Calcul Numeric - Examen - 2 Iunie 2025 - Profir Alexandru - 235

3) Fie $(\pi_k)_{k \in N}$ polinoamele ortogonale Legendere monice.

(a) Arătați că polinoamele

$$\pi_k^-(t^2) = \frac{\pi_{2k+1}(t)}{t}$$

sunt ortogonale monice pe $[0, 1]$ în raport cu ponderea $w(t) = \frac{1}{\sqrt{t}}$.

Polinoamele Legendre $\pi_m(x)$ sunt ortogonale pe intervalul $[-1, 1]$ în raport cu funcția pondere $w(x) = 1$, asta însemnând că:

$$\int_{-1}^{1} \pi_k(x)\pi_l(x)\mathrm{d}x = 0, \text{pentru } k \neq l$$

Paritatea polinoamelor Legendre $\pi_m = \frac{m!}{(2,)!} \cdot \frac{d^t}{\mathrm{d}t^m}(t^2 - 1)^m$ $(\text{forma rezultată din Formula lui Rodrigues})$ depinde de termenul **m**. Demonstrăm acest lucru prin următoarele:

- Fie $g(t) = (t^2 - 1)^m$;
- $g(-t) = [(-t)^2 - 1]^m = (t^2 - 1)^m = g(t)$;
- derivarea de **m** ori a unei funcții pare rezultă tot într-o funcție pară, dacă **m - par**, și într-o funcție impară, dacă **m - impar**.

Rezultă astfel că paritatea lui $\frac{\mathrm{d}^m}{\mathrm{d}t^m}(t^2 - 1)^m$ depinde de paritatea lui **m**:

- **m - par**: $(t^2 - 1)^m$ - par, $\frac{\mathrm{d}^m}{\mathrm{d}t^m}(t^2 - 1)^m$ - par, π_m - multiplu scalar al acestei derivate, rezultă că $\pi_m - \text{par}$;
- **m - impar**: $(t^2 - 1)^m$ - par, $\frac{\mathrm{d}^m}{\mathrm{d}t^m}(t^2 - 1)^m$ - impar, π_m - multiplu scalar al acestei derivate, rezultă că $\pi_m - \text{impar}$;

Rezultă astfel paritatea $\pi_m = \frac{m!}{(2m)!} \cdot \frac{\mathrm{d}^m}{\mathrm{d}t^m}(t^2 - 1)^m - \begin{cases} \text{par} & m - \text{par} \\ \text{impar} & m - \text{impar} \end{cases}$. Ne interesează ortogonalitatea polinoamelor de grad par, astfel considerăm integrala ortogonalității pentru polinoamele $\pi_{2k+1}(x), \pi_{2l+1}(x), \text{pentru } k \neq l$:

$$\int_{-1}^{1} \pi_{2k+1}(x)\pi_{2l+1}(x)\mathrm{d}x = 0 \qquad (*)$$

$\forall\, k \in N, \forall\, l \in N : \pi_{2k+1}(x) - \text{impar}, \pi_{2l+1}(x) - \text{impar} \Rightarrow \pi_{2k+1}(x) \cdot \pi_{2l+1}(x) - \text{par}$

Având interval simetric, și cum $\pi_{2k+1}(x)\cdot\pi_{2l+1}(x)$ este par pentru $\forall\, k\in N, \forall\, l\in N$, integrala $(*)$ pe intervalul simetric $[-1,1]$ rezultă:

$$\int_{-1}^{1}\pi_{2k+1}(x)\pi_{2l+1}(x)\mathrm{dx}=2\cdot\int_{0}^{1}\pi_{2k+1}(x)\pi_{2l+1}(x)\mathrm{dx}=0\Rightarrow\int_{0}^{1}\pi_{2k+1}(x)\pi_{2l+1}(x)\mathrm{dx}=0\quad(**)$$

Monicitatea polinoamelor π_k^- rezultă din următoarele:

1. dacă $\pi_{2k+1}(t)$ este un polinom Legendre monic de grad 2k+1, termenul sau principal este t^{2k+1};
2. rezultă atunci că $\pi_k^-(x)=\dfrac{\pi_{2k+1}(\sqrt{x})}{\sqrt{x}}=\dfrac{x^{k+\frac{1}{2}}+c_{2k}x^{k-\frac{1}{2}}+\ldots}{x^{\frac{1}{2}}}=x^k+c_{2k}x^{k-1}+\ldots$

Deoarece $\pi_{2k+1}(t)$ este un polinom impar, el conține doar puteri impare ale lui t, astfel:

$$\pi_k^-(x)=\frac{\pi_{2k+1}(\sqrt{x})}{\sqrt{x}}=\frac{(\sqrt{x})^{2k+1}+a_{2k-1}(\sqrt{x})^{2k-1}+\ldots+a_1\sqrt{x}}{\sqrt{x}}=(\sqrt{x})^{2k}+a_{2k-1}(\sqrt{x})^{2k-2}+\ldots+a_1=x^k+a_{2k-1}x^{k-1}+\ldots+a_1$$

1. coeficientul acestui termen x^k este 1, deci $\pi_k^-(x)$ este un polinom monic de grad k. (1)

Dorim să demonstrăm că polinoamele $\pi_k^-(t^2)=\dfrac{\pi_{2k+1}(t)}{t}$ sunt ortogonale în raport cu ponderea $w(t)=\dfrac{1}{\sqrt{t}}$. Facem o schimbare de variabilă $x=t^2\Rightarrow t=\sqrt{x}$, iar derivând obținem $\mathrm{dt}=\dfrac{1}{2\sqrt{x}}\mathrm{dx}$. Substituind în integrala $(**)$, obținem:

$$0=\int_0^1\pi_{2k+1}(x)\pi_{2l+1}(x)\cdot\sqrt{x}\,\mathrm{dx}=2\cdot\int_0^1\pi_k^-(x^2)\pi_l^-(x^2)\cdot x\,\mathrm{dx}=2\cdot\int_0^1\frac{1}{t}\pi_k^-(t^2)\pi_l^-(t^2)2t\mathrm{dt}=2\cdot\int_0^1\pi_k^-(t^2)\pi_l^-(t^2)\mathrm{dt},k\neq l\quad(2)$$

Din (1) si (2): rezultă astfel că polinoamele $\pi_k^-(t^2)=\dfrac{\pi_{2k+1}(t)}{t}$ sunt ortogonale monice pe [0, 1] în raport cu ponderea $w(t)=\sqrt{t}$.

(b) Stabiliți formula de cuadratură

$$\int_0^1\sqrt{x}\,f(x)\mathrm{dx}=2\sum_{k=1}^{n}A_kt_k^2f\left(t_k^2\right)+R_n(f)$$

unde A_k și t_k, $k=1,\ldots,2n$ sunt coeficienții și respectiv nodurile formulei de cuadratură Gauss-Legendre cu 2n+1 noduri.

Folosim schimbarea de variabilă $x=t^2\Rightarrow\mathrm{dx}=2\mathrm{tdt}$, rezultând astfel:

$\int_0^1\sqrt{x}\,f(x)\mathrm{dx}=\int_0^1\sqrt{t^2}\,f(t^2)2\mathrm{tdt}=\int_0^1t\cdot f(t^2)\cdot2\mathrm{tdt}=2\int_0^1t^2f(t^2)\mathrm{dt}$. Fie $g(t)=t^2f(t^2)$. Rezulta integrala $2\int_0^1g(t)\mathrm{dt}$. Pentru a aplica Gauss-Legendre, avem nevoie de o noua schimbare de variabila pentru a ajunge in $[-1,1]$. Folosim schimbarea de variabila $t=\frac{u+1}{2}, u\in[-1,1] \Rightarrow \mathrm{dt}=\frac{1}{2}\mathrm{du}$. Rezulta integrala: $2\int_0^1 g(t)\mathrm{dt}=2\int_{-1}^1 g\left(\frac{u+1}{2}\right)\frac{1}{2}\mathrm{du}=\int_{-1}^1 g\left(\frac{u+1}{2}\right)\mathrm{du}=\int_{-1}^1\left(\frac{u+1}{2}\right)^2 f\left(\left(\frac{u+1}{2}\right)^2\right)\mathrm{du}$. Folosind formula Gauss-Legendre cu 2n+1 noduri, rezultă:

$\int_0^1\sqrt{x}\,f(x)\mathrm{dx}=\int_{-1}^1\left(\frac{u+1}{2}\right)^2 f\left(\left(\frac{u+1}{2}\right)^2\right)\mathrm{du}=\sum_{k=1}^{2n+1}A_k\left(\frac{t_k+1}{2}\right)^2 f\left(\left(\frac{t_k+1}{2}\right)^2\right)+R_{2n+1}(f)$. Ținând cont de simetria formulei Gauss-Legendre, avem $t_k=-t_{n-k}$ și $A_k=A_{n-k},\ k=1,\ldots,2n$, rezultă $t_k^2=t_{n-k}^2$ și $\sum_{k=1}^{2n}A_k f\left(t_k^2\right)=2\sum_{k=1}^{n}A_k f\left(t_k^2\right)$.

Pentru a afla restul, dacă $f\in C^{4n}[-1,1], \exists\,\xi\in(-1,1)$ astfel încât:

$$R_n(f)=\frac{f^{(4n)}(\xi)}{(4n)!}\int_{-1}^{1}[\pi_{2n}(t)]^2\mathrm{dt}$$

Rezultă formula de cuadratură:

$$\int_0^1\frac{1}{\sqrt{(x)}}f(x)\mathrm{dx}\ =\int_{-1}^{1}f(x^2)\mathrm{dx}=2\sum_{k=1}^{n}A_k f\left(x_k^2\right)+\frac{f^{(4n)}(\xi)}{(4n)!}\cdot\int_{-1}^{1}[\pi_{2n}(x)]^2\mathrm{dx}, \xi\in(-1,1)$$

(c) Implementarea mai jos și (d) Implementați (în MATLAB) o formulă de cuadratură de tip Gauss pentru integrala $\int_0^1\frac{\sin(x)}{\sqrt{x}}\mathrm{dx}$ cu 8 zecimale exacte.

```
% Functia
f = @(x) sin(x);
% Integrala exacta
I_exact = integral(@(x) (1 ./ sqrt(x)) .* f(x), 0, 1);

% Numarul initial de noduri
n = 2;
% Numarul maxim de iteratii
max_iter = 100;
% Toleranta admisa
tol = 1e-8;

for iter = 1:max_iter
    [g_nodes, g_coeffs] = gauss_rsqrt(n);
    I_approx = sum(g_coeffs(:) .* f(g_nodes(:)));

    if norm(I_approx - I_exact, "inf") < tol
        fprintf("[nodes = %d] Toleranta %g atinsa in %d iteratii!", n, tol, iter);
        fprintf("Exact: %.16g | Approx: %.16g", I_exact, I_approx);
        break;
    end

    n = n + 1;
end
```

```
if iter >= max_iter
    error("Nu s-a atins convergenta in %d maximum iteratii!", max_iter);
end
```

2) Dorim să calculăm $\frac{1}{\sqrt{a}}$, pentru $a > 0$.

(a) Pornind de la o ecuație convenabilă și folosind metoda lui Newton, deduceți o metodă pentru calculul lui $\frac{1}{\sqrt{a}}$.

Fie $x = \frac{1}{\sqrt{a}} \Rightarrow x^2 = \frac{1}{a} \Leftrightarrow \frac{1}{x^2} = a \Leftrightarrow \frac{1}{x^2} - a = 0$. Rezultă formula de recurență:

$$f(x) = \frac{1}{x^2} - a,$$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = \frac{1}{2}x_n\left(3 - \mathrm{ax}_n^2\right)$$

(b) Pentru ce valori ale lui x_0 metoda converge?

Pentru a analiza convergența, considerăm funcția de iterație $\phi(x) = \frac{x}{2}(3 - \mathrm{ax}^2)$, căutăm rădăcina $\alpha = \frac{1}{\sqrt{(a)}}$.

Derivata lui $\phi(x)$ este:

$$\phi'(x) = \left[\frac{x}{2}(3 - \mathrm{ax}^2)\right]' = \frac{1}{2}(3 - \mathrm{ax}^2) + \frac{x}{2}(-2\mathrm{ax}) = \frac{3}{2} - \frac{\mathrm{ax}^2}{2} - 2\mathrm{ax}^2 = \frac{3}{2} - \frac{3}{2}\mathrm{ax}^2$$

Ținând cont că x > 0, iar pentru convergența locală este necesar ca $|\phi'(x)| < 1$:

$\phi'(\alpha) = \phi'\left(\frac{1}{\sqrt{a}}\right) = \frac{3}{2} - \frac{3}{2}a\left(\frac{1}{\sqrt{a}}\right)^2 = \frac{3}{2} - \frac{3}{2} = 0$. Rezultă astfel soluția *falsă* $x_0 = 0$.

Pentru a obține un interval mai larg de convergență, trebuie să ne asigurăm că iterațiile rămân pozitive și că acestea se apropie de rădăcină. Ținând cont că $x_0 > 0$:

Iterația x_{k+1} trebuie să rămână pozitivă, astfel este necesar ca $3 - \mathrm{ax}_k^2 > 0 \Leftrightarrow \mathrm{ax}_k^2 < 3 \Leftrightarrow {x_k}^2 < \frac{3}{a} \Rightarrow x_k < \sqrt{\frac{3}{a}}$. Deci dacă $0 < x_0 < \sqrt{\frac{3}{a}} \Rightarrow x_1 > 0$.

Din condiția $|\phi'(x)| < 1$ pentru convergența monotonă:

$|\frac{3}{2} - \frac{3}{2}\mathrm{ax}^2| < 1 \Leftrightarrow -1 < \frac{3}{2} - \frac{3}{2}\mathrm{ax}^2 < 1$, rezolvăm pentru ambele inegalități:

- $\frac{3}{2} - \frac{3}{2}\mathrm{ax}^2 < 1 \Leftrightarrow \frac{1}{2} < \frac{3}{2}\mathrm{ax}^2 \Leftrightarrow \frac{1}{3} < \mathrm{ax}^2 \Leftrightarrow x^2 > \frac{1}{3a} \Leftrightarrow x > \frac{1}{\sqrt{3a}}$

- $\frac{3}{2} - \frac{3}{2}\mathrm{ax}^2 > -1 \Leftrightarrow \frac{5}{2} > \frac{3}{2}\mathrm{ax}^2 \Leftrightarrow \frac{5}{3} > \mathrm{ax}^2 \Leftrightarrow x^2 < \frac{5}{3a} \Leftrightarrow x < \sqrt{\frac{5}{3a}}$

Rezultă astfel că pentru convergență monotonă în sensul teoremei de punct fix a lui Banach, $x_0 \in \left(\frac{\sqrt{3}}{3\sqrt{a}}, \frac{\sqrt{15}}{3\sqrt{a}}\right)$, iar rădăcina $\alpha = \frac{1}{\sqrt{a}}$aparține acestui interval. Dacă impunem și ca $\phi(x) > 0$, pentru $0 < x_0 < \frac{1}{\sqrt{a}}$ convergența este monotonă, adică șirul iterațiilor este crescător și mărginit superior de soluție.

Însă pentru $x_0 > \frac{1}{\sqrt{a}}$, este necesar ca $\phi(x_0) > 0$, obținându-se astfel că $x_0 < \sqrt{\frac{3}{a}}$, convergența fiind asigurată dacă x_0 este *suficient de aproape* de $\alpha = \frac{1}{\sqrt{a}}$.

(c) Folosiți iterația de la (a) pentru a da o metodă de calcul al radicalului fără împărțiri.

Iterația $x_{k+1} = \frac{x_k}{2}\left(3 - \mathrm{ax}_k^2\right)$ aproximează $x \approx \frac{1}{\sqrt{a}}$. Această formulă nu implică nicio operație de împărțire, împărțirea cu 2 fiind o înmulțire cu 0.5, care dpdv. hardware este trivială sau chiar un shift binar.

Odată aproximat $x_N \approx \frac{1}{\sqrt{a}}$, putem calcula $\sqrt{a}$folosind: $\sqrt{a} = a \cdot \frac{1}{\sqrt{a}} \approx a \cdot x_N$.

```
% Valoarea lui a
a = 3;
real_sqrt = sqrt(a);

% Toleranta
tol = 1e-8;
% Numarul maxim de iteratii
max_iter = 100;

% x0 initial: 0 < x0 < sqrt(3/a)
xk = 1e-1;

for iter = 1:max_iter
    xk1 = xk * (3-a*xk^2) * 0.5;
    approx_sqrt = a * xk1;
    if abs(approx_sqrt - real_sqrt) < tol
        fprintf("Toleranta %g a fost atinsa in %d iteratii!", tol, iter);
        fprintf("Exact: %.16g | Approx: %.16g", real_sqrt, approx_sqrt);
        break;
    end

    xk = xk1;
end

if iter >= max_iter
    error("Nu s-a atins convergenta in %d maximum iteratii!", max_iter);
end
```

```
function [g_nodes, g_coeffs] = gauss_rsqrt(n)
    %% GAUSS_RSQRT  - determina coeficientii si nodurile pentru o cuadratura de tip Gauss-Legen
    %                 cu ponderea w(t) = t^(-1/2)
    %   Inputs:
    %
    %   - n     - numarul de noduri;
    %
    %   Outputs:
    %
    %   - g_nodes       - nodurile cuadraturii;
    %   - g_coeffs       - coeficientii cuadraturii;
    %%

    [nodes, coeffs] = gauss_legendre(2*n);
    g_nodes = nodes(1:n) .^ 2;
    g_coeffs = 2 * coeffs(1:n);
end

function [g_nodes, g_coeffs] = gauss_legendre(n)
    %% GAUSS_LEGENDRE - determina coeficientii si nodurile pentru o cuadratura de tip Gauss-Leg
    %
    %   Inputs:
    %
    %   - n     - numarul de noduri;
    %
    %   Outputs:
    %
    %   - g_nodes       - nodurile cuadraturii;
    %   - g_coeffs       - coeficientii cuadraturii;
    %%

    alpha = zeros(n, 1);
    beta = (1:n-1).^2 ./ (4*(1:n-1).^2 - 1);
    beta = [2; beta(:)];

    [g_nodes, g_coeffs] = gauss_quad(alpha', beta');
end

function [g_nodes, g_coeff] = gauss_quad(alpha, beta)
    %% GAUSS_QUAD - generare cuadratura Gauss
    %               calculeaza noduri si coeficienti pentru cuadraturi
    %               Gauss, cu alpha si beta cunoscuti, folosing matricea
    %               Jacobi
    %
    %   Inputs:
    %
    %   - alpha, beta   - coeficientii cunoscuti pentru matricea Jacobi
    %
    %   Outputs:
    %
    %   - g_nodes       - nodurile cuadraturii Gauss;
    %   - g_coeff       - coeficientii cuadraturii Gauss;
```

```
    % Numarul de noduri
    n = length(alpha);

    % Radacinile elementelor Beta ale termenilor sub/supra-diagonali
    rb = sqrt(beta(2:n));

    % Matricea Jacobi - matrice tridiagonala
    J = diag(alpha) + diag(rb, -1) + diag(rb, 1);

    % Diagonalizarea lui J - d va contine valorile_proprii/nodurile
    [v, d] = eig(J);

    % Nodurile cuadraturii
    g_nodes = diag(d);

    % Coeficientii cuadraturii - patratul primei componente din vectorul
    % propriu
    g_coeff = beta(1) * v(1, :) .^ 2;
end
```