INFORMATIQUE 2

L'exponentielle et XCAS

I - Méthode d'Euler

a. Approximation affine

On s'intéresse aux fonctions f dérivables sur $\mathbb{R}$ satisfaisant la condition

$$f'(x) = k \cdot f(x) \qquad (1)$$

pour tout $x \in \mathbb{R}$, avec k un réel arbitrairement fixé.
On va tenter, dans cette section, d'obtenir une approximation de l'allure de la courbe représentative d'une solution de (1).
On va pour cela considérer que pour h « suffisamment petit » et pour tout réel a, $f'(a) \approx \frac{f(a+h)-f(a)}{h}$.
Montrez alors que $f(a+h) \approx (1+kh)f(a)$.

b. Subdivision

Pour obtenir ce tracé, nous allons choisir un réel k, un segment $[a\,;b]$, nous allons fixer l'image de a par f pour avoir une solution unique de (1) dans un premier temps.
Nous allons subdiviser le segment $[a\,;b]$ en N segments de même longueur : quelle sera la longueur de chaque petit segment ?

c. Tracé d'une ligne brisée

Ouvrons une fenêtre de géométrie en tapant simultanément sur Alt et g.
On définit un point dont on connaît les coordonnées avec la commande `point` :

```
A:=point(1,2);
B:=point(3,-1);
C:=point(4,5);
```

On relie ces points par des segments de droite grâce à la commande `polygone_ouvert(liste de points)` :

```
polygone_ouvert(A,B,C)
```

d. Tracé en boucle

Oublions pour un temps notre problème et voyons comment nous pourrions tracer « point par point » sur $[-3\,;3]$ la courbe représentative d'une fonction g vérifiant :

$$\frac{g(b)-g(a)}{b-a} = 2$$

pour tous réels distincts a et b de $[-3\,;3]$ et $f(-3) = -1$.
On peut par exemple subdiviser le segment $[-3\,;3]$ en segments de longueurs 0,25 et réfléchir au moyen d'obtenir les images par f de chacune des extrémités des segments de la subdivision.
Il suffit de penser que $\frac{g(x+0{,}25)-g(x)}{x+0{,}25-x} = 2$, c'est-à-dire $g(x+0{,}25) = \ldots\ldots + g(x)$.
On peut donc calculer $g(x+0{,}25)$ si l'on connaît $g(x)$.
On va donc partir du point de coordonnées $(-3\,;-1)$ et obtenir de proche en proche les coordonnées de plusieurs points de la courbes en faisant des petits sauts de 0,25 et en s'arrêtant à 3 :

```
S:=NULL; // on crée une suite de points  vide au départ
X:=-3;  // au départ X vaut -3
Y:=-1;  // au départ Y vaut -1
tantque X<=3  faire  // tant que s'écrit tantque en XCAS
   S:=S,point(X,Y);  // on  rajoute le point de coordonnées (X,Y) à notre liste
   X:=X+0.25; // on avance de 0,25 à chaque tour de boucle
   Y:=0.5+Y; // on sait que g(X+0,25)=0,5+g(X)
ftantque:;  // f comme fin de la boucle
polygone_ouvert(S); // on relie les points de la liste S à la règle
```

et on découvre sans surprise qu'il s'agit d'un segment de droite.

e. Procédure

Cela fonctionne pour ce cas particulier mais ça devrait aussi « marcher » si l'on change les valeurs du pas, du coefficient directeur, des bornes de l'intervalle de définition, de l'image de la borne inférieure de cet intervalle.
On a donc envie de créer un fonction informatique, une *procédure*, qui reprend cette méthode mais dans le cas général, avec des coefficients quelconques qu'on donnera à l'ordinateur. On crée ainsi une sorte de fonction de plusieurs variables :
Notons donc h le pas, m le coefficient directeur, $[a\,;b]$ l'intervalle de définition et yo l'image de a.
Donnons également un nom à notre procédure, par exemple `TraceAffine`.
On va ouvrir une fenêtre de programmation en tapant Alt p :

```
TraceAffine(h,m,a,b,yo):={ // on précise quels sont les variables de notre procédure
S:=NULL; // on crée toujours une suite de points  vide au départ
X:=a;  // au départ X vaut a cette fois
Y:=yo;  // au départ Y vaut yo
tantque X<=b  faire  // on s'arrête à X=b
   S:=S,point(X,Y);  // on  rajoute le point de coordonnées (X,Y) à notre liste
   X:=X+h; // on avance de h à chaque tour de boucle
   Y:=h*m+Y; // on sait que g(X+h)=h*m+g(X)
ftantque:;  // f comme fin de la boucle
polygone_ouvert(S); // on relie les points de la liste S à la règle
}:;  // on termine la procédure en fermant l'accolade
```

Pour valider, on clique sur OK.
Pour exécuter cette procédure, on place le curseur sur une ligne de commande.
Par exemple, pour retrouver le cas précédent, on rentre :

```
TraceAffine(0.25,2,-3,3,-1)
```

f. À vous de jouer

Vous êtes maintenant armés pour adapter ce que nous venons de voir à la méthode d'Euler sachant que cette fois $f(X+h) = (1+kh)f(X)$ comme nous l'avons vu au **a.**
Construisez donc une procédure `Euler(h,k,a,b,yo)`...

g. Estimation de l'erreur

Nous avons vu en cours que la solution de l'équation différentielle $y' = y$ avec $y(0) = 1$ s'appelle la fonction exponentielle qui se note exp pour XCAS.
Nous avons utilisé une approximation pour obtenir son tracé par la méthode d'Euler. Il est bien sûr primordial de l'ordre de grandeur de l'erreur commise.
Nous avons écrit que $\exp(a+h) \approx (1+h) \times \exp(a)$.
Créons alors une fonction `Exp(h,x)` qui donne une approximation de e^x pour un h donné.
Il suffit d'utiliser la procédure précédente en ôtant la partie tracé et en prenant $a = 0$, $k = 1$, $b = x$:

```
Exp(h,x):={
X:=0;
Y:=1;
tantque X<=x  faire
   X:=X+h;
```

```
   Y:=(1+h)*Y;
ftantque:;
}:;
```

Par exemple, on peut comparer `Exp(h,1)` et exp(1) pour des valeurs successives de h entre 0,1 et 0,0001, puis en prenant d'autres valeurs de x.
Étudiez le rapport $\frac{Exp(0.01,x)-\exp(x)}{\exp(x)}$ pour différentes valeurs de x.
Que pensez-vous de l'approximation ?

On peut visualiser l'erreur commise en créant une fonction `erreur` :

```
erreur(x):=(1-Exp(0.01,x)/evalf(exp(x)))*100
```

Puis en créant la suite des points de coordonnées (x,erreur(x)) pour x variant de 0 à 100 avec un pas de 1.
La commande `seq` comme séquence[a] permet de créer cette suite de valeurs :

```
seq(point(x,erreur(x)),x=0..100)
```

Des commentaires ?

II - Exploration libre de la fonction exponentielle

Lors de la recherche d'exercices, l'utilisation des capacités formelles de XCAS peut s'avérer utile pour illustrer ses recherches, vérifier ses résultats, guider son intuition.
Pour obtenir un graphe, on utilise `graphe` :

```
graphe(exp(1-x),x=-5..5)
```

Pour la dérivée, on peut obtenir $f'(x)$ de plusieurs manières. Directement :

```
diff(x*exp(x^2+1))
```

Puis on factorise car c'est le signe qui nous intéresse :

```
factoriser(ans())
```

On peut aussi définir la fonction dérivée :

```
f:=x->x*exp(x^2+1);
fp:=fonction_derivee(f);
fp(3);
fp(a);
factoriser(fp(t));
```

On peut calculer des limites :

```
g(x):=(exp(x^2)-1)/x^2;
limite(g(x),x,0)
```

Et pour regarder à droite et à gauche :

```
h(x):=(exp(x^2)-1)/x^3;
limite(h(x),x,0,1);
limite(h(x),x,0,-1);
```

C'est tout pour l'instant.
Commencez à explorer les exercices du chapitre...

[a] C'est-à-dire suite