1] Ecrire une fonction Scilab qui prend en entrée les coordonnées $(x_i, y_i)$ de $n+1$ points et donne en sortie le tableau des différences divisées (Tableau de Nelville)

Construire le tableau de Nelville pour la fonction $f(x) = 1 + \sin(3x)$ aux points

$x \in \{0, 0.2, 0.4, 0.8, 1.2, 1.6, 2.0\}$

2] On va approcher la fonction

$$f(x) = \frac{1}{1+x^2} \qquad x \in [-5, 5]$$

par le polynôme d'interpolation sur des points equidistants dans l'intervalle donné. Visualiser sur un même graphique la fonction $f$ et les polynômes de degré 5 et 10.

3] Sur l'intervalle $[-1, 1]$ on souhaite interpoler la fonction $f(x) = \sin(2\pi x)$ en 22 points $x_i$ équidistants. On genere en suite un ensemble de valeurs perturbées $\tilde{f}(x_i)$ par rapport aux valeurs $f(x_i) = \sin(2\pi x_i)$ avec

$$\max_{i=1,\dots,22} |f(x_i) - \tilde{f}(x_i)| \sim 9.5\ 10^{-4}.$$

On compare donc les polynômes d'interpolation de degré 21 aux points $(x_i, f_i)$ et $(x_i, \tilde{f}_i)$. Noter la difference entre les 2 polynômes aux extremités de l'intervalle par rapport à l'erreur initiale sur les données.

La raison est due au fait que pour des points d'interpolation equidistants, la constant de Lebesgue se comporte comme

$$\Lambda_n([-1,1]) \simeq \frac{2^{n+1}}{e \cdot n \cdot \log n}$$

et

$$\| p_n - \tilde{p}_n \|_\infty = \max_{[-1,1]} \left| \sum_{i=1}^{n+1} \left( f(x_i) - \tilde{f}(x_i) \right) l_i(x) \right|$$

$$\leq \Lambda_n([-1,1]) \max_{i=1,n+1} \left| f(x_i) - \tilde{f}(x_i) \right|$$

Repeter l'exercice (3) en choisissant 22 points $x_i$ de Chebychev sur $[-1,1]$

$$x_i = -\cos\left(\pi \frac{i}{n}\right) \quad \cancel{i} = 0, \dots, n$$

4] Evaluer l'erreur d'interpolation pour les polynômes dans (3), en calculant le max $|f(x) - p(x)|$ pour $x$ variable dans un ensemble de points comme indiqué :

- si $p$ est associé à des noeuds equidist. prendre $x$ parmis les noeuds de Cheb.
- si $p$ est associé aux points de Cheb. prendre $x$ parmis les points equidist.

Remplir donc le Tableau suivant

| $n$ | 5 | 10 | 20 | 40 |
|---|---|---|---|---|
| $Err_{equid}$ | | | | |
| $Err_{cheb}$ | | | | |

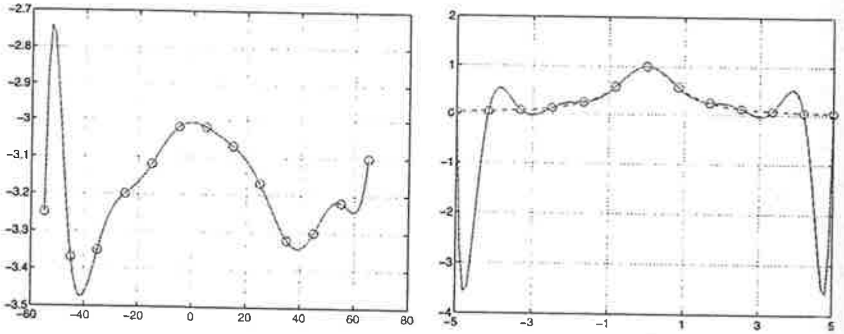

**Figure 3.6.** Deux exemples du phénomène de Runge : *à gauche*, $\Pi_{12}$ calculé avec les données de la Table 3.1, colonne $K = 0.67$ ; *à droite*, $\Pi_{12}f$ *(trait plein)* calculé avec 13 noeuds équidistants pour la fonction $f(x) = 1/(1+x^2)$ *(trait discontinu)*

polyder

Dans MATLAB, $(\Pi_n f)'$ peut être calculé avec `[d]=polyder(c)`, où `c` est le vecteur d'entrée contenant les coefficients du polynôme d'interpolation, et `d` est le vecteur de sortie contenant les coefficients de sa dérivée première (voir Section 1.4.2).

Octave 3.1 L'analogue dans Octave est `d=polyderiv(c)`. ■

Voir les Exercices 3.1-3.4.

### 3.1.2 Interpolation de Chebyshev

On peut éviter le phénomène de Runge en choisissant correctement la distribution des noeuds d'interpolation. Sur un intervalle $[a, b]$, on peut par exemple considérer les ***noeuds de Chebyshev*** (voir Figure 3.7, *à droite*) :

$$\boxed{x_i = \frac{a+b}{2} + \frac{b-a}{2}\widehat{x}_i, \text{ où } \widehat{x}_i = -\cos(\pi i/n), \quad i = 0, \ldots, n} \tag{3.8}$$

On a bien sûr $x_i = \widehat{x}_i$, $i = 0, \ldots, n$, quand $[a, b] = [-1, 1]$. Pour cette distribution particulière de noeuds, il est possible de montrer que, si $f$ est dérivable sur $[a, b]$, alors $\Pi_n f$ converge vers $f$ quand $n \to \infty$ pour tout $x \in [a, b]$.

Les noeuds de Chebyshev, qui sont les abscisses des noeuds équirépartis sur le demi-cercle unité, se trouvent à l'intérieur de $[a, b]$ et sont regroupés près des extrémités de l'intervalle (voir Figure 3.7).

Une autre distribution non uniforme des noeuds dans l'intervalle $]a, b[$, possédant les mêmes propriétés de convergence que les noeuds de



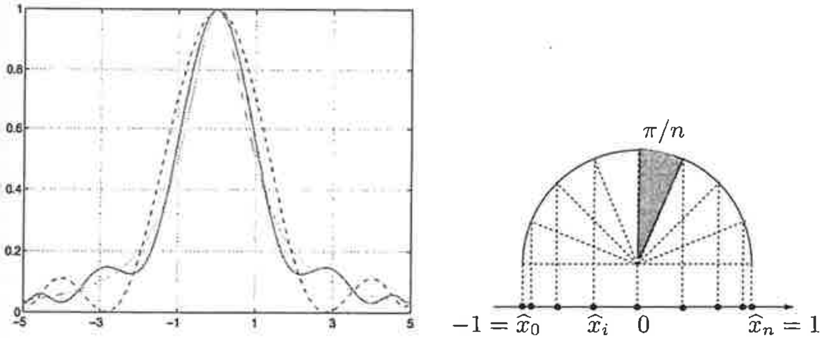


**Figure 3.7.** L'image de gauche montre une comparaison entre la fonction $f(x) = 1/(1+x^2)$ *(trait plein fin)* et ses polynômes d'interpolation de Chebyshev de degré 8 *(trait discontinu)* et 12 *(trait plein)*. Remarquer que l'amplitude des oscillations parasites décroît quand le degré croît. L'image de droite montre la distribution des noeuds de Chebyshev sur l'intervalle $[-1, 1]$

$$\boxed{x_i = \frac{a+b}{2} - \frac{b-a}{2}\cos\left(\frac{2i+1}{n+1}\frac{\pi}{2}\right), \quad i = 0, \ldots, n} \tag{3.9}$$

**Exemple 3.3** On considère à nouveau la fonction $f$ de l'exemple de Runge et on calcule son polynôme d'interpolation aux noeuds de Chebyshev. Ces derniers peuvent être obtenus avec les instructions MATLAB suivantes :

```
» xc = -cos(pi*[0:n]/n); x = (a+b)*0.5+(b-a)*xc*0.5;
```

où `n+1` est le nombre de noeuds, et `a` et `b` les extrémités de l'intervalle d'interpolation (dans la suite on choisit `a=-5` et `b=5`). On calcule alors le polynôme d'interpolation avec les instructions :

```
» f= '1./(1+x.^2)'; y = eval(f); c = polyfit(x,y,n);
```

On calcule enfin le maximum des valeurs absolues des différences entre $f$ et son interpolée de Chebyshev en 1001 points équidistants de l'intervalle $[-5, 5]$ :

```
» x = linspace(-5,5,1000); p=polyval(c,x);
» fx = eval(f); err = max(abs(p-fx));
```

Comme le montre la Table 3.3, le maximum de l'erreur décroît quand $n$ augmente. ■

| $n$ | 5 | 10 | 20 | 40 |
|---|---|---|---|---|
| $E_n$ | 0.6386 | 0.1322 | 0.0177 | 0.0003 |

**Table 3.3.** Erreur de l'interpolation de Chebyshev pour la fonction de Runge $f(x) = 1/(1+x^2)$

The coefficients involved in the Newton formula are the diagonal entries of the matrix.

**Program 66 - dividif** : Newton divided differences

```
function [d]=dividif(x,y)
[n,m]=size(y);
if n == 1, n = m; end
n = n-1;   d = zeros (n+1,n+1);   d (:,1) = y';
for j = 2:n+1
  for i = j:n+1
    d (i,j) = ( d (i-1,j-1)-d (i,j-1))/(x (i-j+1)-x (i));
  end
end
```

Using (8.19), $n(n+1)$ sums and $n(n+1)/2$ divisions are needed to generate the whole matrix. If a new evaluation of $f$ were available at a new node $x_{n+1}$, only the calculation of a new row of the matrix would be required ($f[x_n, x_{n+1}]$, $\ldots$, $f[x_0, x_1, \ldots, x_{n+1}]$). Thus, in order to construct $\Pi_{n+1}f$ from $\Pi_n f$, it suffices to add to $\Pi_n f$ the term $a_{n+1}\omega_{n+1}(x)$, with a computational cost of $(n+1)$ divisions and $2(n+1)$ sums. For the sake of notational simplicity, we write below $D^r f_i = f[x_i, x_{i+1}, \ldots, x_r]$.

**Example 8.3** In Table 8.1 we show the divided differences on the interval (0,2) for the function $f(x) = 1+\sin(3x)$. The values of $f$ and the corresponding divided differences have been computed using 16 significant figures, although only the first 5 figures are reported. If the value of $f$ were available at node $x = 0.2$, updating the divided difference table would require only to computing the entries denoted by italics in Table 8.1. •

| $x_i$ | $f(x_i)$ | $f[x_i, x_{i-1}]$ | $D^2 f_i$ | $D^3 f_i$ | $D^4 f_i$ | $D^5 f_i$ | $D^6 f_i$ |
|---|---|---|---|---|---|---|---|
| 0 | 1.0000 | | | | | | |
| *0.2* | *1.5646* | *2.82* | | | | | |
| 0.4 | 1.9320 | *1.83* | *-2.46* | | | | |
| 0.8 | 1.6755 | -0.64 | *-4.13* | *-2.08* | | | |
| 1.2 | 0.5575 | -2.79 | -2.69 | *1.43* | *2.93* | | |
| 1.6 | 0.0038 | -1.38 | 1.76 | 3.71 | *1.62* | *-0.81* | |
| 2.0 | 0.7206 | 1.79 | 3.97 | 1.83 | -1.17 | *-1.55* | *-0.36* |

TABLE 8.1. Divided differences for the function $f(x) = 1 + \sin(3x)$ in the case in which the evaluation of $f$ at $x = 0.2$ is also available. The newly computed values are denoted by italics



Notice that $f[x_0, \ldots, x_n] = 0$ for any $f \in \mathbb{P}_{n-1}$. This property, however, is not always verified numerically, since the computation of divided differences might be highly affected by rounding errors.

**Example 8.4** Consider again the divided differences for the function $f(x) = 1 + \sin(3x)$ on the interval $(0, 0.0002)$. The function behaves like $1 + 3x$ in a sufficiently small neighbourhood of 0, so that we expect to find smaller numbers as the order of divided differences increases. However, the results obtained running Program 66, and shown in Table 8.2 in exponential notation up to the first 4 significant figures (although 16 digits have been employed in the calculations), exhibit a substantially different pattern. The small rounding errors introduced in the computation of divided differences of low order have dramatically propagated on the higher order divided differences. •

| $x_i$ | $f(x_i)$ | $f[x_i, x_{i-1}]$ | $D^2 f_i$ | $D^3 f_i$ | $D^4 f_i$ | $D^5 f_i$ |
|---|---|---|---|---|---|---|
| 0 | 1.0000 | | | | | |
| 4.0e-5 | 1.0001 | 3.000 | | | | |
| 8.0e-5 | 1.0002 | 3.000 | -5.39e-4 | | | |
| 1.2e-4 | 1.0004 | 3.000 | -1.08e-3 | -4.50 | | |
| 1.6e-4 | 1.0005 | 3.000 | -1.62e-3 | -4.49 | 1.80e+1 | |
| 2.0e-4 | 1.0006 | 3.000 | -2.15e-3 | -4.49 | -7.23 | $\boxed{-1.2e+5}$ |

TABLE 8.2. Divided differences for the function $f(x) = 1+\sin(3x)$ on the interval (0,0.0002). Notice the completely wrong value in the last column (it should be approximately equal to 0), due to the propagation of rounding errors throughout the algorithm

### 8.2.2 *The Interpolation Error Using Divided Differences*

Consider the nodes $x_0, \ldots, x_n$ and let $\Pi_n f$ be the interpolating polynomial of $f$ on such nodes. Now let $x$ be a node distinct from the previous ones; letting $x_{n+1} = x$, we denote by $\Pi_{n+1} f$ the interpolating polynomial of $f$ at the nodes $x_k$, $k = 0, \ldots, n+1$. Using the Newton divided differences formula, we get

$$\Pi_{n+1} f(t) = \Pi_n f(t) + (t - x_0)\ldots(t - x_n) f[x_0, \ldots, x_n, t].$$

Since $\Pi_{n+1} f(x) = f(x)$, we obtain the following formula for the interpolation error at $t = x$

$$\begin{aligned} E_n(x) &= f(x) - \Pi_n f(x) = \Pi_{n+1} f(x) - \Pi_n f(x) \\ &= (x - x_0)\ldots(x - x_n) f[x_0, \ldots, x_n, x] \end{aligned} \qquad (8.20)$$