Les polynômes

Pascal Romon

Définition des polynômes

Définition

Un polynôme (réel) $P \in \mathbb{R}[X]$ est une fonction de la forme

$$P : x \in \mathbb{R} \mapsto a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$

Son degré est $\deg P = \max_{a_i \neq 0} i$.
Si $P(z) = 0$ on dit que z est une racine de P.

Intérêt en informatique ? : un polynôme de degré n code une fonction complexe mais est défini par seulement $n+1$ coefficients réels.

- $\deg(P+Q) \leqslant \max(\deg(P), \deg(Q))$
- $\deg(PQ) = \deg(P) + \deg(Q)$
- quel est le degré du polynôme nul ?

Théorème fondamental de l'algèbre

Théorème de d'Alembert-Gauss

Tout polynôme réel de degré n a n racines complexes (comptées avec multiplicité) et peut s'écrire

$$a_n(x - z_1)(x - z_2)\dots(x - z_n).$$

Exemples:

- $x^2 - 1 = (x - 1)(x + 1)$
- $x^2 + 1 = (x - i)(x + i)$,
- $x^3 - 1 = (x - 1)(x - j)(x - j^2)$ où $j = \cos(2\pi/3) + i\sin(2\pi/3)$.

Évaluation d'un polynôme

Introduction :

Soit un polynôme P à évaluer en x :

$$P(x) = a_n x^n + ... + a_3 x^3 + a_2 x^2 + a_1 x + a_0$$

Évaluation d'un polynôme

Méthode naïve :

$$P(x) = \sum_{i=0}^{n} a_i x^i$$

Calculer à chaque fois x^i comporte une grosse redondance (mais qu'on peut compenser en mémorisant les x^i au fur et à mesure).

Méthode de Horner

Pour accélérer les calculs, on peut récrire $P(x)$:

$$P(x) = \left(\left(\ldots((a_n x + a_{n-1})x + a_{n-2})x + \ldots\right)x + a_1\right)x + a_0$$

Comment procède-t-on?

Méthode de Horner

$$P(x) = \Big(\big((a_4 x + a_3)x + a_2\big)x + a_1\Big)x + a_0$$

$$\begin{aligned}
b_4 &= a_4 & & \\
b_3 &= a_4 x + a_3 & b_3 &= b_4 x + a_3 \\
b_2 &= (a_4 x + a_3)x + a_2 & b_2 &= b_3 x + a_2 \\
b_1 &= ((a_4 x + a_3)x + a_2)x + a_1 & b_1 &= b_2 x + a_1 \\
b_0 &= (((a_4 x + a_3)x + a_2)x + a_1)x + a_0 & b_0 &= b_1 x + a_0
\end{aligned}$$

$$P(x) = b_0$$

Méthode de Horner

Algorithme :

Soit $P(x)$ un polynôme de degré $n \rightarrow n+1$ coefficients

On peut le représenter sous forme d'un tableau de coefficients :
`a[i]` $=$ i-ème coefficient du polynôme P.

Algorithm 1: Horner

---

b $=$ a$[n]$
for $i \leftarrow n$ **to** 0 **do**
  b $=$ b $\times$ x $+$ a$[i]$
end
return b

Méthode de Horner : en pratique

En pratique :

On peut approximer $\sin(x)$ par son développement de Taylor :

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \frac{x^{11}}{11!} + \frac{x^{13}}{13!} - \frac{x^{15}}{15!} + O(x^{16})$$

$\rightarrow$ on obtient un polynôme dont les coefficients sont :

```
double a0 = +1.0;
double a1 = -1.66666666666666666666666666666666667e-1;
double a2 = +8.33333333333333333333333333333333333e-3;
double a3 = -1.98412698412698412698412698412698413e-4;
double a4 = +2.75573192239858906525573192239858907e-6;
double a5 = -2.50521083854417187750521083854417188e-8;
double a6 = +1.60590438368216145993923771701549479e-10;
double a7 = -7.64716373181981647590113198578807044e-13;
```

Méthode de Horner : en pratique

Force brute :

avec une légère optimisation : on précalcule x^2.

```
double sin1(double x)
{
    double result;
    double x2 = x * x;
    result  = a0 * x; x *= x2;
    result += a1 * x; x *= x2;
    result += a2 * x; x *= x2;
    result += a3 * x; x *= x2;
    result += a4 * x; x *= x2;
    result += a5 * x; x *= x2;
    result += a6 * x; x *= x2;
    result += a7 * x;
    return result;
}
```

Complexité ? $\rightarrow$ 16 multiplications et 7 additions.

Méthode de Horner : en pratique

Méthode de Horner :

avec la même légère optimisation : on précalcule x^2.

```
double sin2(double x)
{
    double x2 = x * x;
    return x*(a0+x2*(a1+x2*(a2+x2*(a3+x2*(a4+x2*(a5+x2*(a6+x2*a7)))))));
}
```

$\rightarrow$ 9 multiplications et 7 additions.

Méthode de Horner : en pratique

compilation	$sin1$	$sin2$
normale	0.044 ns	0.031 ns
$-O2$	0.021 ns	0.024 ns

$\rightarrow$ Horner est parfois plus lent !!

Explications :
Certains *pipelines CPU* peuvent effectuer une opération et lire la variable de l'opération suivante en un seul cycle, sauf si la variable suivante dépend du résultat de l'opération courante. Dans ce cas, on perd à chaque fois un cycle.

$\rightarrow$ $sin1$: 30 cycles
$\rightarrow$ $sin2$: 48 cycles

Méthode de Horner et dérivées

Remarque :

La méthode de Horner permet aussi d'évaluer la dérivée n-ième du polynôme P en x.

Méthode de Horner et dérivées

$$P(x) = a_3x^3 + a_2x^2 + a_1x + a_0$$
$$P(x) = ((a_3x + a_2)x + a_1)x + a_0$$

$$\begin{array}{lcl}
b_3 = a_3 & & a_3 = b_3 \\
b_2 = b_3x_0 + a_2 & & a_2 = b_2 - b_3x_0 \\
b_1 = b_2x_0 + a_1 & \Leftrightarrow & a_1 = b_1 - b_2x_0 \\
b_0 = b_1x_0 + a_0 & & a_0 = b_0 - b_1x_0 \\
P(x_0) = b_0 & & a_0 = P(x_0) - b_1x_0
\end{array}$$

On réécrit l'équation de départ en remplaçant les a_i :

$$P(x) = b_3x^3 + (b_2 - b_3x_0)x^2 + (b_1 - b_2x_0)x + P(x_0) - b_1x_0$$

Méthode de Horner et dérivées

Développement et factorisation :

$$P(x) = b_3x^3 + (b_2 - b_3x_0)x^2 + (b_1 - b_2x_0)x + P(x_0) - b_1x_0$$

$$P(x) = b_3x^3 + b_2x^2 + b_1x - (b_3x^2 + b_2x + b_1)x_0 + P(x_0)$$

$$P(x) = (b_3x^2 + b_2x + b_1)x - (b_3x^2 + b_2x + b_1)x_0 + P(x_0)$$

$$P(x) = (b_3x^2 + b_2x + b_1)(x - x_0) + P(x_0)$$

Méthode de Horner et dérivées

On obtient une nouvelle écriture du polynôme :

$$P(x) = (\underbrace{b_3x^2 + b_2x + b_1}_{Q})(x - x_0) + P(x_0)$$

on recommence sur le polynôme $Q(x) = b_3x^2 + b_2x + b_1$

on obtient :

$$P(x) = ((c_2x + c_1)(x - x_0) + Q(x_0))(x - x_0) + P(x_0)$$

Méthode de Horner et dérivées

On obtient une nouvelle écriture du polynôme :

$$P(x) = ((\underbrace{c_2 x + c_1}_{M})(x - x_0) + Q(x_0))(x - x_0) + P(x_0)$$

on recommence sur le polynôme $M(x) = c_2 x + c_1$

on obtient :

$$P(x) = ((d_1(x - x_0) + M(x_0))(x - x_0) + Q(x_0))(x - x_0) + P(x_0)$$

Méthode de Horner et dérivées

Finalement :

$$P(x) = ((d_1(x - x_0) + M(x_0))(x - x_0) + Q(x_0))(x - x_0) + P(x_0)$$

développement :

$$P(x) = d_1(x - x_0)^3 + M(x_0)(x - x_0)^2 + Q(x_0)(x - x_0) + P(x_0)$$

soit dans l'autre sens :

$$P(x) = P(x_0) + Q(x_0)(x - x_0) + M(x_0)(x - x_0)^2 + d_1(x - x_0)^3$$

Méthode de Horner et dérivées

$$P(x) = P(x_0) + Q(x_0)(x - x_0) + M(x_0)(x - x_0)^2 + d_1(x - x_0)^3$$

ressemble au développement de Taylor de f en x_0 :

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + \frac{f'''(x_0)}{3!}(x - x_0)^3 + \ldots$$

Méthode de Horner et dérivées

$$P(x) = P(x_0) + Q(x_0)(x - x_0) + M(x_0)(x - x_0)^2 + d_1(x - x_0)^3 + 0$$

ressemble au développement de Taylor de f en x_0 :

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + \frac{f'''(x_0)}{3!}(x - x_0)^3 + \ldots$$

par identification :

$$P'(x_0) = Q(x_0) \qquad P'''(x_0) = d_1 3!$$
$$P''(x_0) = M(x_0)2! \qquad \ldots = 0$$

Méthode de Horner et dérivées

Pour évaluer $P(x_0)$, $P'(x_0)$, $P''(x_0)$, $P'''(x_0)$, ... il suffit d'appliquer le schéma de Horner plusieurs fois.

$$P(x) = a_3x^3 + a_2x^2 + a_1x + a_0$$

$$\begin{array}{c} P(x_0) \\ a_3 \\ a_2 \\ a_1 \\ a_0 \end{array}$$

Méthode de Horner et dérivées

Pour évaluer $P(x_0)$, $P'(x_0)$, $P''(x_0)$, $P'''(x_0)$, ... il suffit d'appliquer le schéma de Horner plusieurs fois.

$$P(x) = a_3x^3 + a_2x^2 + a_1x + a_0$$

$P(x_0)$	$Q(x_0)$
a_3	b_3
a_2	b_2
a_1	b_1
a_0	b_0

$$\rightarrow P(x) = (b_3x^2 + b_2x + b_1)(x - x_0) + P(x_0)$$

Méthode de Horner et dérivées

Pour évaluer $P(x_0)$, $P'(x_0)$, $P''(x_0)$, $P'''(x_0)$, ... il suffit d'appliquer le schéma de Horner plusieurs fois.

$$P(x) = (b_3x^2 + b_2x + b_1)(x - x_0) + P(x_0)$$

$P(x_0)$	$Q(x_0)$	$M(x_0)$
a_3	b_3	c_2
a_2	b_2	c_1
a_1	b_1	c_0
a_0	b_0	

$$\rightarrow P(x) = ((c_2x + c_1)(x - x_0) + Q(x_0))(x - x_0) + P(x_0)$$

Méthode de Horner et dérivées

Pour évaluer $P(x_0)$, $P'(x_0)$, $P''(x_0)$, $P'''(x_0)$, ... il suffit d'appliquer le schéma de Horner plusieurs fois.

$$P(x) = ((c_2 x + c_1)(x - x_0) + Q(x_0))(x - x_0) + P(x_0)$$

$P(x_0)$	$Q(x_0)$	$M(x_0)$	
a_3	b_3	c_2	d_1
a_2	b_2	c_1	d_0
a_1	b_1	c_0	
a_0	b_0		

$$\rightarrow P(x) = ((d_1(x - x_0) + M(x_0))(x - x_0) + Q(x_0))(x - x_0) + P(x_0)$$

Méthode de Horner et dérivées

Pour évaluer $P(x_0)$, $P'(x_0)$, $P''(x_0)$, $P'''(x_0)$, ... il suffit d'appliquer le schéma de Horner plusieurs fois.

$$P(x) = ((d_1(x - x_0) + M(x_0))(x - x_0) + Q(x_0))(x - x_0) + P(x_0)$$

$P(x_0)$	$Q(x_0)$	$M(x_0)$	
a_3	b_3	c_2	d_1
a_2	b_2	c_1	d_0
a_1	b_1	c_0	
a_0	b_0		

Méthode de Horner et dérivées

$P(x_0)$	$Q(x_0)$	$M(x_0)$	
a_3	b_3	c_2	d_1
a_2	b_2	c_1	d_0
a_1	b_1	c_0	
a_0	b_0		

$$P(x_0) = b_0 \qquad P''(x_0) = d_0 2!$$

$$P'(x_0) = c_0 \qquad P'''(x_0) = d_1 3!$$

Évaluation de polynômes en parallèle

Exemple : $P(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5$

$$
\begin{array}{c|c|c|c|c|c|c|c|c}
a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & 0 & 0 & \cdots
\end{array}
$$

$$
\begin{array}{c|c|c|c|c}
a_0 + a_1x & a_2 + a_3x & a_4 + a_5x & 0 + 0x & \cdots
\end{array}
$$

$$
\begin{array}{c|c|c}
a_0 + a_1x + (a_2 + a_3x)x^2 & a_4 + a_5x + (0 + 0)x^2 & \cdots
\end{array}
$$

$$
a_0 + a_1x + (a_2 + a_3x)x^2 + (a_4 + a_5x + (0+0)x^2)x^4 \quad \cdots
$$

Évaluation de polynômes en parallèle

Complexité : $O(\log_2 n)$

Propriété : meilleure précision numérique.

Racines de polynômes

Rappels :

- Racines du polynôme $P(x)$: $\{z, P(z) = 0\}$
- Racines : réelles / complexes

Racines de polynômes

Degré 2 :

$$P(x) = ax^2 + bx + c$$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Racines de polynômes

Degré 3 :

$$P(x) = ax^3 + bx^2 + cx + d$$

Formules de Cardan

- racines réelles ou complexes
- un peu long à mettre en place

Racines de polynômes

Degré 4 :

$$P(x) = ax^4 + bx^3 + cx^2 + dx + e$$

Formules de Ferrari

- racines réelles ou complexes
- assez long à mettre en place

Racines de polynômes

Degré 5 et plus :

Il n'y a pas de formule comme pour les degrés inférieurs (théorème d'Abel-Ruffini).
$\rightarrow$ Exemple intéressant de résultat négatif en mathématiques.

Racines de polynômes de degré n

n racines réelles :

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + ... + a_2 x^2 + a_1 x + a_0$$

Méthode QR :

- écrire une matrice **C** associée à P, appelée matrice compagnon
- trouver les valeurs propres de **C** (décomp. RQ)
- optimisation non-linéaire (Newton)

Racines de polynômes de degré n

Principe :

$$P(x) = a_n x^n + a_{n-1}x^{n-1} + ... + a_2 x^2 + a_1 x + a_0$$

a les mêmes racines que

$$Q(x) = x^n + \frac{a_{n-1}}{a_n}x^{n-1} + ... + \frac{a_2}{a_n}x^2 + \frac{a_1}{a_n}x + \frac{a_0}{a_n}$$

(qui est un polynôme unitaire)

Racines de polynômes de degré n

Matrice compagnon :

En posant $b_i = a_i / a_n$, on a

$$Q(x) = x^n + b_{n-1}x^{n-1} + \ldots + b_2 x^2 + b_1 x + b_0$$

est le polynôme caractéristique de

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & -b_0 \\ 1 & 0 & 0 & \cdots & 0 & -b_1 \\ 0 & 1 & 0 & \cdots & 0 & -b_2 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & -b_{n-2} \\ 0 & 0 & 0 & \cdots & 1 & -b_{n-1} \end{bmatrix}$$

Racines de polynômes de degré n

Trouver les racines de P, qui sont les valeurs propres de $\mathbf{C}$

méthode récursive

Algorithm 2: Racines du polynôme par matrice compagnon

for $i \leftarrow 1$ **to** k **do**
  résoudre $\mathbf{QR} = \mathbf{C}$
  $\mathbf{C} \leftarrow \mathbf{RQ}$ (renversement du produit)
end
return diag($\mathbf{C}$)

Rappel : $\mathbf{R}$ est une matrice triangulaire supérieure et $\mathbf{Q}$ une matrice de rotation.

Fonctionne aussi avec la décomposition LU, mais est instable numériquement

Racines de polynômes de degré n

En cas de racines multiples,
la convergence est moins bonne
(répulsion des racines multiples)

$\downarrow$

recherche de zéros avec la méthode de Newton

Racines de polynômes de degré n

Résumé :

- écrire la matrice compagnon $\mathbf{C}$ de P
- trouver les valeurs propres de $\mathbf{C}$ (décomp. RQ)
- optimisation non-linéaire (Newton)