f(-10^{10})$ (et faîte ce calcul) qui ne fera pas intervenir le type d'erreur numérique mis en évidence à la première question.
   <3 pts>

2. Soit la routine suivante

```
                    ALGO

| matpds[...][...]        Tableau 2D de flottants
| nrj                     flottant

• nrj= 0.0
• for i = 0 to 500000,  i++   do
  |  ...
  |  ...
  |  for l = 0 to 5000,  l++ do
  |  |  ...
  |  |  ...
  |  |  if  COND1  then   nrj+=matpds[l][i]
  |  |_ else   nrj-=matpds[l][i]
  |_
• return nrj
```

dans laquelle les éléments du tableau MATPDS[][] sont des flottants (simple précision) inférieurs à 1.0 et la COND1 est remplie 9 fois sur 10.
Ce programme marche, par contre, on s'aperçoit que l'on a une perte de précision au niveau de la valeur flottante NRJ renvoyée par cette procédure.
On aimerait résoudre ce problème de perte de précision tout en gardant le calcul en float plutôt que d'utiliser des DOUBLE qui augmenteraientt notre coût calculatoire par un facteur 2. Identifiez le problème numérique qui est à l'origine de cette perte de précision et proposer une solution, c'est à dire un nouveau pseudo-code, permettant de renvoyer une valeur NRJ plus précise (i.e., pour laquelle cette perte de précision numérique sera atténuée) en conservant des valeurs flottantes (FLOAT) simple précision. Justifier votre réponse.
<10 pts>

3. Expliquer pourquoi le calcul numérique de ces expressions

   (a) $\frac{\cos(x)}{\sqrt{x^2+1}-1}$

   (b) $-1 + \exp x$

peuvent conduire à des problèmes d'erreurs numériques (identifiez aussi pour quelle valeur de x). Expliquer pourquoi (i.e., identifier et citer le problème numérique associé) et proposer une formule équivalente qui permettrait d'éviter ces problèmes numériques et qui permettrait d'augmenter la précision des calculs de ces deux expressions.
<6 pts>

4. En cours, l'Epsilon machine a été définie comme étant *la plus petite valeure représentable par la machine* en numérotation flottante (i.e., la plus petite valeur qui ajoutée à 1.0 donnera un résultat numériquement différent de 1.0).
   Il existe une deuxième définition possible de l'Epsilon machine (et qui donnera un résultat légèrement différent de celui donnée par la première définition) qui consiste à dire que
   l'Epsilon machine est la difference entre la plus petite valeur flottante supérieur à 1.0 *et* 1.0.
   Trouver donc cette valeur correspondant à cette deuxième définition. Utiliser à cette fin une représentation flottante binaire avec un format du type $\pm 0.xxxx\ldots$ (avec 24 x, i.e , 24 bits pour la mantisse) dont un bit caché permettant de créer le bit de signe ("0" pour le signe "+") et un exposant sur 8 bits permettant d'exprimer les valeurs de l'exposant de -126 à 127.
   <5 pts>

Réponse

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

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 moins l'infinie $x \to -\infty$.
<3 pts>

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

$$f(x) + f(-x) = \ln(x + \sqrt{x^2+1}) + \ln(-x + \sqrt{x^2+1}) = \ln(x^2 + 1 - x^2) = \ln(1) = 0$$

<2 pts>

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

La question précédente nous montre que $f(x) = -f(-x)$ et donc le calcul de $f(-10^{10})$ [qui pose le problème numérique expliquée en 1.(a)] pourra être remplacée par le calcul de $-f(10^{10})$ qui ne pose, cette fois ci, moins de problème d'erreur numérique.

Le calcul de $f(-10^{10})$ sur ma calculatrice me donne le résultat numériquement faux $f(-10^{10}) =$ Ma ERROR et le calcul (mathématiquement équivalent) de $-f(10^{10})$ me donne le résultat très précis $-f(10^{10}) = -23,71899811$.
<3 pts>

$\boxed{\textbf{2}}$

Le problème qui engendre cette perte de précision provient de l'erreur de décalage nécéssaire lorsque l'ordinateur doit sommer (ou soustraire) deux nombres flottants d'ordre de grandeur très différente. En effet, vers la fin de cette procédure (pour i grand), on additionne (ou on soustrait) un nombre inférieur à 1.0 à une somme partielle qui vraisemblablement sera de l'ordre de plusieurs millions. Certains MATPDS[][] seront donc très approximés et la précision de la somme résultante (i.e., NRJ) en sera affectée.
<4 pts>

Une façon de renvoyer une valeur NRJ plus précise (i.e., pour laquelle cette perte de précision numérique sera atténuée) serait de faire une somme partielle pour les 5000 éléments de la deuxième boucle (FOR L). De ce fait, la somme NRJ+=TMP à la fin de la deuxième boucle concernera des nombres d'ordre de grandeur moins grand et la précision de NRJ sera plus grande.
<6 pts>

```
                         ALGO

| matpds[...][...]        Tableau 2D de flottants
| nrj                     flottant

• nrj= 0.0
• for i = 0 to 500000,   i++   do
  |  ...
  |  tmp=0.0
  |  for l = 0 to 5000,   l++ do
  |    |  ...
  |    |  if   COND1   then    tmp+=matpds[l][i]
  |    |_ else    tmp-=matpds[l][i]
  |_ nrj+=tmp
• return nrj
```

Nota : Lorsque je programme sur mon ordinateur la routine non corrigée, avec des MATPDS[][]=0.1 et la COND1, remplie 9 fois sur 10 ; la vraie valeur de $nrj = 200 \times 10^6$. J'obtient 2097152 en utilisant la routine non corrigée en floattant, soit une erreur sur la valeur NRJ de 98.9% en simple précision (et de 3.65×10^{-6}% en double précision). Pour la routine corrigée, j'obtient une valeur de $nrj = 200.000016 \times 10^6$, soit une erreur sur la valeur NRJ de 8×10^{-6}% en simple précision soit une erreur identique que celle faîte en double précision.

3.

Dans les deux cas, on aura un problème d'annulation de cse due à la soustraction de deux nombres approximés quasi égaux lorsque x tendra vers zéro.
<1 pt>
Pour éviter ce problème, on doit écrire ces relations de telle façon que plus aucune soustraction de deux nombres incertains (i.e., imprécis) quasi égaux apparaissent. On utilise le conjugué de l'expression $\sqrt{x^2+1}-1$ que l'on multiplie au numérateur et au dénominateur dans la première expression et on utilisera un développement limitée de $\exp(x)$ au voisinage de 0 pour la deuxième expression.

$$(a) \qquad \frac{\cos(x)}{\sqrt{x^2+1}-1} = \frac{\cos(x)\left(\sqrt{x^2+1}+1\right)}{x^2} \qquad (x \to 0)$$

<2.5 pts>

$$(b) \qquad -1+\exp x = \frac{x^2}{2!}+\frac{x^3}{3!}+\frac{x^4}{4!}+\ldots\frac{x^n}{n!} \qquad (x \to 0)$$

<2.5 pts>

4.

La plus petite valeur flottante plus grande que 1.0 soustraite à 1.0 est dans cette numérotation flottante permet de calculer l'Epsilon machine selon cette définition

$$0.\underbrace{10000000000000000000001}_{24 \text{ bits}} \times 2^1 \; - \; 0.\underbrace{10..}_{24 \text{ bits}} \times 2^1 = 2^{-23}$$

<5 pts>

Nota : l'une des deux définitions de l'Epsilon machine vient de GNU libc manual et de la documentation de Microsoft Visual C++, l'autre provient du standart ISO C (cf. Wikipedia).

III. Méthode du Point Fixe et de Newton (25 pts)

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

$$f(x) = x - \ln(1 + x) - 0.2 \tag{4}$$

1. **Méthode du point fixe**
   - (a) Montrer qu'il existe une racine unique r pour cette Eq. (4) dans l'intervalle $J =]0, 1]$. En remarquant que l'équation $f(x) = 0$ est équivalente à $g_1(x) = x$ avec $g_1(x) = \ln(1 + x) + 0.2$, montrer que l' intervalle J est un intervalle sur lequel la convergence vers une solution unique par la méthode du point fixe est assurée.
     <**5 pts**>
   - (b) Utiliser le résultat de la question précédente pour calculer les 5 premières estimées $r_1, \ldots, r_5$, en partant de $r_0 = 0.5$.
     <**4 pts**>
   - (c) Trouver deux fonctions $g_2(x)$ et $g_3(x)$ avec lesquelles
     - $g_2(x) = x$ et $g_3(x) = x$ sont équivalentes à l'équation $f(x) = 0$.
     - Pour lesquelles vous demontrerez qu'elles ne sont pas contractantes sur J (i.e., pour laquelle la convergence vers une solution unique par la méthode du point fixe n'est pas assurée).

     <**6 pts**>
2. **Méthode de Newton**
   - (a) Soit $\Upsilon(x)$, la fonction intervenant dans la méthode itérative de Newton pour la résolution de la racine r de l'équation (4) dans J. Donner $\Upsilon(x)$ ainsi que la relation itérative $r_{n+1} = \Upsilon(r_n)$.
     <**3 pts**>
   - (b) Établissez si la méthode itérative précédente sera convergente (si on prend x_0 dans J).
     <**4 pts**>
   - (c) En déduire une valeur approchée de r après 3 itérations (i.e., donner r_1, r_2, r_3) (avec toujours $r_0 = 0.5$).
     <**3 pts**>

Réponse

1.(a)

L'étude des variations de la fonction f sur $J =]0, 1]$ montre que la fonction est continue et décroissante sur J (donc monotone) ($f'(x) = \frac{-x}{1+x} < 0$ sur J. <**1 pt**>
De plus, on a $f(0) = 0.2$ et $f(1) = \ln(2) - 0.8 < 0$ donc $f(0)f(1) < 0$ et il existe donc une racine r unique dans cet intervalle. <**1 pt**>
De plus

$$g_1'(x) = \frac{1}{1 + x} < 1 \quad \forall x \in J =]0, 1]$$

La fonction $g_1(x)$ est donc contractante sur J et la convergence est assurée.
<**3 pts**>

1.(b)

En partant de $r_0 = 0.50$, on a, $r_n = g_1(r_{n-1})$ et,

$$\begin{aligned} r_1 &= 0.6054651081 \\ r_2 &= 0.6734135016 \\ r_3 &= 0.7148655532 \\ r_4 &= 0.7393346829 \\ r_5 &= 0.753502674 \end{aligned}$$

<4 pts>

Nota : Cela semble converger très lentement vers $r = 0.7722498296$.

1.(c)

$f(x) = 0$ est équivalent à $x = g_2(x) = \exp(-0.2)\exp(x) - 1$. Sur J, $g_2(x)$ n'est pas contractante car $g_2'(x) = \exp(-0.2)\exp(x)$ et $g_2'(1) \approx 2.2255 > 1$.

<3 pts>

$f(x) = 0$ est équivalent à $x = g_3(x) = 2x - \ln(1+x) - 0.2$. Sur J, $g_3(x)$ n'est pas contractante car $g_3'(x) = 2 - \frac{1}{1+x}$ et $g'(1) = 1.5 > 1$.

<3 pts>

2.(a)

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

$$g(x) = x - \frac{f(x)}{f'(x)} = x - \frac{x - \ln(x+1) - 0.2}{1 - \frac{1}{1+x}} = x - \frac{x - \ln(x+1) - 0.2}{\frac{x}{1+x}}$$

On a donc la formule itérative suivante,

$$r_{n+1} = r_n - \frac{r_n - \ln(r_n+1) - 0.2}{\frac{r_n}{1+r_n}}$$

<3 pts>

2.(b)

Pour démontrer la convergence de cette suite itérative, le plus simple est de démontrer (dans ce cas) que x_0 et la racine que l'on cherche ne sont pas séparé par un extrema de la fonction $f(x)$ (i.e., un endroit ou $f(x)$ s'annule).

$f'(x) = \frac{-x}{1+x}$ et $f''(x) = \frac{-1}{(1+x)^2} < 0 \ \forall x \in J$ donc f' est décroissante sur J. De plus $f'(0) = 0$ et $f'(1) = -0.5$. Donc mis a part en $x = 0$ (qui est exclu de l'intervalle J), x_0 et la racine que l'on cherche ne sont pas séparé par un extrema de la fonction $f(x)$ et la méthode de Newton devrait converger.

<4 pts>

2.(c)

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

$$\begin{aligned} r_1 &= 0.6054651081 \\ r_2 &= 0.6615776673 \\ r_3 &= 0.6964865586 \\ r_4 &= 0.7195112841 \end{aligned}$$

<3 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**>

$$A = \begin{pmatrix} 1 & 4 & 5 \\ 2 & 12 & 18 \\ 3 & 15 & 22 \end{pmatrix}$$

2. Calculer le déterminant de A.
   <**3 pts**>
3. On aimerait calculer l'inverse de la matrice U. Donner l'inverse de cette matrice en faisant les opérations qui numériquement permettraientt de le faire le plus rapidement possible. Préciser dans le cas d'une matrice U de dimension n, la complexité algorithmique de cette technique.
   <**6 pts**>

Réponse

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

Par factorisation directe, on trouve,

$$A = \begin{pmatrix} 1 & 4 & 5 \\ 2 & 12 & 18 \\ 3 & 15 & 22 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 2 & 4 & 0 \\ 3 & 3 & 1 \end{pmatrix}}_{L<\mathbf{4\ pts}>} \underbrace{\begin{pmatrix} 1 & 4 & 5 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{pmatrix}}_{U<\mathbf{4\ pts}>}.$$

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

$$\begin{aligned} \det(\mathrm{A}) &= \det(\mathrm{L}) \times \det(\mathrm{U}) \\ &= 1 \times (4 \times 1) \\ &= 4 \end{aligned}$$

<**3 pts**>

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

Numériquement, la résolution par substitution arrière des trois systèmes $Ux = (1\ 0\ 0)^t$, $Ux = (0\ 1\ 0)^t$ et $Ux = (0\ 0\ 1)^t$ permettrait d'obtenir les trois colonnes de la matrice U^{-1}.

<**2 pts**>

On trouve facilement, pour la première colonne, $x = (1\ 0\ 0)$. Pour la deuxième colonne, $x = (-4\ 1\ 0)$ et enfin pour la troisième colonne, $x = (3\ \ -2\ 1)$, c'est à dire la matrice U^{-1} suivante

$$U^{-1} = \begin{pmatrix} 1 & -4 & 3 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{pmatrix}$$

<**2 pts**>

Soit une complexité de $n \times O(\frac{n^2}{2}) = O(\frac{n^3}{2})$.
<**2 pts**>

V. Interpolation (16 pts)

1. Trouver, en utilisantla formule de Lagrange, une interpolation de exp(0.5) en utilisant la connaissance des points suivants et trouver ensuite une borne supérieure de l'erreur d'interpolation associée à cette estimation.

x_k	0	1	-2
$y_k = \exp(x_k)$	1	≈ 2.72	≈ 0.135

<10 pts>

2. Trouver, en utilisantla formule de Newton-Gregory, une interpolation de exp(0.5) en utilisant la connaissance des points suivants

x_k	-1	0	1
$y_k = \exp(x_k)$	≈ 0.37	1	≈ 2.72

<7 pts>

Réponse

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

En utilisant donc le polynôme de collocation d'ordre deux qui passe par ces trois points, on a

$$P_2(x) = \frac{(x-1)(x+2)}{-2} + \frac{x(x+2)}{3} \times 2.72 + \frac{2(x-1)}{6} \times 0.135$$

<4 pts>

Une valeur interpolée pour exp(0.5) est donc estimée par

$$\exp(0.5) \approx P_2(x=0.5) \approx 0.625 + 1.13 - 0.0056 \approx 1.7527$$

<2 pt>

Une borne supérieure de l'erreur d'interpolation est donnée par

$$\begin{aligned}
\left|\exp(0.5) - P_2(0.5)\right| &< \left|\frac{f^{n+1}(\xi)}{(n+1)\,!}\,(x-x_0)\,(x-x_1)\,(x-x_2)\right| \qquad \xi \in [-2, 1] \\
&< \left|\frac{\exp(1)}{3\,!}\,(0.5-0)\,(0.5-1)\,(0.5+2)\right| \\
&<\approx 0.283
\end{aligned}$$

<4 pts>

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

En prenant donc ces trois points (équidistants et ordonnés), le tableau des différences s'écrit

x	y	Δy	$\Delta^2 y$
-1	0.37		
		0.63	
0	2		1.09
		1.71	
1	2.72		

<**3 pts**>

On obtient le polynôme suivant

$$P_3(s) = 0.37 + 0.63s + \frac{1.09\, s\, (s-1)}{2}$$

<**3 pts**>

Pour l'interpolation on trouve, puisque $s = \frac{1}{h}(x - x_0) = \frac{1}{1}(0.5 + 1) = 1.5$

$$P(x = 0.5) = P(s = 1.5) \approx 1.7275$$

<**1 pt**>