Appunti di Analisi Numerica*

Giuseppe Profiti

30 ottobre 2006

1 Esempi di curve di Bezier per caratteri

- Differenza tra carattere vettoriale e bitmap in un file PDF
- caratteri pfb in fontforge, modifica delle curve, esempio di riempimento con xccurv
- trasformazione da bitmap a vettoriale con xccurv
- importazione di file postscript in xccurv (il file contiene un'immagine)

2 Interpolazione polinomiale

Date $n+1$ osservazioni su $n+1$ punti distinti, cioé $(x_i, y_i)_{i=0,\dots,n}$, cerco una $\varphi^* \in \phi$ (con $\phi \equiv P_n$) tale che $\varphi^*(x_i) = y_i \; i = 0, \dots, n$

$$\varphi^*(x) = \sum_{i=0}^{n} a_i^* \cdot \varphi_i(x)$$

Sono richieste

- esistenza
- unicità
- buona ricostruzione

φ^* dipende dai punti e dalla base. Cerchiamo di provare l'esistenza nella base $1, x, x^2, \ldots, x^n$.

Una volta dimostrate l'esistenza e unicità con una base, vale per tutto lo spazio.

Teorema 1 *Dati $n+1$ punti $(x_i, y_i)_{i=0,\ldots,n}$ con x_i distinti, $\exists! p(x) \in P_n$ che verifica le condizioni $p(x_i) = y_i$ per $i = 0, \ldots, n$*

Dimostrazione:

$$\begin{array}{ll} i=0 & a_0 + a_1 x_0 + a_2 x_0^2 + \ldots + a_n x_0^n = y_0 \\ i=1 & a_1 + a_1 x_1 + a_2 x_1^2 + \ldots + a_n x_1^n = y_1 \\ \vdots & \\ i=n & a_n + a_1 x_n + a_2 x_n^2 + \ldots + a_n x_n^n = y_n \end{array}$$

Formano un sistema di $n+1$ equazioni lineari nelle incognite $a_0, a_1, \ldots, a_n$ il problema di interpolazione è quindi equivalente a trovare una soluzione del sistema.

$$\underbrace{\begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \ldots & x_n^n \end{pmatrix}}_{V_{(n+1)\times(n+1)}} \underbrace{\begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix}}_{\bar{a}} = \underbrace{\begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}}_{\bar{y}}$$

Il sistema ammette una e una sola soluzione se V non è singolare ($det(V) \neq 0$). V è una matrice di Vandermonde per cui

$$det(V) = \prod_{\substack{i,j=0 \\ j>i}}^{n} (x_j - x_i)$$

è sempre diverso da zero perché nessun fattore sarà zero ($x_i \neq x_i \; \forall j$)

Risolvendo il sistema trovo $p(x)$, ma la matrice di vandermonde è mal condizionata.

2.1 Interpolazione nella base di Bernstein

$$(x_i, y_i)_{i=0,\ldots,n} \quad x_i \text{ distinti}$$

$$p(x_i) = y_i \text{ con } p(x) = \sum_{i=0}^{n} b_i \, B_{i,n}(x)$$

$$\begin{pmatrix} B_{0,n}(x_0) & B_{1,n}(x_0) & \dots & B_{n,n}(x_0) \\ B_{0,n}(x_1) & B_{1,n}(x_1) & \dots & B_{n,n}(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ B_{0,n}(x_n) & B_{1,n}(x_n) & \dots & B_{n,n}(x_n) \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}$$

è un sistema molto meglio condizionato del precedente. Generalmente abbiamo $x_i \in [a, b]$ (con $a = x_0$ e $b = x_n$ come intervallo minimo), ma possiamo fare il cambio di variabile per tornare in $[0, 1]$: $x \in [a, b] \to t \in [0, 1]$. Le matrici con x e con t sono uguali, però con t semplifichiamo (a livello computazionale).

2.2 Interpolazione polinomiale nella base di Newton

La base di Newton è una base locale, costituita da

$$1, (x-x_0), (x-x_0)(x-x_1), (x-x_0)(x-x_1)(x-x_2), \dots, (x-x_0)\cdots(x-x_{n-1})$$

$$p(x) = d_0 + d_1\,(x - x_0) + d_2\,(x - x_0)(x - x_1) + \ldots + d_n\,(x - x_0)\cdots(x - x_{n-1})$$

$$\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & (x_1 - x_0) & 0 & \cdots & 0 \\ 1 & (x_2 - x_0) & (x_2 - x_0)(x_2 - x_1) & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \\ 1 & (x_n - x_0) & (x_n - x_0)(x_n - x_1) & \cdots & (x_n - x_0)\cdots(x_n - x_{n-1}) \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ \vdots \\ \vdots \\ d_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ \vdots \\ y_n \end{pmatrix}$$

(gli altri termini si annullano)

È una matrice triangolare (bassa), ha lo stesso determinante della matrice di Vanbdermonde

$$\begin{aligned} d_0 &= y_0 \\ d_0 + (x_1 - x_0)d_1 &= y_1 \\ &\dots \end{aligned}$$

2.2.1 Implementazione

```
per k=0,n
    d(k)=y(k)
per i=1,n
    per k=n,i
        d(k)=(d(k)-d(k-1))/(x(k)-x(k-i))
```

2.2.2 Valutazione dei polinomi di Newton

Si applica lo stesso metodo di Horner, si valuta dall'interno verso l'esterno:

$$p(x) = d_0 + (x - x_0)[d_1 + (x - x_1)(d_2 + (x - x_2)(d_3 + \ldots)]$$

2.3 Interpolazione nella base di Lagrange

$$L_{0,n}(x)\ L_{1,n}(x)\ \ldots L_{n,n}(x)$$

$$p(x) = \sum_{i=0}^{n} ? \cdot L_{i,n}(x)$$

vogliamo abbattere il costo computazionale.

$$p(x) = \sum_{i=0}^{n} y_i \cdot L_{i,n}(x)$$

$$L_{i,n}(x) = \begin{cases} 1 & \text{se } i = j \\ 0 & \text{se } i \neq j \end{cases}$$

2.3.1 Costruzione di $L_{i,n}(x)$

È un polinomio con radici $x_0, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$, grazie al teorema fondamentale dell'algebra abbiamo che:

$$L_{i,n} = k(x - x_0)(x - x_1) \cdots (x - x_{i-1}(x - x_{i+1}) \cdots (x - x_n)$$

Per trovare k (che indica l'ampiezza) sfrutto l'uguaglianza con 1 nei punti x_i

$$\begin{aligned} L_{i,n}(x_i) = 1 &= k(x_i - x_0)(x_i - x_1) \cdots (x_i - x_{i-1})(x_i - x_{i+1}) \cdots (x_i - x_n) \\ k &= \frac{1}{(x_i - x_0)(x_i - x_1) \cdots (x_i - x_{i-1})(x_i - x_{i+1}) \cdots (x_i - x_n)} \end{aligned}$$

$$
\begin{aligned}
L_{i,n} &= \frac{(x-x_0)(x-x_1)\cdots(x-x_{i-1})(x-x_{i+1})\cdots(x-x_n)}{(x_i-x_0)(x_i-x_1)\cdots(x_i-x_{i-1})(x_i-x_{i+1})\cdots(x_i-x_n)} \\
&= \prod_{\substack{j=0 \\ j\neq i}}^{n} (x-x_j) / \prod_{\substack{j=0 \\ j\neq i}}^{n} (x_i-x_j)
\end{aligned}
$$

Nota: con Lagrange e Newton posso interpolare qualsiasi siano gli y_i una volta trovata la base.

Esempio/esercizio

Dati gli x {0,1,3}

1. trovare i polinomi di lagrange su x_0, x_1, x_2 e rappresentarli graficamente
2. dati $y_0 = 0$ $y_1 = 1$ $y_2 = 2$ esplicitare il polinomio interpolante nella base di Lagrange