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 : *16/02/2017*

I	Amplification d'Erreur (25 pts)
II	Erreur en Arithmétique Flottante (45 pts)
III	Recherche des racines d'une équation (50 pts)
Total	120 points.

Tous documents personnels, calculatrices et calculateurs autorisés

I. Amplification d'Erreur (25 pts)

La densité ρ d'un cylindre de métal circulaire est donnée par la formule ($\rho =$ masse/volume) suivante :

$$\rho = \frac{M}{\pi\, R^2\, L}$$

Supposons que l'on trouve $M = 244$ kilogrammes avec une précision relative de 0.1% et une valeur approximée pour L de $L = 1$ mètre et une valeur approximée pour R de $R = 0.1$ mètre.

1. En utilisant la méthode de propagation d'erreur (basé sur l'approximation de Taylor de la fonction $\rho(M, R, L)$ au premier ordre), donner l'incertitude relative que l'on peut se permettre sur R et l'incertitude relative que l'on peut se permettre sur L pour que les trois erreurs relatives (*i.e.*, celle que l'on fait sur M, R et L) contribuent identiquement dans l'incertitude totale que l'on fera sur $\Delta\rho$.
Donner ensuite l'approximation de ρ et son incertitude relative en arrondissant cette estimation judicieusement et en soulignant le nombre de chiffres significatifs "exact" (cse) qu'il possède.
<15 pts>

2. Soit la suite itérative suivante :

$$x_{[n+1]} = 4\, x_{[n]} - 3(x_{[n]}^2) \qquad \text{avec :} \quad x_{[0]} = 0.5$$

que l'on peut calculer sur ordinateur soit

(a) $\quad 4\, x_{[n]} - (3\, x_{[n]}) \cdot x_{[n]}$

c'est-à-dire en demandant à l'ordinateur de calculer d'abord $(4\, x_{[n]})$ puis $(3\, x_{[n]})$ puis le produit entre $(3\, x_{[n]})$ et $x_{[n]}$ puis la différence entre $4\, x_{[n]}$ et $[(3\, x_{[n]}) \cdot x_{[n]}]$.

(b) ou : $\quad (4 - 3\, x_{[n]}) \cdot x_{[n]}$

c'est-à-dire en demandant à l'ordinateur de calculer $(4 - 3\, x_{[n]})$ puis le produit entre ce dernier terme et $x_{[n]}$

Sachant que $x_{[0]} = 0.5$ est entachée d'une erreur d'affectation de Δx, trouver pour les deux cas (a) et (b), l'amplification de l'erreur obtenue par ces deux suites itératives générée par une itération.
Pour simplifier le raisonnement et ce calcul d'amplification, on supposera que les valeurs approximées de ces suites (*i.e.*, $x_{[0]}$, $x_{[1]}$, $x_{[2]}$, etc.) sont en moyenne égale à 0.5.
<10 pts>

Réponse

$\boxed{1}$

On a donc une erreur absolue de $\Delta M = 0.244$ et pour calculer l'incertitude de ρ, on doit calculer la différentielle $\Delta\rho(M, R, L)$, *i.e.*,

$$\Delta\rho(M,R,L) = \left|\frac{1}{\pi\, R^2\, L}\right| \Delta M + \left|-\frac{M}{\pi\, L} \cdot \frac{2}{R^3}\right| \Delta R + \left|-\frac{M}{\pi\, R^2} \cdot \frac{1}{L^2}\right| \Delta L$$

Soit en remplaçant par nos valeurs approximées pour M, R et L et une contribution due seulement à ΔM (dans l'incertitude totale que l'on fera sur $\Delta\rho$) ;

$$\Delta\rho(M,R,L) = \underbrace{(100/\pi)\, \Delta M}_{\approx 7.767} \;+\; (488000/\pi)\, \Delta R \;+\; (24400/\pi)\, \Delta L$$

Du coup, on en déduit donc, pour que ces trois contributions soient équivalentes :

$$(488000/\pi)\,\Delta R = (100/\pi)\,\Delta M \qquad \Leftrightarrow \qquad \Delta R = \frac{100\cdot\Delta M}{488000} = 5\times 10^{-5}$$
$$(24400/\pi)\,\Delta L = (100/\pi)\,\Delta M \qquad \Leftrightarrow \qquad \Delta L = \frac{100\cdot\Delta M}{(24400} = 10^{-3}$$

Soit une erreur relative de (aussi) de 0.1% pour L et une erreur relative pour R de $\Delta R/R = 0.05$ de 0.05% (en fait deux fois moins grande que celle de M).

L'erreur absolue totale que l'on fera sur $\Delta\rho(M,R,L) = 3\times(100/\pi)\,\Delta M \approx 23.30028$ et comme la valeur approximée de ρ est $\rho^\star = 7766.761223$ avec une erreur absolue de $23.30028 < 0.5\times 10^2$ (*i.e.* le rang du dernier chiffre significatif est celui des centaines ou 2-ième rang), on peut écrire, en toute rigueur :

$$\hat{\rho} = \underline{77}70 \pm 24 \quad \text{ou} \quad \hat{\rho} = \underline{77}70 \pm 50$$

<15 pts>

Nota -1- : On aurait pu arriver plus vite, sans calcul, au même résultat en partant de la différentielle :

$$\begin{aligned}
\Delta\rho(M,R,L) &= \left|\frac{1}{\pi\,R^2\,L}\right|\,\Delta M + \left|-\frac{M}{\pi\,L}\cdot\frac{2}{R^3}\right|\,\Delta R + \left|-\frac{M}{\pi\,R^2}\cdot\frac{1}{L^2}\right|\,\Delta L \\
&= \underbrace{\left|\frac{M}{\pi\,R^2\,L}\right|}_{\mathcal{K}}\,\frac{\Delta M}{M} + \left|-2\frac{M}{\pi\,R^2\,L}\right|\,\frac{\Delta R}{R} + \left|-\frac{M}{\pi\,R^2\,L}\right|\,\frac{\Delta L}{L} \\
&= \mathcal{K}\cdot\mathcal{E}_r(M) + 2\,\mathcal{K}\cdot\mathcal{E}_r(R) + \mathcal{K}\cdot\mathcal{E}_r(L) \\
&= \mathcal{K}\cdot\Big(\underbrace{\mathcal{E}_r(M)}_{0.1\%} + 2\,\mathcal{E}_r(R) + \mathcal{E}_r(L)\Big)
\end{aligned}$$

et on peut conclure que pour 0.1% d'erreur relative pour M, il faut aussi 0.1% d'erreur relative pour L et $0.1\%/2 = 0.05\%$ d'erreur relative pour R pour équilibrer les erreurs de ces trois termes dans la contribution finale de l'erreur pour ρ.

Nota -2- : On aurait pu arriver encore plus vite au même résultat (!) en prenant le log de l'expression $\rho = M/\pi\,R^2\,L$ puis en différenciant les deux expressions et en se rappelant que les erreurs numériques s'ajoutent toujours ! En effet :

$$\log\rho = \log\Big(\frac{M}{\pi\,R^2\,L}\Big) = \log(M) - \log\pi - \log(R^2) - \log(L) \qquad \Leftrightarrow \qquad \frac{\Delta\rho}{\rho} = \underbrace{\frac{\Delta M}{M}}_{0.1\%} + 2\,\frac{\Delta R}{R} + \frac{\Delta L}{L}$$

et là, on s'aperçoit directement que, pour cette expression, si on veut une contribution identique entre les trois termes, on doit avoir, pour une erreur relative de 0.1% de M, une erreur relative identique pour L et une erreur relative deux fois moins grande pour R.

2

1-ère façon

• Pour le cas (a), on a d'abord, pour le calcul de $u = 4\,x_{[n]}$, une amplification de l'erreur Δx par 4 et donc $\Delta u = 4\,\Delta x$. Pour le calcul de $v = 3\,x_{[n]}$, une amplification de l'erreur Δx par 3 et donc $\Delta v = 3\,\Delta x$. Pour le calcul du produit de $v = (3\,x_{[n]})$ avec $w = x_{[n]}$ $(z = v\,w)$, on a $\Delta z = |x_{[n]}|.3\,\Delta x + |3\,x_{[n]}|.\Delta x$; soit si en moyenne on considère les $x_{[n]}$ égaux à 0.5, $\Delta z = 3\,\Delta x$.

Finalement l'imprécision d'une différence de deux nombres approximés (différence entre $4\,x_{[n]}$ d'imprécision $4\,\Delta x$ et $[(3\,x_{[n]})\cdot x_{[n]}]$ d'imprécision $3\,\Delta x$) sera la somme des imprécisions et on obtient donc :

pour l'expression **(a)** $\triangleright$ $7\,\Delta x$

• Pour le cas (b), on a d'abord, pour le calcul de $u = (4 - 3\,x_{[n]})$, une amplification de l'erreur Δx par 3 et donc $\Delta u = 3\,\Delta x$.
Pour le calcul du produit de $u = (4 - 3\,x_{[n]})$ avec $w = x_{[n]}$ $(z = u\,w)$, on a $\Delta z = |x_{[n]}|.3\,\Delta x + |4 - 3\,x_{[n]}|.\Delta x$; soit si en moyenne on considère les $x_{[n]}$ égaux à 0.5, $\Delta z = 4\,\Delta x$ et on obtient donc :
pour l'expression **(b)** $\triangleright$ $4\,\Delta x$

2-ème façon

En posant $x_n = 0.5 \pm \Delta x$
• Pour le cas (a) $\triangleright$ $x_{n+1} = 4x_n - 3x_n^2 = 4(0.5 \pm \Delta x) - 3(0.5 \pm \Delta x) = 1.25 \pm 7\,\Delta x \pm 3\,\Delta^2 x \approx 1.25 \pm \boxed{7\Delta x}$
• Pour le cas (a) $\triangleright$ $x_{n+1} = (4 - 3x_n)\,x_n = 1.25 \pm 5\,\Delta x \pm 3\,\Delta^2 x \approx 1.25 \pm \boxed{4\Delta x}$
<10 pts>

Nota : Dans les deux suites, le calcul du 100-ième termes de cette suite sera désastreux. Même dans le moins pire de ces deux cas (cas (b)), à la 100-ième itération, on aura une amplification de $4 \times 4 \times \ldots \times 4$ (100 fois) de Δx ce qui provoquera une estimation totalement chaotique et désastreuse de ce calcul. À partir du 23-ième terme on commence à voir que la deuxième suite est un peu plus intéressante, en terme de précision que la première ; en effet ; la vraie valeur de $x_{[23]} = 1.3303118878$ et le cas (a) nous donne $x_{[23]} = 1.3269840479$ et le cas (b) nous donne $x_{[23]} = 1.3316118717$ (en notation flottante simple précision sur un système 64 bits). À partir du 30-ième terme, ces deux suites nous donnent des valeurs complètement erronées.

II. Erreur/Amplification d'Erreur en Arithmétique Flottant (45 pts)

1. Écrire un petit programme en pseudo-code ou en language C qui permette de trouver et d'afficher (à l'écran de votre ordinateur) l'epsilon machine ϵ ; c'est-à-dire le plus petit nombre positif flottant (simple précision) ayant une représentation exacte sur votre machine.
<5 pts>
2. À l'intérieur d'un algorithme, on a comme résultat intermédiaire, un produit de n termes flottants $p_1\,p_2\,\ldots, p_n$ ou $0 < p_i < 1$ avec n pouvant être grand. On aimerait garder le maximum de précision de ce produit de termes qui est ensuite utilisé dans une deuxième étape de notre l'algorithme. Préciser le type de problème que l'on pourrait rencontrer si on ne fait rien ? Comment faire pour garder le maximum de précision de ce produit pour la suite de l'algorithme ?
<5 pts>
3. Soit x une variable flottante approximée dont on modélise son imprécision en disant que $x \in [a\ b]$, *i.e.*, un intervalle de confiance dans lequel la valeur exacte de x se trouve. Trouver l'approximation de x, l'erreur absolue, l'erreur relative et le rang du dernier chiffre significatif (cse) en fonction des paramètres a et b.
<5 pts>
4. On veut estimer le terme suivant pour des valeurs élevées de n :
$$\left(1 + \frac{1}{n}\right)^n$$
Expliquez le(s) type(s) de problème numérique avec lequel on sera confronté.
Calculé analytiquement la limite de cette expression quand $n \to \infty$ en utilisant $a^x = \exp(x\,\ln a)$ et un développement limité du premier ordre pour la fonction ln.
En déduire une expression permettant d'estimer plus justement cette expression pour des valeurs

élevées de n.
<5 pts>

5. Soit $x = \underline{100}$, une valeur flottante approximée dont l'imprécision est mesurée ici en terme de chiffre significatif exacte (cse). Soit y, une valeur qui a la même imprécision (en terme d'erreur absolue) que celle de x. Trouver la valeur de y la plus différente de x qui génère une différence $z = x - y$ entachée d'une imprécision, en terme d'erreur relative, de plus de 1000% (!).
<5 pts>
6. Suposons que l'on veuille approximer la fonction exponentielle au voisinage de 0 en utilisant la somme d'un certain nombre de termes de sa série de Taylor :

$$e^x = 1 + x + (x^2/2!) + (x^3/3!) + (x^4/4!) + ...$$

Dans un premier temps, rappeler pourquoi il est tout d'abord interessant de calculer cette somme en partant du plus petit terme au plus grand.
Expliquer pourquoi le calcul de cette approximation sera plus erroné lorsque x est négatif (comparativement à $x > 0$)). Indiquer par ordre d'importance tous les problèmes numériques qui resteront et interviendront dans l'estimation de cette série avec x négatif.
<5 pts>
7. Expliquer pourquoi le calcul numérique de ces trois expressions :

(a) $1 - \cos^2(x)$

(b) $\log(\frac{1}{x^5}) - \log(\frac{1}{x^6}) \qquad (x > 0)$

(c) $\frac{x^2 \sin(x)}{x - \tan(x)}$

peuvent conduire à des problèmes d'erreurs numériques (identifier aussi pour quelle valeur de x). Identifier et citer (tous) le(s) problème(s) numérique(s) associé(s) à chacune de ces expressions et proposer une expression mathématiquement équivalente (ou dans le pire des cas, lorsqu'on ne peut absolument pas, une expression approximativement équivalente) qui permettrait d'éviter ces problèmes numériques et/ou augmenter la précision des calculs de ces trois expressions.
<15 pts>

Réponse

$\boxed{\mathbf{1}}$
On part de $x = 1.0$ qui a une représentation exacte en notation flottante simple précision ($1.0 = 0.1 \times 2^1$).

On divise ensuite itérativement ce nombre par deux ce qui permet en binaire de faire un simple décalage de bit vers la droite et assure au nombre une représentation flottante exacte mais qui permet aussi de trouver le plus petit nombre possible (minorant) pour l'exposant de la base 2 de cette représentation flottante qui ne soit pas arrondi à zéro (instruction WHILE(x !=0.0)) par la machine. Finalement on affiche tous ces nombres en base 10 sur l'écran de l'ordinateur avec beaucoup de chiffres après la virgule (instruction %.50f de PRINTF) et en écrasant le résultat précédent (instruction "\r" de PRINTF).

```
FLOAT X=1.0 ;
WHILE(X !=0.0)
{ PRINTF(" \r x=%.50f ",X) ;
X/=2.0 ; }
```

Si on prend la deuxième définition possible pour l'epsilon machine, en notation flottante simple précision, qui est la plus petite représentation exacte positive qui ajouté à 1 est différent de 1, on a le programme suivant (ou seule l'intérieur du while est différent) :

```
FLOAT X=1.0 ;
WHILE[(1.0+(X/2.0))>1.0]   X/=2.0 ;
PRINTF(" \r x=%.50f ",x) ;
```

<5 pts>

Nota : Sur mon ordinateur 64 bits, je trouve le nombre $x = \mathbf{0}.$ ...44 zéros... **140130** pour la première définition et $x = \mathbf{0}.$ ...15 zéros... **22204460492503130808472633361816406** pour la deuxième définition.

$\boxed{\mathbf{2}}$

Notons que l'on peut rapidement arriver à un UNDERFLOW qui peut s'avérer désastreux pour la deuxième partie de l'algorithme. Plusieurs stratégies sont possibles :

• On peut stocker $P_m = \sum_i \log p_i$ puis présenter à la deuxième étape de l'algorithme 10^{P_m} (ou stocker $P_m = \sum_i \ln p_i$ puis présenter à la deuxième étape de l'algorithme e^{P_m}) qui pourra éventuellement se combiner avec d'autres calcul, évitant ainsi l'UNDERFLOW.

• On peut aussi multiplier par une constante c toutes les valeurs et présenter ensuite le résultat de ce produit, à la deuxième partie de l'algorithme, par $c^{-n} P_m$.

• Une autre possibilité serait de décomposer ce produit en n produits (résultat de n multiplications partielles) et de garder cela pour la deuxième partie de l'algorithme.

<5 pts>

$\boxed{\mathbf{3}}$

• La valeur approximée de x, est par définition le centre de cet intervalle ; soit : $x^\star = (a+b)/2$.
• Son erreur absolue est la demi-largeur de cet intervalle ; soit : $\Delta x = (b-a)/2$.
• L'erreur relative de x est : $\mathcal{E}_x = \Delta x / x^\star = \frac{(b-a)/2}{(a+b)/2} = \frac{b-a}{a+b}$.
• Le rang r du dernier chiffre significatif (cse) est donné par la formule :

$$\begin{aligned} \Delta x = \frac{b-a}{2} &< 0.5\ 10^r \\ \log(10^r) &> \log(b-a) \\ r &= \lceil \log(b-a) \rceil \qquad \text{où } \lceil . \rceil \text{ est la valeur arrondie à l'entier supérieur} \end{aligned}$$

<5 pts>

$\boxed{\mathbf{4}}$

Pour de fortes valeurs de n, on aura l'addition de deux nombres d'ordre de grandeur différente (en fait avant même le problème d'UNDERFLOW de l'expression $(1/n)$ tout seul créé par de plus forte valeur de n) et donc l'expression $(1 + \frac{1}{n})^n$ sera ainsi faussement évaluée à 1.

Or, analytiquement, on peut calculer quelle devrait être la limite de cette expression pour n élevé (en utilisant $a^x = \exp(x \ln a)$ et $\ln(1+\epsilon) \approx \epsilon - \epsilon^2/2$ avec $\epsilon = 1/n$ (formule de Taylor du premier ordre) :

$$\Big(1 + \frac{1}{n}\Big)^n = \exp\Big(n \ln\big(1 + \frac{1}{n}\big)\Big) \approx \exp\Big(n\big[\frac{1}{n} - \frac{1}{2n^2}\big]\Big) \approx \exp\Big(1 - \frac{1}{2n}\Big) \underbrace{\longrightarrow}_{n \to \infty} e \quad (\approx 2.718281828)$$

Pour de fortes valeurs de n, on pourra donc utiliser l'expression du dévelopement limité d'exponentiel au voisinage de $x = 1$, en utilisant la formule de Taylor du premier ordre ; $f(x+\epsilon) \approx f(x) + \epsilon f'(x)$;

$$\text{Pour n élevé ;} \quad \exp\Big(1 - \frac{1}{2n}\Big) \approx e - \frac{e}{2n}$$

<5 pts>

Nota - : Sur ma calculatrice (CASIO *fx-7400G Plus*), à partir de $n = 10^{14}$, l'expression $(1 + \frac{1}{n})^n$ me donne faussement $''1''$ (à cause du problème de l'addition de deux nombres d'ordre de grandeur différent et avant celle liée au problème d' d'UNDERFLOW de l'expression $(1/n)$ tout seul) au lieu d'une valeur très proche (mais légèrement inférieure à celle) de $e \approx 2.718281828$. Grâce au développement limité, on voit que cette estimation devrait être plus faible de e de $\approx e/2n \approx 1.35\ E^{-14}$.

$\boxed{5}$

$x = \underline{100}$ signifie que $\Delta x < 0.5\,10^0 = 0.5$. Puisque y a la même imprécision (en terme d'erreur absolue) que celle de x, on a aussi comme incertitude $\Delta y < 0.5$ et puisque la différence de deux nombres incertains est la différence des approximations mais la somme des incertitudes, on a $z = (100 \pm 0.5) - (y \pm 0.5) = (100 - y) \pm 1.0$ (avec $z^\star = (100 - y)$) et on veut :

$$\mathcal{E}_z = \Delta z/z = 1.0/(100 - y) > 10 \quad \Leftrightarrow \quad (100 - y) < 0.1 \quad \Leftrightarrow \quad y > 99.9 \quad (\text{avec } \; y < 100)$$

Donc sachant que l'erreur relative se construit à partir d'une erreur relative qui utilise une différence en valeur absolue, on peut déterminer que y doit être compris entre $99.9 < y < 100.1$, est un intervalle ou $100 - y$, avec y ayant la même imprécision que $\underline{100}$ générera une différence $100 - y$ avec plus de 1000 % d'erreur relative. Puisque l'on cherche la valeur de y la plus différente de $x = \underline{100}$, on a nécesairement $y = \underline{99}.9$ ou $y = \underline{100}.1$.

<5 pts>

Nota -1- On peut vérifier $z = (100 \pm 0.5) - (99.9 \pm 0.5) = 0.1 \pm 1.0$, *i.e.*, $z = z^\star \pm \Delta z = 0.1 \pm 1.0$ et la différence approximée $z^\star$ est bien obtenue avec une erreur relative $\mathcal{E}_z = \Delta z/z^\star = 1.0/0.1 = 10$, *i.e.*, de l'ordre de $\mathcal{E}_z = 1000\%$ d'erreur relative (!)

Nota -2- On peut arriver au même résultat en disant que l'erreur relative d'une différence est la somme des erreur relative et donc que :
$\mathcal{E}_z = \Delta z/z = (\Delta x + \Delta y)/|x - y| = (0.5 + 0.5)/|100 - y| = 1/|100 - y| = 1000\%$ pour $y = 100.1$ ou $y = 99.9$

$\boxed{6}$

La sommation du plus petit terme au plus grand permet de limiter les erreurs numériques, et spécialement le problème de l'addition de deux nombres d'ordre de grandeur différents.

Cependant, même dans le cas d'un calcul de cette série à l'envers ; lorsque x est positif ou négatif, on aura possiblement par ordre croissant de problème numérique :

• Une erreur d'approximation due au fait que certains termes ou la somme partielle obtenue n'ont pas de représentation exacte en flottant.

• Un problème d'UNDERFLOW pour des termes de cette somme trop petit.

• Et seulement pour les nombres $x < 0$, la soustraction de deux nombres approximés quasi-égaux (ou voisins). En effet, entre le 24 et 25 ième termes de cette série (par exemple) en partant de la gauche, on aura pour une valeur de x négative une l'opération entre deux valeurs de signes différents, approximées et quasi égaux et qui est l'erreur numérique la plus dangereuse. Par contre, on pourrait arrangé cela facielement en sommant (en sens inverse) tous les termes négatives d'une part et tous les termes positives d'autres part puis sommer ces deux sommes partielles.

Pour des valeurs de x très négatives, ajoutons aussi que la valeur de $\exp(-x)$ deviendra très proche de zéro et que, considérant les erreurs dues à ce que nous avons dit précédemment, l'erreur relative de l'estimation de $\exp(-x)$ deviendra inexorablement de plus en plus grande.

<5 pts>

$\boxed{7}$

[-a-] Pour la première expression, on aura un problème d'annulation de cse (chiffres significatifs exacts) due à la soustraction de deux nombres approximés quasi égaux (en fait, un seul des deux nombres est approximé ici). Cette annulation de cse se passe lorsque $x \to 0$. Dans le cas où $x \to 0$, il vaudrait mieux donc réécrire cette expression en utilisant une propriété trigonométrique :

$$1 - \cos^2(x) = \sin^2(x)$$

<**5 pts**>

Nota : Sur ma calculatrice, (CASIO *fx-7400G Plus*), à partir de $x = 10^{-7}$, la première expression donne faussement 0, et la deuxième justement 10^{-14}. On sait que la deuxième approximation est la meilleure car le développement limité de ces deux expressions, au voisinage de $x = 0$, est : $1 - \cos^2(x) = \sin^2(x) = x^2 - x^4/4 + o(x^6)$.

[-b-] Pour la deuxième expression, on aura soit un problème d'annulation de cse qui se passe lorsque $x \to 1$ ou soit un problème d'OVERFLOW pour $x \to \infty$ pour le calcul de x^6 ou un problème d'UNDERFLOW pour le calcul de $1/x^6$.
On peut ré-écrire cette expression de telle façon que plus aucune soustraction de deux nombres incertains (i.e., imprécis) quasi-égaux apparaissent, ou que le problème d'OVERFLOW s'atténue en écrivant cette expression de la façon suivante :

$$\ln\big(\frac{1}{x^5}\big) - \ln\big(\frac{1}{x^6}\big) = \ln\Big(\frac{1/x^5}{1/x^6}\Big) = \ln(x)$$

<**5 pts**>

Nota : Sur ma calculatrice (CASIO *fx-7400G Plus*), à partir de $x = 10^{17}$, l'expression de gauche nous donne MA ERROR et l'expression de droite est égale à 39.14394658.

[-c-] Pour la troisième expression, il y a deux problèmes ; un pour $x \to 0^+$ et un pour $x \to \pi/2$

▷ pour $x \to \pi/2$ (modulo π), on a un problème d'OVERFLOW du au fait que $\tan(\pi/2) \to \infty$ au dénominateur (ou problème d'UNDERFLOW si on considère l'évaluation de $1/\tan(x)$ quand $x \to \pi/2$). Cependant cette expression peut se réécrire de la façon suivante :

$$\frac{x^2 \sin(x)}{x - \tan(x)} = \frac{x^2 \sin(x)}{x - \tan(x)} \cdot \frac{\cos x}{\cos x} = \frac{x^2\,\sin(x)\,\cos(x)}{x\,\cos(x) - \sin(x)}$$

▷ pour $x \to 0^+$, il y a deux problèmes d'importance égale. Le premier est aussi un problème d'annulation de cse au dénominateur. Le deuxième est une expression dangereuse indéterminée du type 0/0 pouvant potentiellement engendrer en notation flottante un INF ou NaN. Le plus simple, pour cette expression est de considérer, pour de très faibles valeurs, proche de 0^+, son développement limité.

$$\frac{x^2 \sin(x)}{x - \tan(x)} \approx \frac{x^2(x)}{x - (x + x^3/3)} = -3 + \frac{x^2}{2} + o(x^4)$$

<**5 pts**> (pour l'un des deux ▷)

Nota -1- : pour $x \to 0^+$, sur ma calculatrice (CASIO *fx-7400G Plus*), à partir de $x = 10^{-7}$, l'expression de droite nous donne MA ERROR alors que pour de très faibles valeurs de x, l'expression devrait tendre vers une valeur très proche de -3 mais légèrement supérieur à -3 (cf. son dev. limité). Pour $x = 10^{-3}$, l'expression de droite nous donne assez justement -2.99999836, puis pour $x = 10^{-4}$, cela se dégrade et l'expression nous donne -3.000002995, avec $x = 10^{-5}$, cela se dégrade encore plus avec -3.00030003, puis pour $x = 10^{-6}$, cela se dégrade encore plus avec -3.0303030, puis pour $x = 10^{-7}$, on obtient MA ERROR.

Nota -2- : pour $x \to \pi/2$, sur ma calculatrice (CASIO *fx-7400G Plus*), pour $x = \pi/2$ pour la première expression, on trouve MA ERROR alors que pour la deuxième (mathématiquement équivalente), on trouve correctement la valeur 0.

III. Recherche des racine d'une équation (50 pts)

On se propose de trouver numériquement dans $R^\star$ une valeur approchée de la racine de la fonction

$$f(x) = \frac{e^x}{e^2 \cdot x} - 1 = 0 \qquad x \neq 0 \tag{1}$$

1. Méthode de la Bissection
   (a) Trouver un intervalle $[a\ b]$ de longueur 1 avec a, b deux entiers $\in [1, \ldots, 5]$ dans lequel vous aurez démontré qu'il existe une **unique** racine de f.
   <**5 pts**>
   (b) Donner le nombre d'itérations qui serait nécessaire à la méthode de la bissection pour arriver avec cet intervalle à une estimation de la racine avec 7 cse après la virgule (sans estimer explicitement cette racine par la méthode de la bissection).
   <**5 pts**>
2. Méthode de Newton
   (a) Donner la relation itérative $r_{n+1} = g_{\text{newt.}}(r_n)$ intervenant dans la méthode itérative de Newton pour la résolution itérative de la racine r de l'équation $f(x) = 0$.
   <**5 pts**>
   (b) Montrer qu'avec $r_0 = 3$, la relation itérative de Newton converge pour une racine de f supérieure à 1.
   <**5 pts**>
   (c) Calculer les 4 premières estimées $r_1, \ldots, r_4$, en partant de $r_0 = 3$.
   <**5 pts**>
3. Méthode du point fixe
   (a) Donner une forme $x = g_1(x)$ mathématiquement équivalente pour $f(x) = 0$ et une valeur initiale $r^{[0]}$ prise dans un intervalle de longueur contenant la racine supérieure à 1 de f pour laquelle la suite itérative $r^{[n+1]} = g_1(r^{[n]})$ ne converge pas et démontrer pourquoi.
   <**5 pts**>
   (b) Donner une forme $x = g_2(x)$ mathématiquement équivalente pour $f(x) = 0$ et une valeur initiale $r^{[0]}$ prise dans un intervalle de longueur contenant la racine supérieure à 1 de f pour laquelle la suite itérative $r^{[n+1]} = g_2(r^{[n]})$ converge et donner les 4 premières estimées $r_1, \ldots, r_4$, obtenue en partant de r_0.
   <**10 pts**>
4. Misc.
   (a) Si on a raison de croire que la racine que l'on recherche est double, quelle est la méthode qui sera la plus rapide, en terme de coût calculatoire, entre la méthode de la bissection et la méthode de Newton ? Indiquer la méthode qui sera la plus rapide entre ces deux méthodes et essayer d'estimer de combien ce coût calculatoire sera plus rapide (par rapport à l'autre méthode). Justifier bien votre réponse.
   <**5 pts**>
   (b) Supposons que le cahier des charges nous demande de trouver une estimation de cette racine selon le critère de la tolérance sur x uniquement ou selon le critère de la tolérance sur x **et** le critère de la tolérance sur la fonction. Discuter quel serait le ou les désavantages de la méthode de l'interpolation linéaire sur les 3 autres méthodes que l'on a vues en cours.
   <**5 pts**>

Réponse

1.(a)

Il faut donc trouver un intervalle pour lequel on a donc $f(a) \cdot f(b)$ de signe opposé et démontrer ensuite que la fonction f est continue et **surtout monotone** pour démontrer qu'il existe une racine **unique** de f contenue dans cet intervalle.

On trouve $f(1) \approx -0.6321$, $f(2) = -0.5$, $f(3) \approx -0.0939$, $f(4) \approx 0.847$. Donc un intervalle de longueur 1 ou $f(a) \cdot f(b)$ est de signe opposé est l'intervalle $[3,\ \ 4]$.

Maintenant, il reste à démontrer qu'il existe une racine unique dans cet intervalle. $f(x)$ est une fonction continue donc au moins une racine est assurée d'y être. De plus :

$$f'(x) = \frac{e^x\, e^2\, x - e^2\, e^x}{e^4\, x^2} = \frac{e^x\,(x-1)}{e^2\, x^2} > 0 \quad \forall x > 1 \quad \text{donc pour :} \quad x \in [3 \;\; 4]$$

<5 pts>

1.(b)

L'intervalle de longueur 1 se divise de moitié par la méthode de la bissection donc il faut que celui-ci en largeur soit :

$$\frac{1}{2^n} < 0.5\, 10^{-7} \Leftrightarrow n > \frac{\log 2 + 7 \log 10}{\log 2} = 24.25 \quad \text{soit} \;\; 25 \;\; \text{itérations}$$

<5 pts>

2.(a)

La fonction $f(x)$ est dérivable sur $\mathbb{R}^\star$ et la méthode itérative de Newton permet d'écrire (cf. question 1.(a)) :

$$r_{n+1} = r_n - \frac{f(r_n)}{f'(r_n)} = r_n - \frac{r_n\,(e^{r_n} - e^2\, r_n)}{e^{r_n}\,(r_n - 1)}$$

<5 pts>

2.(b)

Dans ce cas, le plus simple est de montrer que la fonction $f(x)$, sur l'intervalle $J = [3, 4]$ (cf. question 1.(a)), intervalle dans lequel on prendra $r_0 = 3$ et dans lequel une racine unique s'y trouve, ne présente pas d'*extrema*, i.e., de valeurs pour laquelle, $f'(x)$ s'annule. Dans notre cas, la dérivée a été déjà calculé à la question 1.(a) ; $f'(x) = \frac{e^x\,(x-1)}{e^2\,x^2}$ qui ne s'annule jamais sur $[3 \;\; 4]$ (seulement pour $x = 1$ qui est en dehors de cet intervalle) donc aucun problème, la méthode de Newton va converger.
<5 pts>

Nota : Il aurait été beaucoup beaucoup plus difficile de démontrer que cette suite converge à partir de la condition de convergence du point fixe, *i.e.*, en démontrant que $|g'(\xi)| < 1 \quad \forall \xi \in J$ et $r_{n+1} = g(r_n)$.

2.(c)

En partant de $r_0 = 3$, on a, $r_n = g_{\text{newt.}}(r_{n-1})$ et :

$$\begin{aligned} r_1 &= 3.155457485 \\ r_2 &= 3.146228754 \\ r_3 &= 3.146193221 \\ r_4 &= 3.146193221 \end{aligned}$$

<5 pts>

Nota : Sur ma calculatrice, l'estimation se stabilise très rapidement, à partir de $r_4 = 3.146193221$!! Cela nous indique aussi que la fonction dont on cherche la racine est très très linéaire au voisinage de la racine. On peut voir l'effet d'une racine simple sur la convergence de Newton qui, dans ce cas, est quadratique.

3.(a)

$$\frac{e^x}{e^2 \cdot x} - 1 = 0 \quad \Leftrightarrow \quad x = e^{-2}\, e^x = g_1(x)$$

avec $|g_1'(x)| = |e^{-2}\, e^x| > 1$ dans l'intervalle $[3 \;\; 4]$ (qui est le seul intervalle de longueur 1 contenant la racine supérieure à l'unité de f ; cf. question 1.(a)). Donc avec cette suite itérative, on n'est pas assuré de converger vers la racine unique de f qui se trouve dans l'intervalle $[3 \;\; 4]$.

<5 pts>

Nota -1- : Avec cette suite itérative : $r_{[n+1]} = e^{-2}\, e^{r_{[n]}}$ en partant de $r_{[0]} = 3$ on converge néanmoins vers la racine de f qui se trouve dans l'intervalle $]0 \;\; 1]$ c'est-à-dire $r = 0.1585943396$. Ce qui montre que cette condition de convergence du point fixe est une condition suffisante de convergence mais pas (absolument) nécessaire comme on peut le constater dans cet exemple car $|g_1'(x)| = |e^{-2}\,e^x| < 1$ dans $]0 \;\; 1]$ mais $r_{[0]} = 3$ est à l'exterieur de cet intervalle.

Nota -2- : On peut trouver aussi qui ne converge pas :

$$\frac{e^x}{e^2 \cdot x} - 1 = 0 \quad \Leftrightarrow \quad \frac{e^x}{e^2 \cdot x} - 1 + x = x = g_1(x) \quad \text{parce que :} \quad \left|g_1'(x)\right| = \left|\frac{e^x\,(x-1)}{(.)^2} + 1\right| > 1 \quad \forall x > 1$$

$\boxed{\textbf{3.(b)}}$ $$\frac{e^x}{e^2 \cdot x} - 1 = 0 \quad \Leftrightarrow \quad \ln(e^x) - \ln(e^2 \cdot x) = \ln(1) \quad \Leftrightarrow \quad x = 2 + \ln(x) = g_2(x)$$

avec $|g_2'(x)| = |1/x| < 1$ dans l'intervalle $[3 \;\; 4]$ (qui est le seul intervalle de longueur 1 contenant la racine supérieure à l'unité de $f(x) = 0$, cf. question 1.(a)). Donc avec cette suite itérative, on est assuré de converger vers la racine unique de f qui se trouve dans l'intervalle $[3 \;\; 4]$.

En partant de $r_0 = 3$, on a, $r_n = g_2(r_{n-1})$ et :

$$\begin{aligned} r_1 &= 3.098612289 \\ r_2 &= 3.130954362 \\ r_3 &= 3.141337866 \\ r_4 &= 3.144648781 \end{aligned}$$

<10 pts>

Nota : Avec cette suite itérative on converge moins vite que la suite de Newton, ce qui est normale puisque celle-ci est de convergence linéaire. La solution à convergence se stabilise à $r_{18} = 3.14619322$ au bout de 18 itérations ce qui est légèrement mieux dans ce cas à ce que nous avions prévu pour la convergence de la méthode de la bissection (cf. question 1). En fait la convergence est ici légérement plus rapide, et on peut le montrer théoriquement car la constante assymptotique de cette méthode pour cette fonction et racine est $C = |g'(r = 3.144648781)| \approx 0.32$ ce qui n'est, en fait, pas beaucoup plus faible que celle associée à la méthode de la bissection ($C = 0.5$).

$\boxed{\textbf{4.(a)}}$

Dans le cas d'une racine double, la méthode de Newton a un taux de convergence linéaire et une constante asymptotique de $C = 0.5$ (cf. cours). C'est-à-dire que la méthode n'est pas plus rapide, en terme de nombre d'itérations que la méthode de bissection. Mais la méthode de bissection demande une évaluation de fonction par itération alors que la méthode de Newton demande deux évaluations de fonction par itération. Donc, dans le cas d'une racine double, la méthode de Newton arriverait à la solution avec deux fois plus de calcul (ou deux fois plus de temps) que la méthode de la bissection.

<5 pts>

$\boxed{\textbf{4.(b)}}$

On a vu en cours, qu'il existait des cas ou l'intervalle qui est à chaque fois redéfini par la méthode de l'interpolation et qui est censé encadrer la racine pouvait être coincé de telle façon que l'une des deux bornes de cet intervalle soit la racine et l'autre reste coincé vers la borne supérieure ou inférieur de cet interval ; de telle façon qu'il ne diminue plus et que le critère de la tolérance sur x ne puisse être donc utilisé contrairement à la méthode de la bissection où ce cas de figure n'existe jamais.

Pour la méthode de Newton, l'intervalle $[x_{[n-1]} \;\; x_{[n]}]$ n'encadre pas la racine (pseudo-tolérance sur x) et ne peut donc être utilisé comme un vrai intervalle qui encadre la racine. Pour la méthode du point fixe, cet intervalle $[x_{[n-1]} \;\; x_{[n]}]$ est une tolérance sur x uniquement dans le cas où les itérés de la suite itérative progrèssent autour de la solution en colimaçon ; ce que l'on peut facilement détecter si $(x_{[n]} - x_{[n-1]})$, change continuellement de signe lorsque n s'incrémente.

<5 pts>