Analyse Numérique

Travaux Pratiques - Série 2

Interpolation polynomiale

Le but de l'interpolation polynomiale est de construire un polynôme $p_n(x)$ de degré au maximum n satisfaisant

$$p_n(x_i) = y_i \quad \text{pour} \quad i = 0, 1, 2, \dots, n,$$

où les (x_i, y_i) sont les points d'interpolation donnés avec $a \leq x_0 < x_1 < \cdots < x_n \leq b$. Dans les exercices suivants, on discutera de la construire le polynôme d'interpolation.

1. Polynômes de Lagrange.

1. Les fonctions ℓ_k sont des polynômes de degré n. Écrire une fonction de MATLAB

   `c=coeff_LK(k,nodes)`

   qui calcule les coefficients de ℓ_k, étant donné les (x_k) comme le vecteur `nodes`. On peut utiliser les coefficients `c` pour évaluer ℓ_k en x comme ci :

   `Lx=polyval(c,x)`.

   Indication 1 : On peut utiliser `poly(r)`.
   Indication 2 : `sum(a)` et `prod(a)` calculent la somme et le produit des éléments du `a`.

2. Utiliser `coeff_LK` pour écrire une fonction MATLAB

   `p=lagrangepoly(nodes, data)`

   qui calcule les coefficients de Lagrange avec `nodes` est un vecteur qui contient les x_i et `data` est le vecteur des images. Vérifier votre programme en interpolant la fonction

   $$f(x) = \sin(10 \log(1 + x))$$

   sur $[0, 1]$. Utiliser $n = 5, 15$ et des `nodes` équidistants. Reproduire la figure 1.

3. Qu'est-ce qui se passe quand on interpole

   $$f(x) = 2x^3 + x - 1$$

   en utilisant les mêmes points ?

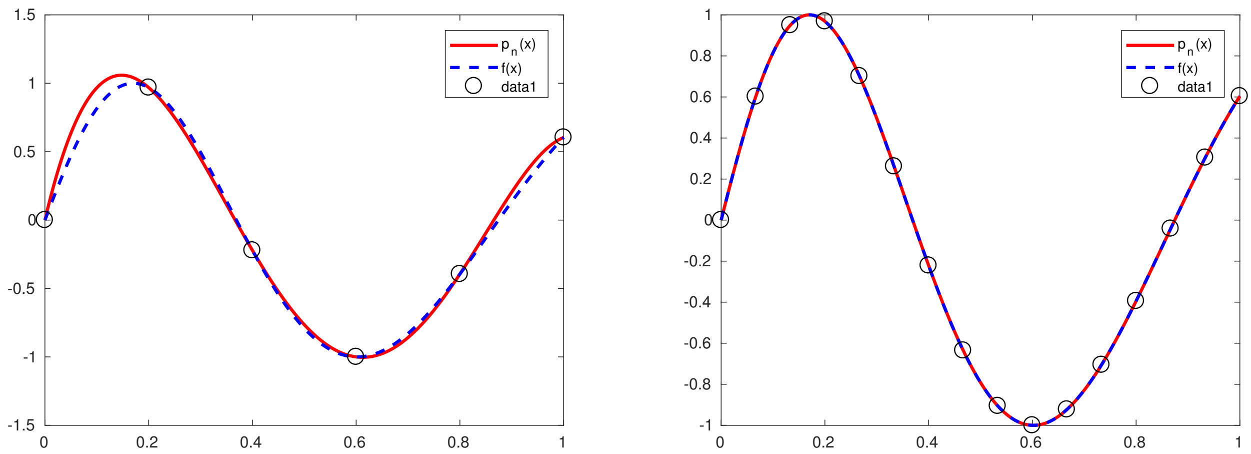

FIGURE 1 – Lagrange polynomial for $f(x) = \sin(10 \log(1+x))$ and nodes $x_i = \frac{i}{n}, i = 0, 1, ...n$ for $n = 5$ and $n = 15$.

2. Points d'interpolation. Dans les exercices précédents, on a utilisé des points d'interpolation équidistants. Dans la pratique, les abscisses de Chebyshev sont souvent utilisées. Les abscisses de Chebyshev dans l'intervalle $(-1, 1)$ sont définies par

$$x_i = \cos\left(\frac{2i+1}{2n+2}\pi\right), \quad \text{pour} \quad i = 0, \ldots, n.$$

a. Calculer les poids w_i pour $n = 50$ pour les points équidistants et les abscisses de Chebyshev. Vérifier numériquement les formules suivantes :

$$w_i = \frac{(-1)^{n-i}}{h^n\, n!}\binom{n}{i}, \quad \text{où} \quad h = 2/n \ \ \text{(pas de discrétisation)}$$

pour les points équidistants, et

$$w_i = (-1)^i \frac{2^{n-1}}{n} \sin\theta_i, \quad \text{où} \quad \theta_i = \frac{(2i+1)\pi}{2n+2}, \quad \text{pour} \quad i = 0, ..., n$$

pour les abscisses de Chebyshev.

b. Utiliser les abscisses de Chebyshev comme points d'interpolation pour les deux fonctions suivantes sur l'intervalle $(-1, 1)$

$$f(x) = |x| + x/2 - x^2 \quad \text{et} \quad g(x) = \frac{1}{1+25x^2}.$$

Comparer le résultat avec le cas de points équidistants – voir par exemple la Figure 2. (**Indice** : Utiliser un repère semi-logarithmique `semilogy` pour tracer le graphique de l'erreur, car autrement dans une échelle normale l'erreur serait trop petite pour être discernée.)

3. Formule de Newton.

a. On peut aussi construire le polynôme d'interpolation en utilisant la formule de Newton

$$p_n(x) = c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1) + \cdots + c_n(x - x_0)\cdots(x - x_{n-1}), \tag{1}$$

où

$$c_k = \delta^k y[x_0, x_1, \ldots, x_k], \quad \text{pour} \quad k = 0, \ldots, n,$$

sont les différences divisées (voir la Définition 1.2 dans les polycopiés du cours).

(i) Écrire une fonction Matlab pour générer les coefficients $[c_0, c_1, \ldots, c_n]$ pour des noeuds différents deux à deux.

```
function c = differences_divisees(data)
%  La fonction y = differences_divisees(x, data)
% input: data=[x_0, x_1,...,x_n;
%              y_0, y_1,...,y_n].
% output: les valeurs différences divisées c_k
```

(ii) Vérifier que les coefficients $[c_0, c_1, ..., c_n]$ ne dépendent pas de l'ordre des couples (x_i, y_i).

b. Écrire une fonction Matlab pour évaluer $p_n(x)$ de l'équation (1), étant donnés $\{x_i\}_{i=0}^n$, $\{c_i\}_{i=0}^n$, et le point d'évaluation `x`. Assurez-vous que `x` peut être un vecteur.

c. Comme dans l'exercice (1.c), essayer d'interpoler la fonction $f(x)$ en utilisant votre programme.

Bonus : ($\bigstar$) Si on ajoute un nouveau point d'interpolation (x_{n+1}, y_{n+1}), comment peut-on calculer efficacement $p_{n+1}(x)$ à partir de $p_n(x)$?

d. Est-il possible de traiter le cas où $x_{n-1} = x_n$ et $y_n = f'(x_n)$? Dans ce cas $p_n(x)$ est un polynôme de Hermite de $f(x)$, comme expliqué dans la Section 1.5 des polycopiés.

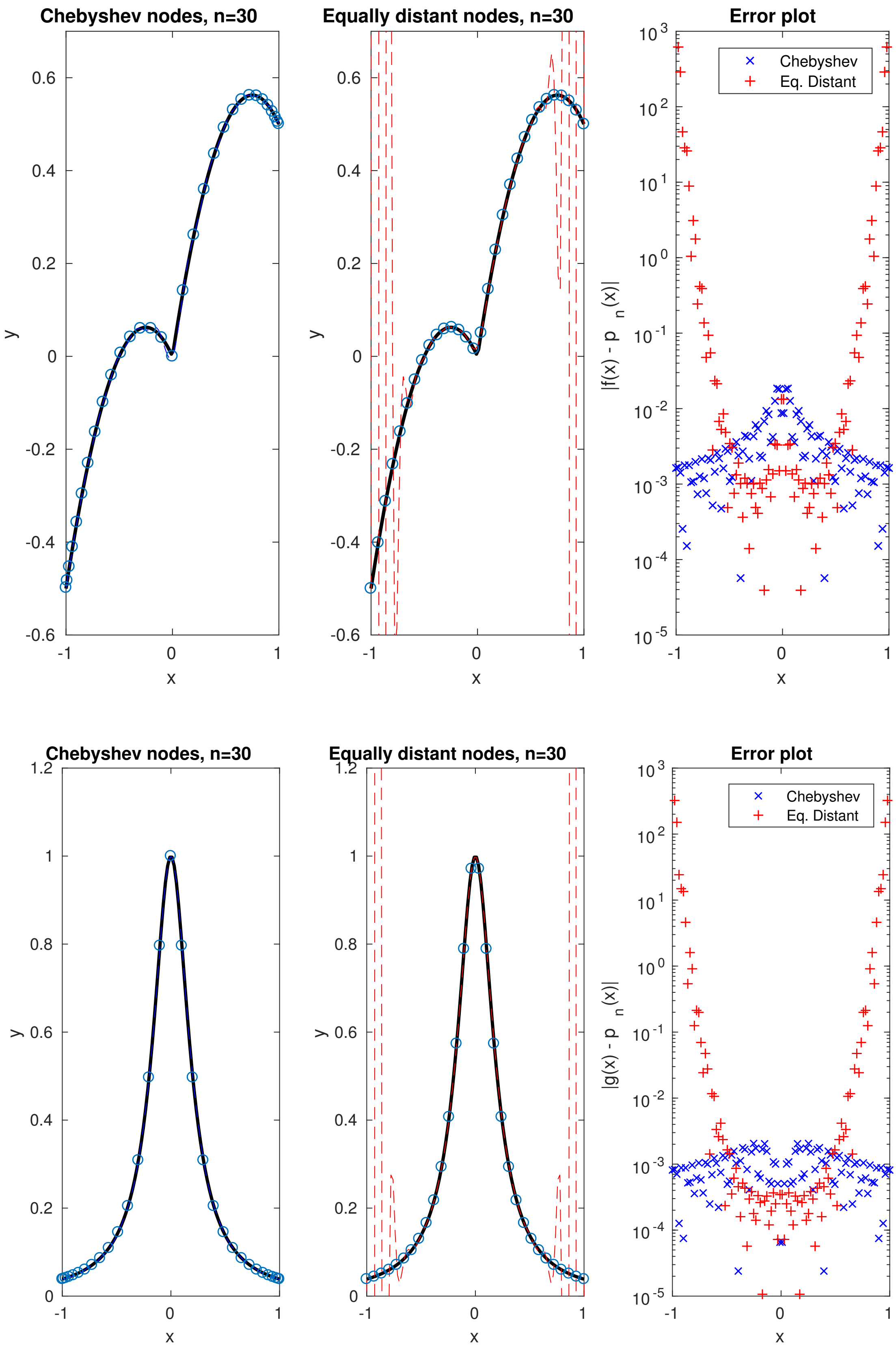

FIGURE 2 – **En haut** : La fonction $f(x) = |x| + x/2 - x^2$ (ligne noire, solide) et son polynôme d'interpolation $p_n(x)$ (ligne rouge, discontinue). Les points d'interpolation sont dessinés sous forme de $\circ$. En haut à droite : erreur absolue $|f(x) - p_n(x)|$. **En bas** : La fonction de Runge $g(x) = (1 + 25x^2)^{-1}$ (ligne noire, solide) et son polynôme d'interpolation $p_n(x)$ (ligne rouge, discontinue). En bas à droite : erreur absolue $|g(x) - p_n(x)|$.