# Interpolarea funcţiilor.

*– Metode de interpolare pe porţiuni. –*

Prof.dr.ing. Gabriela Ciuprina

Universitatea "Politehnica" Bucureşti, Facultatea de Inginerie Electrică

Suport didactic pentru disciplina *Algoritmi Numerici*, 2017-2018



Notes



## Cuprins





Notes

# Preliminarii

## Scrierea formală a unei probleme

$$\mathbf{y} = f(\mathbf{x}), \tag{1}$$

$\mathbf{x}$ - datele problemei (parametri independenţi);
$\mathbf{y}$ - mărimile de interes ce se doresc a fi estimate.

De exemplu, $f$ poate reprezenta:

- un proces de măsurare a mărimilor $\mathbf{y}$ pentru o anumită stare complet caracterizată de $\mathbf{x}$;
- un program software complicat, capabil să analizeze configuraţia caracterizată complet de datele $\mathbf{x}$ şi să calculeze printr-un algoritm de postprocesare mărimile $\mathbf{y}$.



Notes



# Preliminarii

## Formularea problemei (neriguros)

Se dă o funcţie reprezentată prin date:
$(\mathbf{x}_k, \mathbf{y}_k)$, $k = 0, \ldots, n$, unde $\mathbf{y}_k = f(\mathbf{x}_k)$.
Se doreşte găsirea unei expresii analitice pentru o funcţie $g$ care să aproximeze aceste date adică
$g(\mathbf{x}_k) \approx \mathbf{y}_k$ sau chiar $g(\mathbf{x}_k) = \mathbf{y}_k$.

- Interpolare setului de date: $g$ trece <u>prin</u> punctele mulţimii de date: $g(\mathbf{x}_k) = \mathbf{y}_k$
- Aproximarea (regresia) setului de date = $g$ trece <u>printre</u> punctele mulţimii de date: $g(\mathbf{x}_k) \approx \mathbf{y}_k$



Notes

# Preliminarii

**Observaţii:**

1. $\mathbf{x}_k$ se numeşte şi reţea (grid) de discretizare.
2. Interpolarea/aproximarea este utilă şi dacă funcţia este reprezentată prin cod = există un software capabil să calculeze $f(\mathbf{x})$ pentru orice $\mathbf{x}$ dorit, dacă efortul de evaluare al lui $f$ este mare.

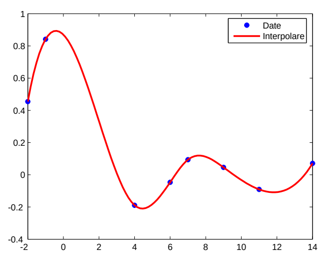


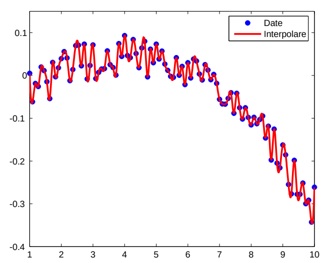
Notes



# Preliminarii

Exemple: interpolare



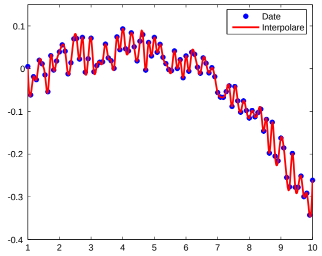

Interpolarea unui set de date. În cazul în care setul de date are foarte multe valori, interpolarea poate genera oscilaţii nedorite.

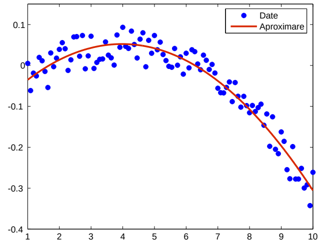


Notes

# Preliminarii

Exemple: interpolare vs. aproximare




Avantajul aproximării: se diminuează erorile de măsurare din rezultatul final.



Notes



# Precizări $f :? \to ?$

- Cazul scalar unidimensional (1D): $f, g : [a, b] \to \mathbb{R}$.
- Cazul vectorial unidimensional $f : [a, b] \to \mathbb{R}^m, \quad m > 1$ se reduce la $m$ interpolări/aproximări 1D.
- Cazul scalar bidimensional (2D) $f, g : [a, b] \times [c, d] \to \mathbb{R}$
- Cazul scalar $n$-dimensional ($n$D) $f, g : \mathcal{D} \subset \mathbb{R}^n \to \mathbb{R}$.
- Cazul cel mai general $f, g : \mathcal{D} \subset \mathbb{R}^n \to \mathbb{R}^m$ se reduce la $m$ situaţii de tip $n$D.

În cele ce urmează vom pp. cazul 1D.



Notes

## Distanţa dintre două funcţii

Se doreşte ca $g : [a, b] \to \mathbb{R}$ să aproximeze/interpoleze cât mai bine funcţia $f : [a, b] \to \mathbb{R}$.

$\Leftrightarrow$

distanţa dintre cele două funcţii

$$d(f, g) = \|f - g\| \tag{2}$$

să fie cât mai mică.

Există mai multe procedee de definire a normei.



Notes



## Distanţa dintre două funcţii

Procedee de definire a normei.

- Aria dintre graficele celor două funcţii

$$d_1(f, g) = \frac{1}{b-a} \int_a^b |f(x) - g(x)| \, \mathrm{d}x. \tag{3}$$

Dezavantaj: local, pot exista diferenţe foarte mari între $f$ sî $g$.

- Abaterea medie pătratică

$$d_2(f, g) = \sqrt{\frac{1}{b-a} \int_a^b (f(x) - g(x))^2 \, \mathrm{d}x}. \tag{4}$$

Acelaşi dezavantaj.



Notes

## Distanţa dintre două funcţii

Procedee de definire a normei.

- Abaterea maximă dintre cele două funcţii

$$d_3(f,g) = \max_{x \in [a,b]} |f(x) - g(x)|. \tag{5}$$

Din pdv al acurateţii - este cea mai avantajoasă.

OBS: Niciuna din aceste norme nu se poate evalua.



Notes



## Distanţa dintre două funcţii

Normele discrete:

$$d_{1d}(f,g) = \sum_{k=0}^{n} |g(x_k) - f(x_k)|, \tag{6}$$

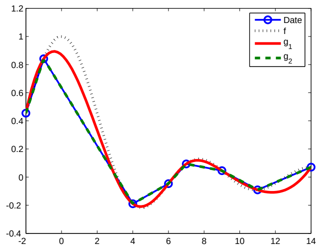
$$d_{2d}(f,g) = \sqrt{\sum_{k=0}^{n} (g(x_k) - f(x_k))^2}, \tag{7}$$

$$d_{3d}(f,g) = \max_{k=0,n} |g(x_k) - f(x_k)|. \tag{8}$$



Notes

## Distanţa dintre două funcţii




Avantaj: pot fi evaluate cu uşurintă.
Dezavantaj: se pierde posibilitatea evaluării acurateţii între noduri. Mai mult $d_d(f, g_1) = 0$; $d_d(f, g_2) = 0$; $\Rightarrow$ problemă prost formulată.



Notes



## Formularea problemei interpolării

Se caută $g$ pentru care $d_d(f, g) = 0$, unde $f$ este cunoscută într-un număr finit de puncte $f(x_j) = y_j$.
Echivalent cu a impune condiţiile de interpolare

$$g(x_j) = f(x_j), \quad j = 0, \ldots, n, \tag{9}$$

$\Leftrightarrow$

$$g(x_j) = y_j, \quad j = 0, \ldots, n. \tag{10}$$

Pentru a face ca problema să fie bine formulată matematic (soluţia să existe şi să fie unică) funcţia $g$ se caută în spaţiul polinoamelor generalizate
$\Leftrightarrow$
$g$ adică se caută de forma unei combinaţii liniare de $m$ funcţii $\varphi_k$, $k = 1, \ldots, m$ numite funcţii de bază:

$$g(x) = \sum_{k=0}^{m} c_k \varphi_k(x). \tag{11}$$



Notes

## Formularea problemei interpolării

Funcţiile de bază se aleg înainte de rezolvarea propriu-zisă a problemei interpolării. Exemple:

- $\varphi_0(x) = 1$, $\varphi_1(x) = \sin x$, $\varphi_2(x) = \cos x$, $\varphi_3(x) = \sin(2x)$, etc.
- $\varphi_0(x) = 1$, $\varphi_1(x) = x$, $\varphi_2(x) = x^2$, $\varphi_3(x) = x^3$, etc.

Cei $m$ coeficienţi $c_k$ se calculează din impunerea condiţiilor de interpolare:

$$\sum_{k=0}^{m} c_k \varphi_k(x_j) = y_j, \quad j = 0, \ldots, n, \tag{12}$$

$\Rightarrow$ Sistem algebric liniar cu $n+1$ ecuaţii şi $m+1$ necunoscute.



Notes



## Formularea problemei interpolării

Pentru buna formulare matematică se impune ca $m = n$ şi

$$\Delta = \begin{vmatrix} \varphi_0(x_0) & \varphi_1(x_0) & \cdots & \varphi_n(x_0) \\ \varphi_0(x_1) & \varphi_1(x_1) & \cdots & \varphi_n(x_1) \\ \cdots & & & \\ \varphi_0(x_n) & \varphi_1(x_n) & \cdots & \varphi_n(x_n) \end{vmatrix} \neq 0. \tag{13}$$

$\Delta \neq 0 \Leftrightarrow x_j$ sunt distincte şi $\varphi_k$ sunt liniar independente.



Notes

## Formularea problemei interpolării

**Date:**

- un tabel de valori $(x_k, y_k)$, $k = 0, \ldots, n$, unde punctele reţelei de discretizare $x_k$ sunt distincte două câte două;
- $n + 1$ funcţii de bază liniar independente $\varphi_k(x)$, $k = 0, \ldots, n$.

**Se cer:**

- coeficienţii $c_k$, $k = 0, \ldots, n$ pentru care sunt satisfăcute condiţiile de intepolare $g(x_j) = y_j$, $j = 0, \ldots, n$ unde $g(x) = \sum_{k=0}^{n} c_k \varphi_k(x)$ este polinomul de interpolare al datelor din tabel.



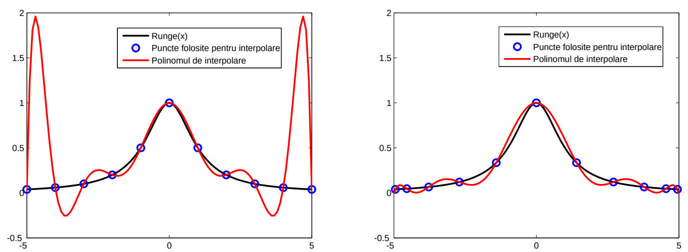
Notes



## Interpolarea globală - concluzii

- Funcţiile de bază $\varphi_k(x)$ au o singură expresie pe tot domeniul de definiţie $\Rightarrow$
  $\Rightarrow$ polinomul de interpolare $g(x)$ are o singură expresie pe tot domeniul de definiţie.

**Dezavantaj:** efectul Runge - oscilaţii la capete.
**Remediu:** Interpolarea Cebyshev






Notes

## Interpolarea globală - concluzii

- Efectul Runge poate fi evitat dacă nodurile gridului de discretizare se aleg în conformitate cu rădăcinile polinoamelor Cebyshev.

**Dezavantaj:** funcţia trebuie să fie definită prin cod, dar acest lucru nu este posibil întotdeauna.
**Remediu: Interpolarea pe porţiuni.**

- Funcţiile de bază $\varphi_k(x)$ sunt "funcţii cu acoladă", au expresii diferite pe intervalele care discretizează domeniul de definiţie $\Rightarrow$
  $\Rightarrow$ polinomul de interpolare $g(x)$ este o "funcţie cu acoladă".



Notes



## Interpolarea liniară pe porţiuni

$\varphi_k(x)$ sunt polinoame Lagrange $l_k(x)$ definite pe porţiuni.
$l_k(x_k) = 1$
$l_k(x_j) = 0$ pentru $j \neq k$

$$l_0(x) = \begin{cases} \frac{x-x_1}{x_0-x_1} & \text{dacă} \quad x \in [x_0, x_1] \\ 0 & \text{dacă} \quad x \in (x_1, x_n] \end{cases}$$

$$l_k(x) = \begin{cases} \frac{x-x_{k-1}}{x_k-x_{k-1}} & \text{dacă} \quad x \in [x_{k-1}, x_k) \\ \frac{x-x_{k+1}}{x_k-x_{k+1}} & \text{dacă} \quad x \in [x_k, x_{k+1}] \\ 0 & \text{dacă} \quad x \in [x_1, x_{k-1}) \cup (x_{k+1}, x_n] \end{cases} \quad k = 2, n-1$$

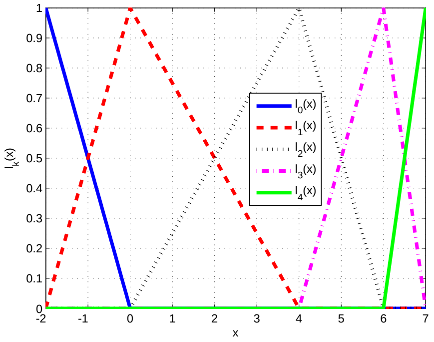
$$l_n(x) = \begin{cases} \frac{x-x_{n-1}}{x_n-x_{n-1}} & \text{dacă} \quad x \in [x_{n-1}, x_n] \\ 0 & \text{dacă} \quad x \in [x_0, x_{n-1}) \end{cases}$$

$$g(x) = \sum_{k=0}^{n} c_k l_k(x) \tag{14}$$

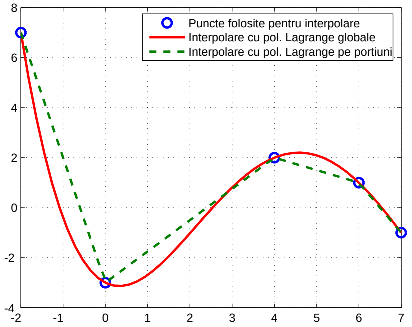


Notes

## Interpolarea liniară pe porţiuni




Polinoame Lagrange pe porţiuni.




Polinoame de interpolare.



Notes



## Interpolarea liniară pe porţiuni

<u>`funcţie`</u> interpolare_lpp($n, x, y, xcrt$)
; evaluează polinomul de interpolare liniară pe porţiuni în *xcrt*

; declaraţii - parametri de intrare
<u>`întreg`</u> $n$ ; dimensiunea problemei - nr. de intervale
<u>`tablou`</u> <u>`real`</u> $x[n], y[n]$ ; tabelul de valori, indici de la zero
<u>`real`</u> *xcrt* ; punctul de evaluat

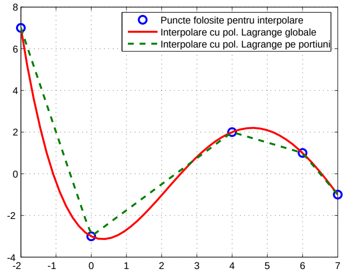
$k$ = cauta($n, x, xcrt$)

<u>`întoarce`</u> $(y_{k+1} - y_k)/(x_{k+1} - x_k) * (xcrt - x_k) + y_k$



Notes

## Interpolarea liniară pe porţiuni




**Dezavantaj:**
funcţia de interpolare nu este derivabilă în noduri.
**Remediu:**
creşterea gradului polinoamelor care se folosesc pe porţiuni.



Notes



## Interpolarea Hermite

- interpolarea unei funcţii pe o reţea de noduri $x_k$ în care se cunosc valorile funcţiei $y_k$ şi valorile derivatelor acesteia $y'_k$.

| $x$ | $x_0$ | $x_1$ | $\cdots$ | $x_k$ | $\cdots$ | $x_n$ |
|---|---|---|---|---|---|---|
| $y$ | $y_0$ | $y_1$ | $\cdots$ | $y_k$ | $\cdots$ | $y_n$ |
| $y'$ | $y'_0$ | $y'_1$ | $\cdots$ | $y'_k$ | $\cdots$ | $y'_n$ |

**Condiţii de interpolare:** $\forall\ k = 0, \ldots, n-1$ :

$$\begin{cases} g(x_k) = y_k, \\ g(x_{k+1}) = y_{k+1}, \\ g'(x_k) = y'_k, \\ g'(x_{k+1}) = y'_{k+1}. \end{cases} \quad (15)$$

$\Rightarrow$ polinom de interpolare de gradul 3 pe fiecare subinterval.



Notes

## Interpolarea Hermite

$$\begin{aligned} g(x) &= c_{0k} + c_{1k}(x-x_k) + c_{2k}(x-x_k)^2 + c_{3k}(x-x_k)^3, (16) \\ g'(x) &= c_{1k} + 2c_{2k}(x-x_k) + 3c_{3k}(x-x_k)^2, \quad x \in [x_k, x_{k+1}]. \end{aligned}$$

$$\begin{cases} g(x_k) = y_k, \\ g(x_{k+1}) = y_{k+1}, \\ g'(x_k) = y'_k, \\ g'(x_{k+1}) = y'_{k+1}. \end{cases} \tag{17}$$

$\Rightarrow$

$$\begin{cases} c_{0k} = y_k, \\ c_{0k} + c_{1k}(x_{k+1}-x_k) + c_{2k}(x_{k+1}-x_k)^2 + c_{3k}(x_{k+1}-x_k)^3 = y_{k+1}, \\ c_{1k} = y'_k, \\ c_{1k} + 2c_{2k}(x_{k+1}-x_k) + 3c_{3k}(x_{k+1}-x_k)^2 = y'_{k+1}, \end{cases}$$



Notes



## Interpolarea Hermite

$$\begin{cases} c_{0k} = y_k, \\ c_{1k} = y'_k, \\ c_{2k} = \left[3(y_{k+1}-y_k) - (x_{k+1}-x_k)(2y'_k + y'_{k+1})\right]/(x_{k+1}-x_k)^2, \\ c_{3k} = \left[(y'_{k+1}+y'_k) - 2(2y_{k+1}-y_k)/(x_{k+1}-x_k)\right]/(x_{k+1}-x_k)^2. \end{cases} \tag{18}$$

**Dificultate:**
de cele mai multe ori nu se cunosc informaţii despre $y'_k$.
**Soluţie:**
Evaluarea derivatelor se poate face numeric (interpolare Bessel):

$$y'_0 = \frac{y_1 - y_0}{x_1 - x_0} \quad y'_n = \frac{y_n - y_{n-1}}{x_n - x_{n-1}},$$

$$y'_k = \frac{\beta^2 y_{k+1} + (\alpha^2 - \beta^2)y_k - \alpha^2 y_{k-1}}{\alpha\beta(\alpha+\beta)h},$$

$x_{k+1} - x_k = \alpha h$, $x_k - x_{k-1} = \beta h$. Nenatural (ordinul interpolării la formulele de derivare nr.).



Notes

# Interpolarea Hermite

Cel mai avantajos: evaluarea derivatelor din impunerea unor condiţii de continuitate suplimentare pentru derivata a doua a polinomului de interpolare în nodurile reţelei de interpolare $\Rightarrow$ **Interpolare spline cubică clasică** impune

$$g''(x_k - 0) = g''(x_k + 0), \quad k = 1, \ldots, n-1. \tag{19}$$

Deoarece

$$g''(x) = 2c_{2k} + 6c_{3k}(x - x_k), \quad x \in [x_k, x_{k+1}]. \tag{20}$$

$\Rightarrow$

$$2c_{2,k-1} + 6c_{3,k-1}(x_k - x_{k-1}) = 2c_{2k}, \quad k = 1, \ldots, n-1. \tag{21}$$

$\Rightarrow$



Notes



# Interpolarea Hermite

$$\frac{1}{x_k - x_{k-1}} y'_{k-1} + 2\left(\frac{1}{x_k - x_{k-1}} + \frac{1}{x_{k+1} - x_k}\right) y'_k + \frac{1}{x_{k+1} - x_k} y'_{k+1} =$$
$$= 3\frac{y_k - y_{k-1}}{(x_k - x_{k-1})^2} + 3\frac{y_{k+1} - y_k}{(x_{k+1} - x_k)^2}. \tag{22}$$

$n-1$ relaţii şi $n+1$ necunoscute.
Pentru unicitate se impun încă două condiţii la capete.



Notes

## Interpolarea Hermite

Pentru unicitate se impun încă două condiţii la capete.
Variante:

- **Condiţii forţate la capete:** se impun $y'_0$ şi $y'_n$ ca la interpolarea Bessel
- **Condiţii naturale la capete:**

$$g''(x_0) = 0, \quad g''(x_n) = 0.$$

$\Rightarrow$

$$2y'_0 + y'_1 = 3\frac{y_1 - y_0}{x_1 - x_0}, \qquad y'_{n-1} + 2y'_n = 3\frac{y_n - y_{n-1}}{x_n - x_{n-1}}. \quad (23)$$



Notes



## Interpolarea Hermite

Sistemul de rezolvat pentru evaluarea derivatelor - tridiagonal:

$$2y'_0 + y'_1 = 3\frac{y_1 - y_0}{x_1 - x_0},$$

$$\frac{1}{x_k - x_{k-1}}y'_{k-1} + 2\left(\frac{1}{x_k - x_{k-1}} + \frac{1}{x_{k+1} - x_k}\right)y'_k + \frac{1}{x_{k+1} - x_k}y'_{k+1} =$$

$$= 3\frac{y_k - y_{k-1}}{(x_k - x_{k-1})^2} + 3\frac{y_{k+1} - y_k}{(x_{k+1} - x_k)^2},$$

$$y'_{n-1} + 2y'_n = 3\frac{y_n - y_{n-1}}{x_n - x_{n-1}}.$$



Notes

## Interpolarea Hermite

**Avantajul interpolării spline cubice clasice:**
minimizează curbura pătratică medie a polinomului de intepolare

$$\int_a^b \left(g''(x)\right)^2 \,\mathrm{d}x$$

$\Rightarrow$ nu apar oscilaţii nedorite între punctele reţelei de interpolare



Notes



## Interpolarea Hermite

Algoritmul interpolării spline cubice:

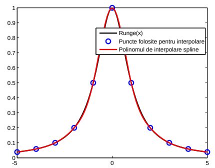
<u>funcţie</u> pregăteşte_spline($n, x, y, yder$)
; calculează derivatele funcţiei în nodurile reţelei de discretizare
; declaraţii - parametri de intrare
<u>întreg</u> $n$ ; dimensiunea problemei - numărul de intervale
<u>tablou</u> <u>real</u> $x[n]$, $y[n]$ ; tabelul de valori, indici de la zero
<u>tablou</u> <u>real</u> $yder[n]$ ; parametri de ieşire
<u>tablou</u> <u>real</u> $p[n]$, $q[n]$, $r[n]$ ; matricea tridiagonală asamblată
; asamblează matricea tridiagonală
$q_0 = 2$
$r_0 = 1$
$b_0 = 3(y_1 - y_0)/(x_1 - x_0)$
<u>pentru</u> $k = 1, n-1$
    $p_k = 1/(x_k - x_{k-1})$
    $q_k = 2/(x_k - x_{k-1}) + 2/(x_{k+1} - x_k)$
    $r_k = 1/(x_{k+1} - x_k)$
    $b_k = 3(y_k - y_{k-1})/(x_k - x_{k-1})^2 + 3(y_{k+1} - y_k)/(x_{k+1} - x_k)^2$
•
$p_n = 1$
$q_n = 2$
$b_n = 3(y_n - y_{n-1})/(x_n - x_{n-1})$
Gauss_tridiag($n + 1, p, q, r, b, yder$)



Notes

# Interpolarea Hermite

funcţie aproximare_spline($n, x, y, yder, xcrt$)
; evaluează polinomul de aproximare spline în punctul $xcrt$
; declaraţii - parametri de intrare
întreg $n$ ; dimensiunea problemei - numărul de intervale
tablou real $x[n], y[n], yder[n]$ tabelul de valori, indici de la zero
real $xcrt$ ; punctul de evaluat
$k$ = cauta($n, x, xcrt$)
$c0k = y_k$
$c1k = yder_k$
$h = x_{k+1} - x_k$
$c2k = (3(y_{k+1} - y_k) - h(2 * yder_k + yder_{k+1}))/h^2$
$c3k = (yder_{k+1} + yder_k - 2(y_{k+1} - y_k)/h)/h^2$
$hcrt = xcrt - x_k$
întoarce $c0k + c1k * hcrt + c2k * hcrt^2 + c3k * hcrt^3$




Interpolarea *spline* a funcţiei Runge pe o reţea uniformă de puncte.



Notes



# Eşantionare adaptivă pentru interpolarea funcţiilor date prin cod

- Cod foarte mare consumatoare de resurse de calcul $\Rightarrow$ se poate genera un tabel de valori
- Pentru estimări ulterioare ale funcţiei: se evaluează rapid interpolarea sau aproximarea tabelului de valori.

Acurateţea polinomului de interpolare/aproximare depinde de numărul şi relevanţa punctelor din tabelul de valori.



Notes

## Interpolarea funcţiilor date prin cod

- **Se dă** codul unei funcţii vectoriale de o variabilă reală $\mathbf{F}(x)$, unde $x \in [a, b]$.
- **Se cere** să se găsească o mulţime minimă de puncte $\mathcal{S} = \{x_k\}$, $k = 1, \ldots, n$, astfel încât eroarea relativă calculată pentru o mulţime de puncte de test $\mathcal{S}' = \{x'_k\}$, $k = 1, \ldots, n'$, intercalate printre punctele mulţimii $\mathcal{S}$ să fie mai mică sau egală decât o valoare impusă:

$$\varepsilon \leq \varepsilon_{\mathrm{impus}}, \tag{24}$$

unde

$$\varepsilon = \max_{k=1,n'} \left\| \frac{\mathbf{F}(x'_k) - \mathbf{G}(x'_k)}{\mathbf{F}(x'_k)} \right\|. \tag{25}$$



Notes



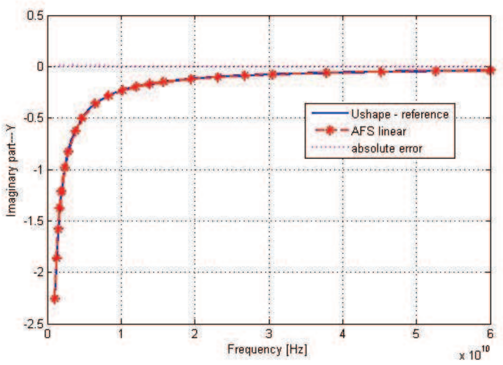
## Interpolarea funcţiilor date prin cod

; Calculează o interpolare inteligentă a unei funcţii definită prin cod
0. Alege un set iniţial $\mathcal{S}$ de eşantioane
`repetă`
    1. Alege un set de puncte de test $\mathcal{S}'$
    2. Calculează o interpolare **G** folosind setul de eşantioane $\mathcal{S}$
    3. Evaluează **G** în setul de puncte de test $\mathcal{S}'$
    4. Calculează eroarea cu (25)
    5. Mută punctele de test în setul de eşantioane $\mathcal{S} = \mathcal{S} \cup \mathcal{S}'$
`până când` este îndeplinită condiţia (24)
6. Calculează interpolarea finală **G** folosind setul de eşantioane $\mathcal{S}$



Notes

# Interpolarea funcţiilor date prin cod




Eşantionare adaptivă folosind interpolarea liniară pe porţiuni a tabelului de valori $\{x, f(x)\}, x \in \mathcal{S}$. Setul final are 23 puncte pentru o eroare impusă de 10%.

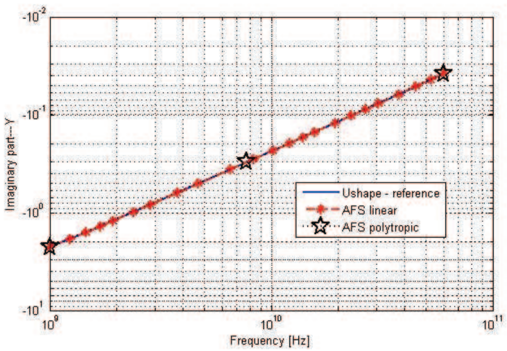
Referinţa - curba albastră, obţinută prin evaluarea funcţiei în 1000 puncte echidistante.



Notes



# Interpolarea funcţiilor date prin cod



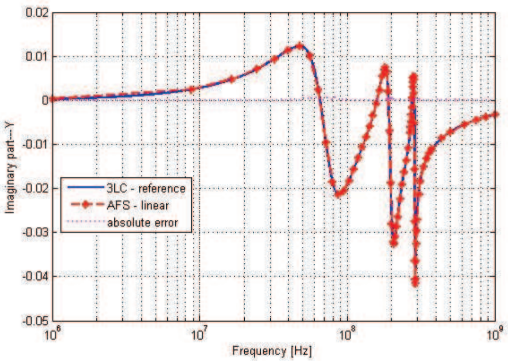

Eşantionare adaptivă folosind interpolarea politropică a tabelului de valori $\{\log(x), log(f(x))\}, x \in \mathcal{S}$. Setul final are 3 puncte pentru o eroare impusă de 10%.

Referinţa - curba albastră, obţinută prin evaluarea funcţiei în 1000 puncte echidistante.



Notes

## Interpolarea funcţiilor date prin cod




Eşantionare adaptivă folosind interpolarea liniară pe porţiuni a tabelului de valori $\{x, f(x)\}, x \in \mathcal{S}$. Setul final are 75 puncte pentru o eroare impusă de 10%.

Referinţa - curba albastră, obţinută prin evaluarea funcţiei în 1000 puncte echidistante.



Notes



## Referinţe


- [Ciuprina13a] Gabriela Ciuprina - Algoritmi numerici pentru calcule ştiinţifice în ingineria electrică , Editura MatrixROM, 2013, pag. 162-168

  disponibilă la http://www.lmn.pub.ro/∼gabriela/books/AlgNr_MatrixRom2013.pdf
- [Cheney04] Ward Cheney and David Kincaid, *Numerical Mathematics and Computing*, Brooks/Cole publishing Company,2000. (9.3 *Interpolation by B Splines*)
- [deBoor] *B(asic)-Spline Basics*

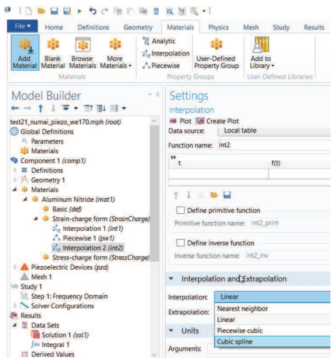
  disponibilă la http://ftp.cs.wisc.edu/Approx/bsplbasic.pdf




Notes

# Mesajul principal

Fiţi atenţi la astfel de informaţii (capturi din COMSOL).
Consultaţi documentaţia pentru detalii!






Notes

Notes