Implémentation de la méthode de Runge Kutta

Considérons le problème de Cauchy

$$\begin{cases} x'(t) &= f(t, x(t)) \\ x(t_0) &= x_0 \end{cases}$$

sur un ouvert Ω. Notons $\varphi :]a; b[\to \mathbb{R}$ la solution maximale de ce système. Le but est d'approcher numériquement φ.

1 Les méthodes à pas simple

Soit $H > 0$ tel que $[x_0; x_0 + H] \subset]a, b[$. Fixonx $M \in \mathbb{N}_0$, posons $h = \frac{H}{M}$ et $t_m = t_0 + mh$ pour tout $m \in \{0, ..., M\}$. Supposons avoir une fonction $\underline{f}(h, t, x)$ continue sur un ouvert contenant $\{0\} \times \Omega$, localement lipschitzien par rapport à x et tel que $\underline{f}(0, t_0, x_0) = f(t_0, x_0)$.

Définissons la relation de récurrence

$$\begin{cases} \underline{x}_{m+1} &= \underline{x}_m + h\underline{f}(h, t_m, \underline{x}_m) \\ \underline{x}_0 &= x_0 \end{cases}$$

pour tout $m \in \{0, ..., M-1\}$.

Si M est suffisemment grand, $\underline{x}_m$ approxime $\varphi(t_m)$. Plus précisémment, si on pose

$$\underline{\varphi}(h, t) = \underline{x}_m + (t - t_m)\underline{f}(h, t_m, \underline{x}_m)$$

pour tout $t \in [t_m; t_{m+1}[$ et tout $m \in \{0, ..., M-1\}$ alors l'application $t \mapsto \underline{\varphi}(h, t)$ est affine par morceaux sur $[t_0; t_0 + H]$ et

$$\underline{\varphi}(h, t) \to \varphi(t) \qquad \text{si } h \to 0$$

uniformément en t sur $[t_0; t_0 + H]$.

2 Runge-Kutta

La méthode de Runge-Kutta est une méthode à pas simple où la fonction $\underline{f}$ est définie par

$$\underline{f}(h, t_m, \underline{x}_m) = \sum_{j=1}^{n} c_j f(t_{m,j}, \underline{x}_{m,j})$$

où $t_{m,j} = t_m + \rho_j h$ et

$$\underline{x}_{m,j} = \underline{x}_m + h \sum_{k=1}^{j-1} a_{j,k} f(t_{m,k}, \underline{x}_{m,k}).$$

Les coefficients $(\rho_j)_j$, $(c_j)_j$ et $(a_{j,k})_{jk}$ sont définis dans le tableau suivant :

$$\begin{array}{c|cccccc} \rho_1 & a_{1,1} & & & & & \\ \rho_2 & a_{2,1} & 0 & & & & \\ \vdots & \vdots & & \ddots & & & \\ \rho_j & a_{j,1} & \dots & a_{j,j-1} & 0 & & \\ \vdots & \vdots & & & & \ddots & \\ \rho_n & a_{n,1} & \dots & \dots & \dots & a_{n,n-1} & 0 \\ \hline & c_1 & \dots & \dots & \dots & \dots & c_n \end{array}$$

où $\rho_1 = a_{1,1} = 0$, $\rho_j = \sum_{k=1}^{j-1} a_{j,k}$ et $\sum_{j=1}^{n} c_j = 1$. La méthode est au moins d'ordre 2 si $\sum_{j=1}^{n} c_j \rho_j = \frac{1}{2}$.

On vous demande d'implémenter en langage C la méthode de Runge-Kutta.

3 Conseils d'implémentation

Pour le tableau associé à la méthode, utiliser la structure suivante :

```
struct tabRungeKutta{
   double *rho;
   double *c;
   double **a;
   int n;
};
```

Découper votre code avec les fonctions suivantes :

```
//la fonction f de l'équation différentielle x'(t)=f(t,x(t))
double f(double t, double x);
//la fonction f_barre(h,tm,x_barre_m)=somme des cj f(tmj,x_barre_mj)
double fBarre(double h, double tm,double xm, tabRungeKutta* tab);
//la fonction x_barre_mj=x_barre_m+h*somme des ajk f(tmk,x_barre_mk)
double xBarre(int tm, int j, double xm, double h, tabRungeKutta* tab);
//calcul de x_barre_0, x_barre_1,...,x_barre_M
double* rungeKutta(tabRungeKutta* tab, double x0, double t0, double H, int M);
//enregistrement dans un fichier de deux colonnes, dont la première contient les tm et la deuxième les x_barre_m correspondant
void saveInFile(char* nameFile, double *xBarreM, double t0, double H, double M);
```

Tester votre code avec le tableau suivant (la méthode est d'ordre 4) :

0	0			
1/2	1/2	0		
1/2	0	1/2	0	
1	0	0	1	0
	1/6	2/6	2/6	1/6

et l'équation différentielle de la liste 5, exercice 2, point (d) c'est à dire

$$x'(t) + tx(t) = tx^3(t).$$

On a $f(t,x) = -tx + tx^3$. Avec la condition initial $x(0) = x_0$, on obtient

$$x(t) = \begin{cases} x_0 & \text{si } x_0 \in \{-1, 0, 1\} \\ \dfrac{\text{sign}(x_0)}{\sqrt{1+e^{t^2}\left(\frac{1}{x_0^2}-1\right)}} & \text{sinon} \end{cases}$$

pour tout $t \in]-t_M; t_M[$ avec $t_M = \sqrt{\ln\left(\frac{x_0^2}{x_0^2-1}\right)}$ si $x_0 \in]-\infty; -1[\cup]1; +\infty[$ et pour tout $t \in \mathbb{R}$ si $x_0 \in [-1; 1]$.