Numerical Analysis

Derivation of Gauss-Quadrature Formula using Legendre Polynomials

목 차

Introduction to Legendre Polynomials

Legendre polynomials: Eigen functions of the singular Sturm-Lioville problem

N-th Order $$\frac{d}{d\tau}\left\{(1-\tau^2)\frac{dL_N(\tau)}{d\tau}\right\}+N(N+1)L_N(\tau)=0 \qquad L_N(1)=1$$

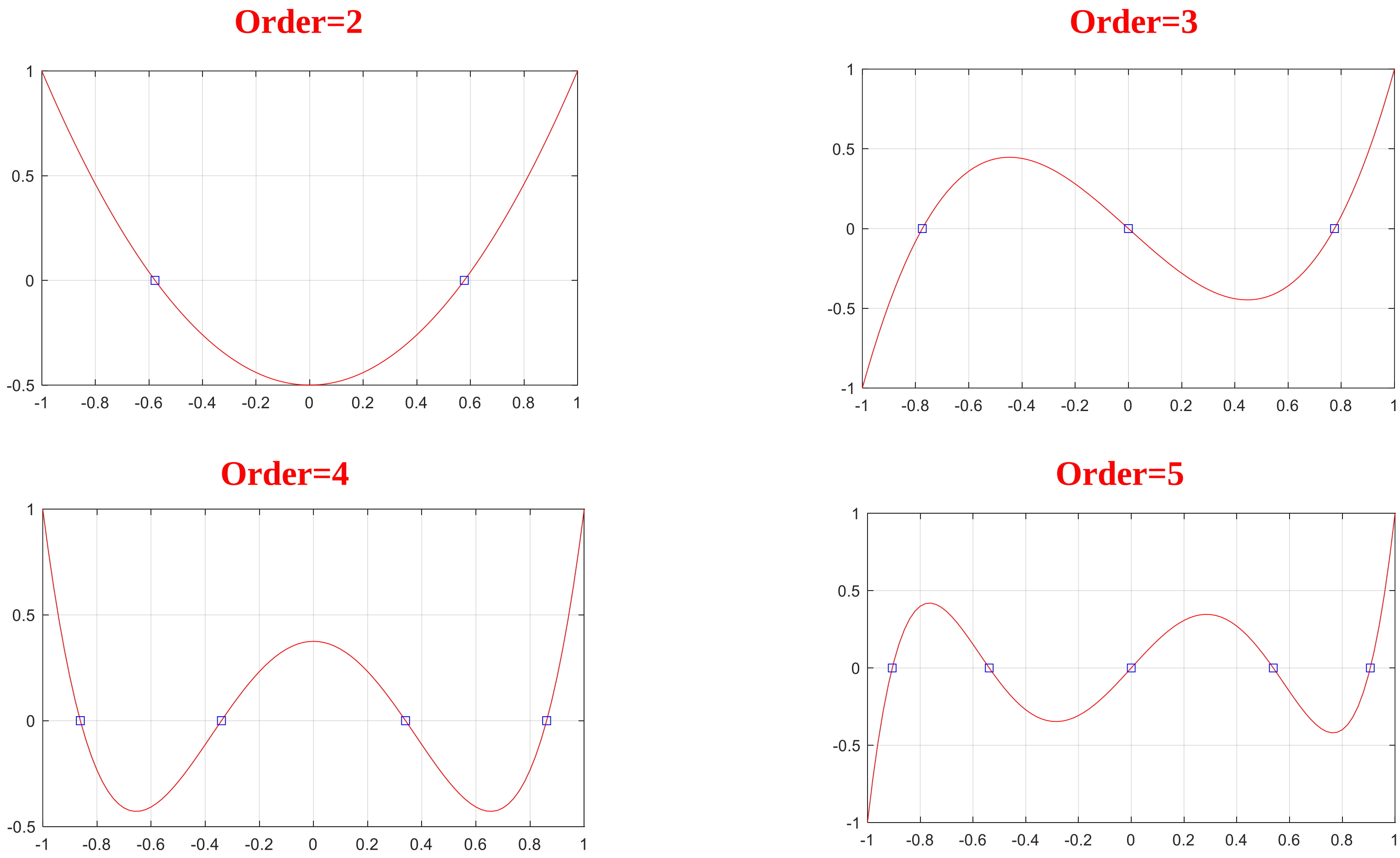
Derivation of Legendre polynomials using Recursive Relations

$$\boxed{\begin{aligned} &L_0(\tau)=1\\ &L_1(\tau)=\tau\\ &(N+1)L_{N+1}(\tau)=(2N+1)\tau L_N(\tau)-NL_{N-1}(\tau)\\ &\qquad\qquad\qquad\qquad (N=1,2,3,\cdots)\end{aligned}}$$

$$\begin{aligned}
L_0(\tau)&=1\\
L_1(\tau)&=\tau\\
L_2(\tau)&=\frac{3}{2}\tau L_1(\tau)-\frac{1}{2}L_0(\tau)=\frac{1}{2}(3\tau^2-1)\\
L_3(\tau)&=\frac{5}{3}\tau L_2(\tau)-\frac{2}{3}L_1(\tau)=\frac{5}{6}(3\tau^3-\tau)-\frac{2}{3}\tau=\frac{1}{2}(5\tau^3-3\tau)\\
L_4(\tau)&=\frac{7}{4}\tau L_3(\tau)-\frac{3}{4}L_2(\tau)=\frac{7}{8}(5\tau^4-3\tau^2)-\frac{3}{8}(3\tau^2-1)=\frac{1}{8}(35\tau^4-30\tau^2+3)\\
L_5(\tau)&=\frac{9}{5}\tau L_4(\tau)-\frac{4}{5}L_3(\tau)=\frac{9}{40}(35\tau^5-30\tau^3+3\tau)-\frac{2}{5}(5\tau^3-3\tau)=\frac{1}{8}(63\tau^5-70\tau^3+15\tau)\\
L_6(\tau)&=\frac{11}{6}\tau L_5(\tau)-\frac{5}{6}L_4(\tau)=\frac{11}{48}(63\tau^6-70\tau^4+15\tau^2)-\frac{5}{48}(35\tau^4-30\tau^2+3)\\
&=\frac{1}{16}(231\tau^6-315\tau^4+105\tau^2-5)\\
&\vdots
\end{aligned}$$

Introduction to Legendre Polynomials

Shapes of Legendre polynomials: Program=A2_Legendre_Polynomial_Root_Plot.m



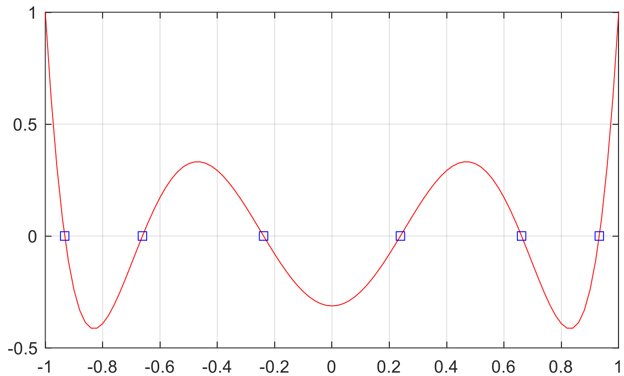
Orthogonal Polynomials Meeting the orthogonality condition of

$$\boxed{(L_j, L_k)_{\varpi} = \int_{-1}^{1} L_j(\tau) L_k(\tau) d\tau = \frac{2}{2j+1}\delta_{jk}}$$

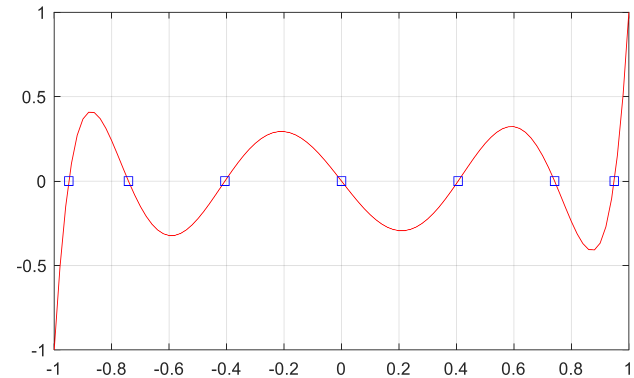
Introduction to Legendre Polynomials

Shapes of Legendre polynomials: Program=A2_Legendre_Polynomial_Root_Plot.m

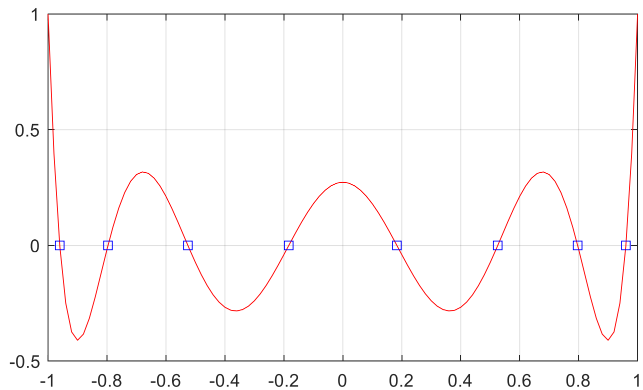
Order=6



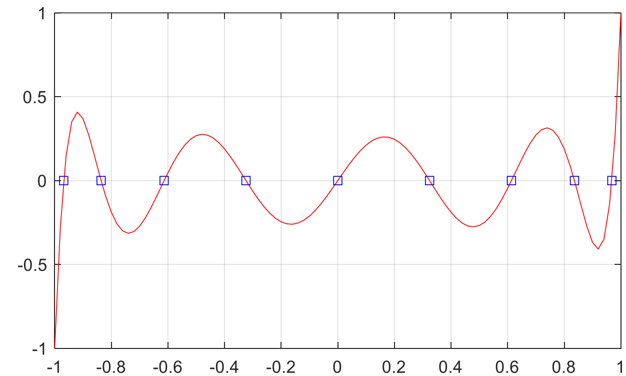
Order=7



Order=8



Order=9



$$\boxed{(L_j, L_k)_{\varpi} = \int_{-1}^{1} L_j(\tau) L_k(\tau) d\tau = \frac{2}{2j+1}\delta_{jk}}$$

The roots of Legendre Polynomials corresponds to the Quadrature Nodes for the Gauss Integration Formula

Introduction to Legendre Polynomials

Useful Recursive Formula to Compute the Gauss Quadrature Nodes

$$(N+1)L_{N+1}(\tau) = (2N+1)\tau L_N(\tau) - NL_{N-1}(\tau), \quad \text{with } L_0(\tau)=1, L_1(\tau)=\tau$$

$$(2N+1)L_N(\tau) = L'_{N+1}(\tau) - L'_{N-1}(\tau), \quad N \geq 1$$

$$(1-\tau^2)L'_N(\tau) = \frac{N(N+1)}{2N+1}\left(L_{N-1}(\tau) - L_{N+1}(\tau)\right)$$

Computational tips to Compute the Gauss Quadrature Nodes: L2_Legendre_Poly_Root_one.m

(i) There exists only one root of the (k+1)-th order Legendre polynomial between two adjacent roots of the k-th order polynomial.

(ii) Thus, the average of such tow roots may become a good approximation for one root of the (k+1)-th order Legendre polynomial and can be used initial solution of the root.

(iii) The rest one root is less than the smallest root of the the k-th order polynomial and another one root is greater than the biggest root of the the k-th order polynomial.

(iv) The Newton-Raphson method can be adopted using the recursive estimation of the gradient of the (k+1)-th order Legendre polynomial. Only a few iterations are required to get a fully converged solution for each root.

$$\boxed{\begin{aligned}&(2N+1)L_N(\tau) = L'_{N+1}(\tau) - L'_{N-1}(\tau), \quad N \geq 1\\&(1-\tau^2)L'_N(\tau) = \frac{N(N+1)}{2N+1}\left(L_{N-1}(\tau) - L_{N+1}(\tau)\right)\end{aligned}}$$

목 차

Lagrange Polynomial Interpolation Using Gauss Quadrature Nodes

Gauss Quadrature Nodes $\left\{\tau_j\right\}_{j=0}^{j=N} \leftarrow \tau_j = 2\dfrac{x_j - x_0}{x_N - x_0} - 1 \in [-1,1]$

Functions at Gauss Quadrature Nodes

$$\left\{ f_j \right\}_{j=0}^{j=N} \leftarrow f_j = f_j(x_j) \leftarrow x_j = \frac{x_N - x_0}{2}\tau_j + \frac{x_N + x_0}{2}$$

Data set for Lagrange Interpolation $\left\{(\tau_j, f_j)\right\}_{j=0}^{j=N}$

Lagrange Interpolation using Gauss Quadrature Nodes

$$f(\tau) = \sum_{j=0}^{j=N} \phi_j(\tau) f_j, \leftarrow \phi_j(\tau) = \prod_{\substack{k=0 \\ k \neq j}}^{k=N} \frac{(\tau - \tau_k)}{(\tau_j - \tau_k)}$$

$$\tau = 2\frac{x - x_0}{x_N - x_0} - 1 \in [-1,1]$$

$$d\tau = \frac{2}{x_N - x_0} dx$$

Gauss Quadrature Weights (1)

Integration Formula with $f(\tau)=\sum_{j=0}^{j=N}\phi_j(\tau)f_j, \leftarrow \phi_j(\tau)=\prod_{\substack{k=0\\k\neq j}}^{k=N}\frac{(\tau-\tau_k)}{(\tau_j-\tau_k)}$

$$\tau = 2\frac{x-x_0}{x_N-x_0}-1\in[-1,1]$$

$$d\tau=\frac{2}{x_N-x_0}dx$$

$$I=\int_{x_0}^{x_N} f(x)dx=\frac{x_N-x_0}{2}\int_{-1}^{1} f(\tau)d\tau=\frac{x_N-x_0}{2}\sum_{j=0}^{j=N} w_j f_j$$

Gauss Quadrature Weights

$$\int_{-1}^{1} f(\tau)d\tau=\int_{-1}^{1}\sum_{j=0}^{j=N}\phi_j(\tau)f_j d\tau=\sum_{j=0}^{j=N}\left(\int_{-1}^{1}\phi_j(\tau)d\tau\right)f_j=\sum_{j=0}^{j=N}w_j f_j$$

$$\boxed{w_j=\int_{-1}^{1}\phi_j(\tau)d\tau}$$

Gauss Quadrature Weights (2)

Weight Computation

$$\boxed{w_j = \int_{-1}^{1} \phi_j(\tau)d\tau \leftarrow \phi_j(\tau) = \prod_{\substack{k=0 \\ k\neq j}}^{k=N} \frac{(\tau - \tau_k)}{(\tau_j - \tau_k)}}$$

Expansion of Lagrange Polynomial

$$\phi_j(\tau) = \prod_{\substack{k=0 \\ k\neq j}}^{k=N} \frac{(\tau - \tau_k)}{(\tau_j - \tau_k)} = \alpha_{N-1}\tau^{N-1} + \alpha_{N-2}\tau^{N-2} + \cdots + \alpha_2\tau^2 + \alpha_1\tau + \alpha_0$$

$$w_j = \int_{-1}^{1} \phi_j(\tau)d\tau = \left(\frac{\alpha_{N-1}}{N}\tau^N + \frac{\alpha_{N-1}}{N-1}\tau^{N-1} + \cdots + \frac{\alpha_2}{3}\tau^3 + \frac{\alpha_1}{2}\tau^2 + \alpha_0\tau \right)\Bigg|_{\tau=-1}^{\tau=1}$$

Computational Tip for Polynomial Expansion

$$p(\tau) = a_{k-1}\tau^{k-1} + \cdots + a_2\tau^2 + a_1\tau + a_0$$

$$q(\tau) = p(\tau)\frac{(\tau - \tau_k)}{(\tau_j - \tau_k)} = p(\tau)(b_1\tau + b_0) \leftarrow b_1 = \frac{1}{(\tau_j - \tau_k)}, b_1 = \frac{(\tau - \tau_k)}{(\tau_j - \tau_k)}$$

목 차

Computer Model to Compute Quadrature Nodes and Weights (1)

Program Structure

```
%---------------------------------------------------------------------------
% Legendre polynomial and its derivative
%---------------------------------------------------------------------------
function [tau_vec,Weight_vec] = B1_Legedre_Gauss_Quadrature(N)
%---------------------------------------------------------------------------
%   (1) Quadrature Nodes
%---------------------------------------------------------------------------
   [tau_vec] = L2_Legendre_Poly_Roots(N) ;
%---------------------------------------------------------------------------
%   (2) Quadrature Weights
%---------------------------------------------------------------------------
%   (2-1) Polynomial Expansion of Lagrange Polymonials
%---------------------------------------------------------------------------
   [c_mat] = L5_Lagrange_Poly_Expansion(N,tau_vec) ;
%---------------------------------------------------------------------------
%   (2-2) Integration of Lagrange Polymonials
%---------------------------------------------------------------------------
   for J=1:N
       cvec(1:N) = c_mat(1:N,J) ;
%
       [cintg_vec] = L6_Poly_Intg_Coef(N-1,cvec) ; % (N)-th order
%
       taup =  1.0 ;
       [funp] = L6_Poly_Computing(N,cintg_vec,taup) ;
%
       taum = -1.0 ;
       [funm] = L6_Poly_Computing(N,cintg_vec,taum) ;
%
       Weight_vec(J) = funp - funm ;
   end
%---------------------------------------------------------------------------
end
%---------------------------------------------------------------------------
```

Quadrature Nodes

Polynomial Expansion

Integration of Lagrange Polynomials

Quadrature Weights

Computer Model to Compute Quadrature Nodes and Weights (1)

Applications

N=2;   >> N=2;[tau_vec,Weight_vec] = B1_Legedre_Gauss_Quadrature(N)

```
tau_vec      =  -0.5774    0.5774
Weight_vec =      1            1
```

N=3 ; >> N=3;[tau_vec,Weight_vec] = B1_Legedre_Gauss_Quadrature(N)

```
tau_vec      =  -0.7746         0       0.7746
Weight_vec =     0.5556    0.8889    0.5556
```

N=4 ; >> N=4;[tau_vec,Weight_vec] = B1_Legedre_Gauss_Quadrature(N)

```
tau_vec      =  -0.8611   -0.3400    0.3400    0.8611
Weight_vec =    0.3479    0.6521    0.6521    0.3479
```

N=5 ; >> N=5;[tau_vec,Weight_vec] = B1_Legedre_Gauss_Quadrature(N)

```
tau_vec      =  -0.9062   -0.5385         0       0.5385    0.9062
Weight_vec =    0.2369    0.4786    0.5689    0.4786    0.2369
```

N=6 ; >> N=6;[tau_vec,Weight_vec] = B1_Legedre_Gauss_Quadrature(N)

```
tau_vec      =  -0.9325   -0.6612   -0.2386    0.2386    0.6612    0.9325
Weight_vec =    0.1713    0.3608    0.4679    0.4679    0.3608    0.1713
```

End of Lecture