# Rapport de TP de Mathématiques

*Frédéric Torcheux & Florian Chaulet*

Cette première séance de TP était consacrée à l'étude des séries entières. Nous devions, à l'aide du logiciel Scilab proposé un algorithme permettant de calculer avec une précision plus ou moins grande les approximations des fonctions suivantes :

1. $e^{1.37}$
2. sin(0.21466) cos(0.21466)
3. ln(1.338)
4. $\sqrt[3]{1.34}$
5. $(1.2)^{1.3}$

Afin de réaliser ces calculs, nous avons pris le développement en séries entières des fonctions remarquables comme nous le détaillerons par la suite, nous avons fixé un epsilon afin de déterminer le niveau d'approximation du calcul et ensuite dans notre programme général nous avons effectué les tests avec les valeurs appropriées. Nous les comparerons ensuite aux valeurs théoriques afin de déterminer l'exactitude de notre travail.

Voici donc le détail de nos calculs pour chacun des cas cités ci-dessus.

1. Pour l'exponentielle, le développement en série entière se fait de la manière suivante :
   $e^x = \sum_{n=0}^{+\infty} \frac{x^n}{n!}$ Nous avons donc tout d'abord créé une fonction factorielle que voici :

```
function f = factoriel(n)
 f = 1;
 cpt = 1;
 while cpt <> n + 1
  f = f * cpt;
  cpt = cpt + 1;
 end
endfunction
```

```
function a = elementExpo(x,n)
        a = x^n/factoriel(n);
endfunction
```

résultat = 3.9352965 pour epsilon = 0,0001

résultat = 3,9349887 pour epsilon = 0,001

résultat = 3,8837902 pour epsilon = 0,1

valeur théorique z = 3,9353507

Ci- dessus sont présentés également la fonction calculant la somme et ses résultats pour les valeurs proposés

2. Pour la fonction sinus, le développement en série entière de la fonction s'écrit : $\sin x = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n+1}}{(2n+1)!}$ Nous avons donc utilisé la fonction factorielle ainsi que la fonction puissance prédéfinie par le logiciel Scilab. Voici l'algorithme

$$function\ a = elementSin(x, n)$$
$$a = ((-1)\hat{}n) * (x\hat{}(2 * n + 1))/factoriel(2 * n + 1);$$
$$endfunction$$

Avec les valeurs appropriés et les différents epsilons, nous avons obtenue les résultats suivants :

pour epsilon = 0,1 z = 0;

pour epsilon = 0,001 z= 0.0230395;

pour epsilon = 0,0001 z= 0.0230395

valeur théorique z = 0,2130152

Pour la fonction cosinus il s'agit d'une formule analogue à celle du sinus comme ceci :

$\cos x = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n}}{2n!}$ ce qui nous donne l'algorithme

```
function a=elementCos(x,n)
a = (-1)^n*x^(2*n)/factoriel(2*n);
endfunction
```

Appliqué aux valeurs demandées nous obtenons :

pour epsilon = 0,1 z = 1;

pour epsilon = 0,001 z= 0.9769605

pour epsilon = 0,0001 z= 0.9769605

valeur théorique z = 0,99999

3. Le logarithme népérien contient une subtilité car sa formule mathématique s'écrit pour la variable x+1, il faudra donc adapter notre code à ce point. Nous choisirons la simplicité et pour se faire nous poserons y = x-1 d'où pour la valeur 1.338, y = 0,338.

   $\ln(x+1) = \sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{n} x^n$ Il est à noter également que la somme commence à 1 ce qui nous donne l'algorithme suivant :

```
function a=elementLn(x,n)
  y=x-1;
  a = (-1)^n*y^(n+1)/(n+1);
endfunction
```

   pour epsilon = 0,1 z = 0.338;

   pour epsilon = 0,001 z= 0.2904866

   pour epsilon = 0,0001 z= 0.2911203

   valeur théorique z = 0,2911759

4. $$(1+x)^{\alpha} = \sum_{n=0}^{\infty} \frac{\alpha(\alpha-1)\dots(\alpha-n+1)}{n!} x^n$$

Une nouvelle fois, il va falloir appliquer la subtilité de poser y = x-1 afin de parvenir à utiliser la formule du développement limité de (1+x)^alpha. Ce développement va nous permettre de calculer les approximations pour la racine cubique et la puissance.

```
function a=elementRacineCubique(x,n)
   y=x-1;
   a=produitRacine(3,n)*y^n/factoriel(n);
endfunction
```

```
function a=elementPuissance(x,n,pui)
   y=x-1;
   a=produitPuissance(pui,n)*y^n/factoriel(n);
endfunction
```

Mais ici il va falloir rajouter une fonction. En effet le développement limité est : Somme de n = 1 à l'infini de ((alpha(alpha -1) ….(alpha – n +1))/n!)*x^n
Il va donc nous falloir calculer le produit des (alpha(alpha -1) ….(alpha – n +1)). D'où la fonction

```
function m=produitRacine(a,n)
   m=1;
   cpt = 0;
   while cpt<>n
   m = m*(1/a-cpt);
   cpt = cpt+1;
 end
endfunction
```

```
function m=produitPuissance(a,n)
   m=1;
   cpt = 0;
   while cpt<>n
   m = m*(a-cpt);
   cpt = cpt+1;
 end
endfunction
```

Appliqué aux valeurs demandées nous obtenons pour la racine cubique:

pour epsilon = 0,1 z = 1.1133333

pour epsilon = 0,001 z= 1.1133333

pour epsilon = 0,0001 z= 1.1133333

Appliqué aux valeurs demandées nous obtenons pour la puissance:

pour epsilon = 0,1 z = 1,28

pour epsilon = 0,001 z= 1.2678

pour epsilon = 0,0001 z= 1.2678

valeur théorique z = 1.2678

La concordance entre les résultats théoriques et expérimentaux est donc bien établie à une certaine marge d'erreur près. Nous pouvons donc conclure que l'équivalence de nos fonctions sous formes algorithmique avec les fonctions théoriques est justifiée.