Université Paris I, Panthéon - Sorbonne

Licence M.I.A.S.H.S. deuxième année 2015 – 2016

Méthodes Numériques S4

Contrôle continu n°2, avril 2016

Examen de 1h30. Tout document ou calculatrice est interdit.

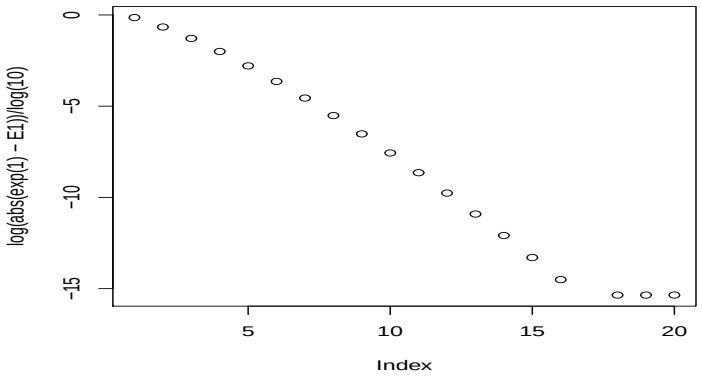
1. **(Sur 8 points)** Soit le programme:

```
n=20; E1=0
factorial(4); factorial(20)
for (k in c(1:n))
E1[k]=1+sum(1/factorial(c(1:k)))
plot(log(abs(exp(1)-E1))/log(10))
```

   (a) Les premiers résultats obtenus sont `24 2.432902e+18`. Pourquoi a-t-on obtenu `24`? **(0.5pts)**

   (b) Donner la formule mathématique de `E1[k]` **(0.5pts)**. Quelle est sa limite lorsque $k \to \infty$ et pourquoi? **(1pt)**

   (c) Le graphe obtenu est le suivant:



   Que représente `abs(exp(1)-E1)`? **(0.5pts)** Si $x = 10^{-y}$, que vaut y en fonction de x **(0.5pts)**. En déduire ce que représente `log(abs(exp(1)-E1))/log(10)` **(0.5pts)**.

   (d) Montrer que pour $k \in \mathbf{N}$, $\frac{1}{(k+1)!} <$`abs(exp(1)-E1[k])`$< \frac{e}{(k+1)!}$ **(2pts)**. En déduire que pour k grand, `log(abs(exp(1)-E1[k]))/log(10)`$\sim -\ln((k+1)!)/\ln(10)$ **(1pt)**.

   (e) On sait que pour n grand, alors $\ln(n!) \sim n\ln(n)$. En déduire l'allure du graphe **(0.5pts)**. Pourquoi pour $k \geq 18$, le graphe devient-il horizontal? **(0.5pts)** Que peut-on en conclure quant à `E1[18]`? **(0.5pts)**

Proof. (a) On a obtenu 4! qui vaut bien $24 = 4 * 3 * 2 * 1$.

(b) $E1[k] = \sum_{i=0}^{k} \frac{1}{i!}$ avec la convention $0! = 1$. On sait que $e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!}$ pour tout $x \in \mathbf{R}$, donc en prenant le cas particulier $x = 1$, on obtient que $E1[k] \to e$ quand $k \to \infty$.

(c) `abs(exp(1)-E1)` représente l'écart en valeur absolue entre `E1` et sa limite e.
Si $x = 10^{-y}$ alors $x = e^{-y\ln(10)}$, d'où $-y\ln(10) = \ln(x)$ soit $y = -\ln(x)/\ln(10)$.
On en déduit que `log(abs(exp(1)-E1))/log(10)` représente le nombre de chiffres représentatifs de l'approximation de e par `E1`.

(d) Il est clair que `abs(exp(1)-E1[k])`$= \sum_{i=k+1}^{\infty} \frac{1}{i!} = \frac{1}{(k+1)!}\left(1+\frac{1}{k+2}+\frac{1}{(k+2)(k+3)}+\cdots\right)$. Mais $1+\frac{1}{k+2}+\frac{1}{(k+2)(k+3)}+\cdots > 1$, d'où `abs(exp(1)-E1[k])`$> \frac{1}{(k+1)!}$. De même pour $k \geq 1$, $\frac{1}{(k+2)(k+3)\times(k+p)} < \frac{1}{p}$, d'où $1+\frac{1}{k+2}+\frac{1}{(k+2)(k+3)}+\cdots \leq \sum_{i=1}^{\infty}\frac{1}{i!} < e$, d'où `abs(exp(1)-E1[k])`$< \frac{e}{(k+1)!}$. Comme la fonction ln est croissante, de la double inégalité précédente on obtient:

$$-\ln((k+1)!) < \ln(\texttt{abs(exp(1)} - \texttt{E1[k])}) < 1 - \ln((k+1)!)$$

Et comme $\ln((k+1)!) \to \infty$ quand $k \to \infty$, on a bien l'équivalent proposé.

(e) On déduit des 2 équivalents que `log(abs(exp(1)-E1))/log(10)`$\sim -(k+1)\ln(k+1)$: on voit bien sur le graphe que la décroissance de la courbe est un peu plus rapide qu'une décroissance linéaire.
Pour $k \geq 18$, on vérifie sur le graphe que `log(abs(exp(1)-E1))/log(10)`< -15: on a une approximation avec 15 chiffres décimaux significatifs et on atteint ainsi la limite de précision du logiciel.
`E1[18]` est donc une approximation de e à 10^{-15} près.

□

2. **(Sur 10 points)** On désire désormais avoir une bonne et rapide approximation de ln(2).

(a) On a tapé les commandes suivantes:

```
e=exp(1); u=0.5
for (k in c(1:20))
u[k+1]=u[k]+ (2^(1/u[k])-e)/(log(2)*2^(1/u[k])/u[k]^2)
```

Ecrire mathématiquement la formule de `u[k+1]` en fonction de `u[k]` **(0.5pts)**. Quelle méthode a-t-on utilisée et pourquoi? **(1pt)** En affichant les premières valeurs de `u`, on obtient `[1] 0.5000000 0.6155705 0.6803004 0.6927908 0.6931469 0.6931472 0.6931472`. Qu'en conclure? **(0.5pts)**

(b) Dans la première commande du programme, on remplace `u=0.5`, par `u=2`, et on obtient `2.000000e+00 -3.321326e+00 -4.070670e+01 -4.260025e+03 -4.500347e+07 -5.020662e+15`. Que s'est-il passé? **(1pt)**

(c) Dites pourquoi la méthode proposée pour approcher ln(2) est en fait un contre-sens **(1pt)**.

(d) On remplace le programme précédent par celui-ci:

```
v=0.5
for (k in c(1:20))
v[k+1]=v[k]*(2-e*2^(-1/v[k]))
```

Déterminer la fonction g telle que `v[k+1]=g(v[k])` **(0.5pts)**. Montrer que $g(\ln(2)) = 0$, $g'(\ln(2)) = 0$ et en déduire un développement limité de Taylor-Lagrange d'ordre 2 de `g(v[k])` en ln(2) **(2pts)**. En déduire que $|v[k+1] - \ln(2)| \leq \frac{1}{2} M_2 |v[k] - \ln(2)|^2$, où l'on précisera la définition de M_2 **(2pts)**. Cela peut-il induire la convergence de $(v[k])_k$? **(1pts)**

(e) On a obtenu pour premières valeurs de `v`, `0.5000000 0.6602148 0.6923393 0.6931467 0.6931472 0.6931472`. Qu'en conclure sur cette suite par rapport à $(u[k])_k$? **(0.5pts)**

Proof. (a) On a $u[k+1] = u[k] + \frac{2^{1/u[k])} - e}{\frac{\ln(2)}{u[k]^2}\, 2^{1/u[k]}}$.
On a utilisé la méthode de Newton-Raphson pour approcher la racine d'une équation qui est $f(x) = 2^{1/x} - e$, et $f(x) = 0 \iff \frac{\ln(2)}{x} = 1$ d'où $x = \ln(2)$. On note que pour $x \neq 0$, $f'(x) = -\frac{\ln(2)}{x^2}\, 2^{1/x}$ et on a bien $u[k+1] = u[k] - \frac{f(u[k])}{f'(u[k])}$.
On s'aperçoit que ln(2) est déterminée avec 6 décimales significatives dès le calcul de $u[6]$.

(b) Lorsque l'on a changé la condition initiale de l'algorithme, on est devenu trop éloigné de ln(2) la limite. Ainsi on a désormais $\frac{M_2}{2m_1}|2 - \ln(2)| > 1$, où $M_2 = \sup_x |f''(x)|$ et $m_1 = \inf_x |f'(x)|$, ce qui ne garantit plus la convergence.

(c) La méthode est un contre-sens pour approcher ln(2) car dans l'algorithme et la définition de $u[k]$ ... on utilise la valeur de ln(2) déjà en mémoire dans le logiciel!

(d) La fonction g est telle que $g(x) = x\,(2 - e\,2^{-1/x})$.
On a pour $x = \ln(2)$, $2^{-1/x} = e^{-\ln(2)/x} = e^{-1}$, d'où $g(\ln(2)) = \ln(2)\,(2-1) = \ln(2)$. On a $g'(x) = 2 - e * 2^{-1/x} - \frac{\ln(2)}{x}\,e\,2^{-1/x} = 2 - 1 - 1 = 0$.
Par conséquent, un DL d'ordre 2 de Taylor-Lagrange de $g(v[k])$ en $\ln(2)$ est $g([v[k]) = \ln(2) + \frac{g''(\theta)}{2}\big(v[k] - \ln(2)\big)^2$ avec $\theta \in [v[k]\,,\,\ln(2)]$. On en déduit que $|v[k+1] - \ln(2)| = |g([v[k]) - \ln(2)| = \left|\frac{g''(\theta)}{2}\right| \big(v[k] - \ln(2)\big)^2 \leq \frac{1}{2}\,M_2\,|v[k] - \ln(2)|^2$ avec $M_2 = \sup_x |g''(x)|$.
On déduit de l'inégalité précédente, $|v[k] - \ln(2)| \leq \frac{2}{M_2}\Big(\frac{M_2}{2}\,(v[1] - \ln(2))\Big)^{2^{k-1}}$. Donc si $\frac{M_2}{2}\,|v[1] - \ln(2)| < 1$, alors $(v[k])$ converge vers $\ln(2)$ à une vitesse quadratique, sinon ce n'est pas obligatoire (notamment si $v[1]$ est trop éloigné de $\ln(2)$).

(e) On remarque que l'on est dans le cadre d'une convergence, et $v[5]$ donne déjà $\ln(2)$ avec 6 chiffres significatifs.

□

3. **(Sur 5 points)** On utilise une autre méthode pour obtenir une approximation de $\ln(2)$.

(a) Déterminer l'unique b tel que $\int_1^b \frac{dx}{x} = \ln(2)$ **(0.5pts)**.

(b) Ecrire en `R` un programme permettant l'approximation numérique `I1` de l'intégrale ci-dessus par la méthode des trapèzes pour un découpage en $n = 1000$ trapèzes **(2pts)**.

(c) Avec un tel programme on a obtenu `> I1 [1] 0.6931472`. En revenant à la formule théorique donnant une majoration de l'erreur commise par une telle approximation, pensez-vous avoir obtenu une approximation à 10^{-15} près? **(2pts)**

(d) Numériquement de toutes les méthodes proposées, laquelle retenir pour approcher au mieux $\ln(2)$? **(0.5pts)**

Proof. (a) Il est clair que $b = 2$ car une primitive de $1/x$ est $\ln(x)$.

(b) On pourrait par exemple utiliser le programme:

```
n=1000
i=1+(0:(n-1))/n
I=0.5*sum(1/i+1/(i+1/n))/n
I
```

(c) On sait d'après le cours que $|I - \ln(2)| \leq \frac{1}{12}\,\frac{M_2}{n^2}$ avec $M_2 = \sup_{x\in[1,2]} |h''(x)|$ et $h(x) = 1/x$, d'où $M_2 = \sup_{x\in[1,2]} |2/x^3| = 2$. Ainsi $|I - \ln(2)| \leq \frac{1}{6\,n^2}$. Pour obtenir une approximation à 10^{-15}, il faudrait plutôt $n \sim \sqrt{5/3}10^7$, et non $n = 1000$.

(d) On préfèrera la méthode de calcul avec $v[k]$.

□