L2 - Méthodes numériques
Université de Nice-Sophia Antipolis
Sylvain Rubenthaler
`http://math.unice.fr/~rubentha/cours.html`

TP n$^{\mathbf{o}}$ 4

Toutes les réponses s'écrivent dans un programme.

1. Définir une fonction `f` (par exemple le cos).
2. Définir une fonction `interpol` qui prend en argument :

$$x \in \mathbb{R}\,,\ n \in \mathbb{N}^*\,,\ A = \begin{bmatrix} x_1 & y_1 \\ \vdots & \vdots \\ x_n & y_n \end{bmatrix} \in \mathcal{M}_{n,2}(\mathbb{R})$$

   et qui renvoie $P(x)$ où P est le polynôme d'interpolation de Lagrange aux points $(x_1, y_1), \ldots, (x_n, y_n)$ (rappel : $P(X) = \sum_{i=1}^{n} y_i \prod_{1 \leq j \leq n, j \neq i} \left(\frac{X - x_j}{x_j - x_i} \right)$).
3. Écrire un programme qui dessine f sur un intervalle $[a; b]$ (par exemple : $a = 0$, $b = 3\pi$) ainsi que le polynôme de Lagrange interpolant f aux points $a, a + \frac{b-a}{n-1}, a + 2 \times \frac{b-a}{n-1}$, ...,$b$ (prendre par exemple $n = 7$).

Corrigé du TP no. 4

1.

Algorithme 1

```
function z=f(x)
  z=cos(x);
endfunction
```

2.

Algorithme 2

```
function z=interpol(x,n,X)
  s=0;
  for i=1:n
    p=1;
    for j=1:n
      if (j<>i) then
        p=p*(x-X(j,1))/(X(i,1)-X(j,1));
      end
    end
    s=s+X(i,2)*p;
  end
  z=s;
endfunction
```

3.

Algorithme 3

```
//interpolation de Lagrange
delta=0.01;
a=0;
b=3*%pi;
n=7;
over=1;
X=[];//on commence avec X matrice vide
for i=0:n-1
  X=[X; a+(b-a)*i/(n-1) f(a+(b-a)*i/(n-1))];//on remplit
end
clf();
drawaxis(x=[-10,20,0,1],y=0,dir='u',tics='i',seg=1);//dessine
un axe horizontal plot(X(:,1),X(:,2),'rd'); //dessine
les points d'interpolations avec des diamants rouges
t=[a-over:delta:b+over];//remplit un vecteur avec des reels entre a-over
et b+over, espaces de delta
plot(t,feval(t,f));//dessine le graphe de f, feval(t,f) contient les
valeurs de f en les coordonnees de t
Y=[];//on commence avec Y matrice vide
for i=1:length(t)
  Y=[Y interpol(t(i),n,X)];//on remplit Y avec les valeurs du polynome
de Lagrange
end
plot(t,Y,'r'); //dessine le graphe du polynome
```