Numerical Methods for Computational Science and Engineering

Fall Semester 2017 (HS17)

Prof. Rima Alaifari, SAM, ETH Zurich

Implementation of Chebychev interpolation cont'd:

(2) Computation of coefficients α_j in

$$p(t) = \sum_{j=0}^{n} \alpha_j T_j(t)$$

Interpolation conditions:

$$p(t_k) = f(t_k) = y_k \qquad k = 0,\dots,n$$

$$t_k = \cos\left(\frac{2k+1}{2(n+1)}\pi\right)$$

$$s_k := \frac{2k+1}{4(n+1)} \qquad t_k = \cos(2\pi s_k)$$

Cheb. interp. $\qquad$ Def T_j

$$q(s) := p(\cos 2\pi s) = \sum_{j=0}^{n} \alpha_j T_j(\cos 2\pi s) \overset{\text{Def. 6.1.76}}{=} \sum_{j=0}^{n} \alpha_j \cos(2\pi j s)$$

$$= \sum_{j=0}^{n} \tfrac{1}{2}\alpha_j (\exp(2\pi\imath j s) + \exp(-2\pi\imath j s)) \quad [\text{by } \cos z = \tfrac{1}{2}(e^z + e^{-z})]$$

$$= \sum_{j=-n}^{n+1} \beta_j \exp(-2\pi\imath j s), \quad \text{with} \quad \beta_j := \begin{cases} 0 & \text{, for } j = n+1, \\ \frac{1}{2}\alpha_j & \text{, for } j = 1,\dots,n, \\ \alpha_0 & \text{, for } j = 0, \\ \frac{1}{2}\alpha_{-j} & \text{, for } j = -n,\dots,-1. \end{cases} \tag{6.1.108}$$

$$T_j(t) = \cos(j \cdot \arccos(t))$$

$$T_j(\cos(2\pi s)) = \cos(j \cdot 2\pi s) \quad \leftarrow \quad s \in [0, \tfrac{1}{2}] \,!$$

Symmetry of q due to cosine:

$$q(s) = q(1-s)$$

$$\Downarrow \leftarrow (6.1.109)$$

$$q\left(1 - \frac{2k+1}{4(n+1)}\right) = y_k, \quad k = 0,\dots,n.$$

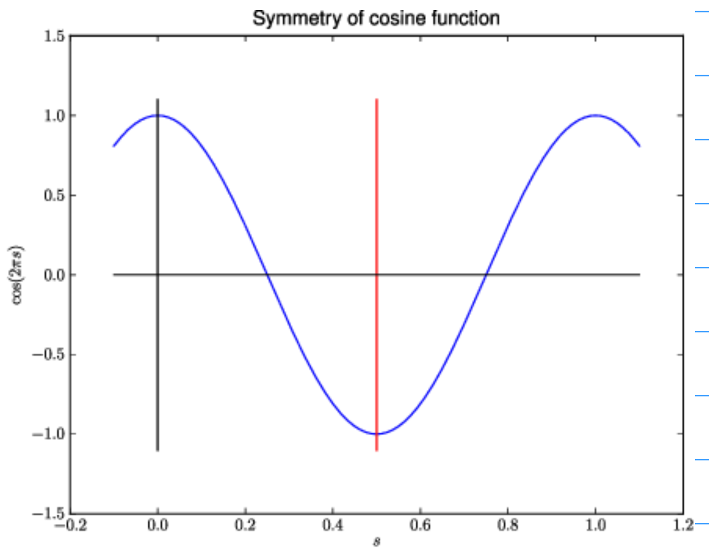


Fig. 231

⇒ Extend I.C.:

$$q\left(\frac{k}{2(n+1)}+\frac{1}{4(n+1)}\right) \underline{\underline{=}} z_k := \begin{cases} y_k & \text{, for } k=0,\dots,n\,, \\ y_{2n+1-k} & \text{, for } k=n+1,\dots,2n+1\,. \end{cases} \qquad (6.1.110)$$

because for $k = n+1, \dots, 2n+1$:

$$q(s_k) = q(1-s_k) = q(s_{2n+1-k}) = y_{2n+1-k}$$

Now: Find coefficients β_j using FFT:

LSE for β_j:

$$q\left(\frac{k}{2(n+1)}+\frac{1}{4(n+1)}\right) = z_k \qquad k=0,\dots,2n+1$$

Use $q(s) = \sum_{j=-n}^{n+1} \beta_j \exp(-2\pi i j s)$:

$$\sum_{j=-n}^{n+1} \beta_j \exp\left[-2\pi i j\left(\frac{k}{2(n+1)}+\frac{1}{4(n+1)}\right)\right] = z_k$$

$$\sum_{j=-n}^{n+1} \beta_j \exp\left(-\frac{\pi i j k}{n+1}\right) \exp\left(-\frac{\pi i j}{2(n+1)}\right) = z_k$$

$$\sum_{j=0}^{2n+1} \beta_{j-n} \exp\left(-\frac{\pi i (j-n)k}{n+1}\right) \exp\left(-\frac{\pi i (j-n)}{2(n+1)}\right) = z_k$$

$$\left[\sum_{j=0}^{2n+1} \beta_{j-n} \exp\left(-\frac{\pi i (j-n)}{2(n+1)}\right) \underbrace{\exp\left(-\frac{\pi i j k}{n+1}\right)}_{= \exp\left(-2\pi i \frac{jk}{2(n+1)}\right) = \omega_{2(n+1)}^{jk}}\right] \exp\left(\frac{\pi i n k}{n+1}\right) = z_k$$

$$\sum_{j=0}^{2n+1} \beta_{j-n} \exp\left(-\frac{\pi i (j-n)}{2(n+1)}\right) \omega_{2(n+1)}^{jk} = \exp\left(-\frac{\pi i n k}{n+1}\right) z_k$$

$$\mathbf{c} := \left[\beta_{j-n} \exp\left(-\frac{\pi i (j-n)}{2(n+1)}\right)\right]_{j=0}^{2n+1}$$

$$\mathbf{b} := \left[z_k \exp\left(-\frac{\pi i n k}{n+1}\right)\right]_{k=0}^{2n+1}$$

$$\boxed{F_{2(n+1)}\, c = b}$$

$(2n+2) \times (2n+2)$ Fourier matrix (Ch. 4)

inverse DFT to recover c $\Rightarrow$ can recover β_j's

$\Rightarrow$ recover α_j's.

$\underline{\underline{O(n \log n)}}$

~~MATLAB~~ EIGEN-code 6.1.112: Efficient computation of Chebychev expansion coefficient of Chebychev interpolant

```
// efficiently compute coefficients α_j in the Chebychev expansion
// p = Σ_{j=0}^{n} α_j T_j of p ∈ P_n based on values y_k,
// k = 0,...,n, in Chebychev nodes t_k, k = 0,...,n
// IN: values y_k passed in y
// OUT: coefficients α_j
VectorXd chebexp(const VectorXd& y) {
  const int n = y.size() - 1;          // degree of polynomial
  const std::complex<double> M_I(0, 1); // imaginary unit
  // create vector z, see (6.1.110)
  VectorXcd b(2*(n + 1));
  const std::complex<double> om = -M_I*(M_PI*n)/((double)(n+1));
  for (int j = 0; j <= n; ++j) {
    b(j) = std::exp(om*double(j))*y(j); // this cast to double is
        necessary!!
    b(2*n+1-j) = std::exp(om*double(2*n+1-j))*y(j);
  }

  // Solve linear system (6.1.111) with effort O(n log n)
  Eigen::FFT<double> fft;   // EIGEN's helper class for DFT
  VectorXcd c = fft.inv(b); // -> c = ifft(z), inverse fourier
    transform
  // recover β_j, see (6.1.111)
  VectorXd beta(c.size());
  const std::complex<double> sc = M_PI_2/(n + 1)*M_I;
  for (unsigned j = 0; j < c.size(); ++j)
    beta(j) = ( std::exp(sc*double(-n+j))*c[j] ).real();
  // recover α_j, see (6.1.108)
  VectorXd alpha = 2*beta.tail(n); alpha(0) = beta(n);
  return alpha;
}
```

6.5.1. Piecewise polynomial Lagrange interpolation

Recall estimate for Chebychev interpolation:

$$\|f - L_{\mathcal{T}} f\|_{L^\infty(I)} \leq \frac{2^{-2n-1}}{(n+1)!} |I|^{n+1} \|f^{(n+1)}\|_{L^\infty(I)}$$

one way to reduce RHS: cut I into small pieces

Given interval $I = [a,b] \subset \mathbb{R}$, take a mesh $\mathcal{M}$ of I:

$$\mathcal{M} := \{ a = x_0 < x_1 < \dots < x_{m-1} < x_m = b \}$$

Local Lagrange interpolation of $f \in C(I)$ on $\mathcal{M}$:

Terminology:

- $x_j \;\hat{=}$ nodes of the mesh $\mathcal{M}$,
- $[x_{j-1}, x_j[\;\hat{=}$ intervals/cells of the mesh,
- $h_{\mathcal{M}} := \max_j |x_j - x_{j-1}| \;\hat{=}$ mesh width,
- If $x_j = a + jh \;\hat{=}$ equidistant (uniform) mesh with meshwidth $h > 0$

General local Lagrange interpolation on a mesh

1. Choose local degree $n_j \in \mathbb{N}_0$ for each cell of the mesh, $j = 1, \dots, m$.
2. Choose set of *local* interpolation nodes

$$\mathcal{T}^j := \{t_0^j, \dots, t_{n_j}^j\} \subset I_j := [x_{j-1}, x_j], \quad j = 1, \dots, m,$$

for each mesh cell/grid interval I_j.

3. Define piecewise polynomial interpolant $s : [x_0, x_m] \to \mathbb{K}$:

$$s_j := s_{|I_j} \in \mathcal{P}_{n_j} \quad \text{and} \quad s_j(t_i^j) = f(t_i^j) \quad i = 0, \dots, n_j, \quad j = 1, \dots, m. \tag{6.5.5}$$

Owing to Thm. 5.2.14, s_j is well defined.

for every cell: size of node set $n_j + 1$

Corollary 6.5.7. Continuous local Lagrange interpolants

If the local degrees n_j are at least 1 and the local interpolation nodes t_k^j, $j = 1,\dots,m$, $k = 0,\dots,n_j$, for local Lagrange interpolation satisfy

$$t_{n_j}^j = t_0^{j+1} \quad \forall j = 1,\dots,m-1 \quad \Rightarrow \quad \underline{s \in C^0([a,b])}\,, \tag{6.5.8}$$

then the piecewise polynomial Lagrange interpolant according to (6.5.5) is continuous on $[a,b]$: $s \in C^0([a,b])$.

$$t_{n_j}^j = t_0^{j+1} = x_j$$

$\Rightarrow$ $x_1, \dots, x_{m-1}$ are interpolation nodes

Example:

$f(t) = \arctan(t)$ on $I = [-5, 5]$

$m = 4$ $\quad \mathcal{M} = \{-5 = x_0 < x_1 < x_2 < x_3 < x_4 = 5\}$

piecewise linear : $n_j = 1$ $\quad \mathcal{T}^j = \{t_0^j = x_{j-1}, t_1^j = x_j\}$

piecewise quadratic : $n_j = 2$ $\quad \mathcal{T}^j = \{x_{j-1}, \frac{x_{j-1}+x_j}{2}, x_j\}$

overall a C^0 interpolant (but not C^1)

Interpolant will be C^0 if $x_1, \dots, x_{m-1}$ are interp. nodes

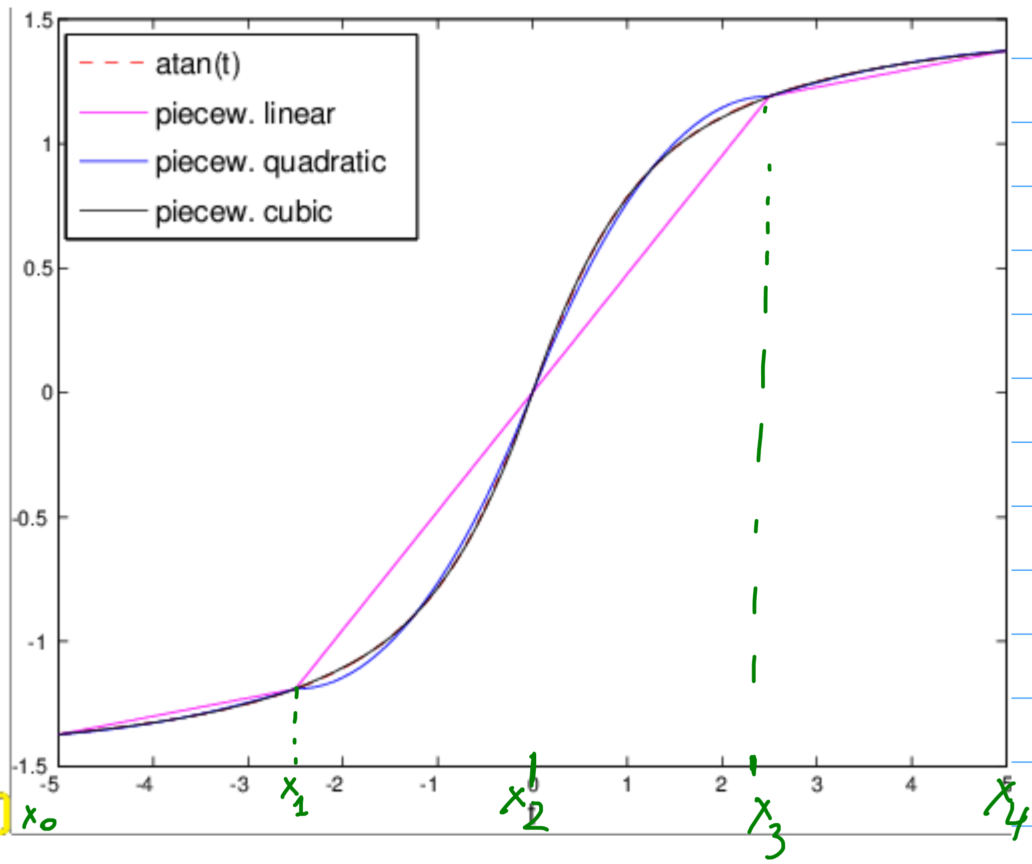


Fig. 250

Special case $n_j = n$ (fixed)

Can we improve our error estimate by decreasing the mesh width $h_{\mathcal{M}} := \max_j |x_{j-1} - x_j|$?

i.e. asymptotics as $h_{\mathcal{M}} \to 0$ "h-convergence"

Note: as we decrease $h_{\mathcal{M}}$: increasing total number of nodes as number of cells increases

$$\# \text{ of cells} \geq \frac{|b-a|}{h_{\mathcal{M}}}$$

$$\# \text{ of nodes} \geq \frac{|b-a|(n-1)}{h_{\mathcal{M}}}$$

Example: $f(t) = \arctan(t)$ on $[-5,5]$

equidistant mesh with $\frac{10}{h_{\mathcal{M}}}$ cells

$\|f-s\|_{\infty}$

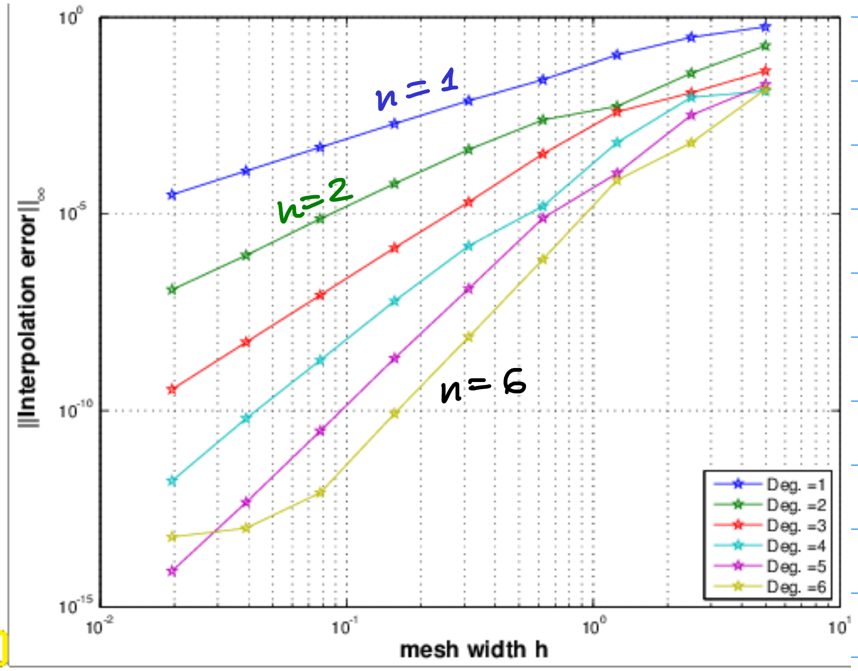


Fig. 252

log-log plot

⇝ algebraic convergence in $h_{\mathcal{M}}$

Derivation of error estimate:

Apply old estimate on the subintervals:

$$\|f - L_{\mathcal{T}} f\|_{L^\infty(I_j)} \leq \frac{\|f^{(n+1)}\|_{L^\infty(I_j)}}{(n+1)!} \underbrace{\max_{t \in I_j} |(t-t_0^j)\cdots(t-t_n^j)|}_{\leq h_{\mathcal{M}}^{n+1}}$$

$$I_j = [x_{j-1}, x_j]$$

$$h_{\mathcal{M}} := \max\{|I_j| : j = 1,\dots,m\}$$

$$\|f-s\|_{L^\infty([x_0,x_m])} = \max_{j\in\{1,\dots,m\}} \|f - L_{\mathcal{T}} f\|_{L^\infty(I_j)}$$

$$\leq \frac{h_{\mathcal{M}}^{n+1}}{(n+1)!} \|f^{(n+1)}\|_{L^\infty([x_0,x_m])}$$

→ algebraic convergence in $h_{\mathcal{M}}$ with rate $n+1$.

Remark: L^2-estimate:

$$\|f - L_j f\|_{L^2(I_j)} \leq \frac{2^{(n-1)/4}\,|I_j|^{n+1}}{\sqrt{n!\,(n+1)!}} \|f^{(n+1)}\|_{L^2(I_j)}$$

$$\|f - s\|^2_{L^2(I)} = \sum_{j=1}^{m} \|f - L_j f\|^2_{L^2(I_j)}$$

$$\leq \frac{2^{(n-1)/2}}{n!\,(n+1)!} \sum_{j=1}^{m} \underbrace{|I_j|}_{\leq h_\mu}{}^{2(n+1)} \|f^{(n+1)}\|^2_{L^2(I_j)}$$

$$\leq h_\mu^{2(n+1)} \frac{2^{(n-1)/2}}{n!\,(n+1)!} \|f^{(n+1)}\|^2_{L^2([x_0, x_m])}$$

$\uparrow$

$I_k \cap I_l = \emptyset$ for $k \neq l$

$$\Rightarrow \quad \|f - s\|_{L^2(I)} \leq h_\mu^{n+1} \frac{2^{(n-1)/4}}{\sqrt{n!\,(n+1)!}} \|f^{(n+1)}\|_{L^2([x_0, x_m])}.$$

$\rightarrow$ algebraic converge in h_μ at rate $n+1$.

Note:
- n is now fixed (and small)

  e.g. for p.w. linear $n=1$ and estimate holds if $f|_{I_j} \in C^2$
- piecewise smoothness of f is sufficient!
- since n can be small, convergence result also for f with low regularity
- locality
- BUT: slow convergence (algebraic as opposed to exp. for Chebychev interp.)

Remark:

Similar estimate is possible for cubic spline interp.

For equidistant mesh with mesh width h:

$$f \in C^4([t_0, t_n]) \qquad \|f - s\|_{L^\infty([t_0,t_n])} \leq \underbrace{\frac{5}{384} h^4}_{\text{alg. conv. in } h} \left\| f^{(4)} \right\|_{L^\infty([t_0,t_n])}$$

7. Numerical Quadrature

[approximate numerical evaluation of integrals]

Approximate $\int_a^b f(t)\,dt$ using only point evaluations of f.

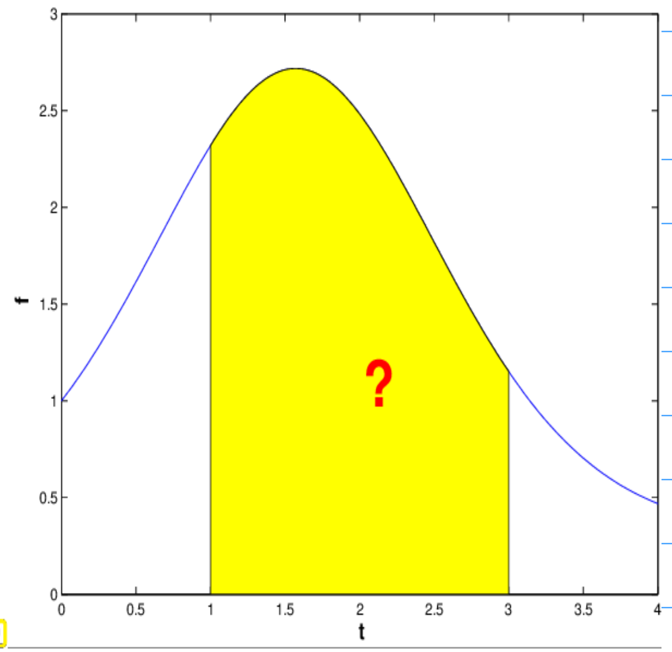


Fig. 259

Why numerical integration?

- $f(t)$ might only be available at sampling points
- we have a routine (formula) for $f(t)$ but $\int f(t)\,dt$ is difficult to compute
- We may have a formula for $\int f(t)\,dt$ but numerical integration is easier

Applications:
- direct application of eval. integrals
- indirect application for solving ODEs/PDEs (e.g. FEM)

7.1 Quadrature Formulas

Definition 7.1.1. Quadrature formula/quadrature rule

An n-point quadrature formula/quadrature rule on $[a,b]$ provides an approximation of the value of an integral through a *weighted sum* of point values of the integrand:

$$\int_a^b f(t)\,\mathrm{d}t \approx \underbrace{Q_n(f)}_{\text{QF}} := \sum_{j=1}^{n} \underbrace{w_j^n}_{\substack{\text{quadrature}\\ \text{weights } \in\mathbb{R}}} f(\underbrace{c_j^n}_{\substack{\text{quadrature nodes}\\ \in[a,b]}}) \,. \tag{7.1.2}$$

cost of evaluation of $Q_n(f)$: n point eval. of f &

n multiplications & additions

Remark: as in interpolation:

sufficient to consider reference interval $[-1,1]$

(& then use affine pullback on general intervals)

More precisely:

Suppose given QF $(\hat{c}_j, \hat{\omega}_j)_{j=1}^{n}$ on $[-1,1]$

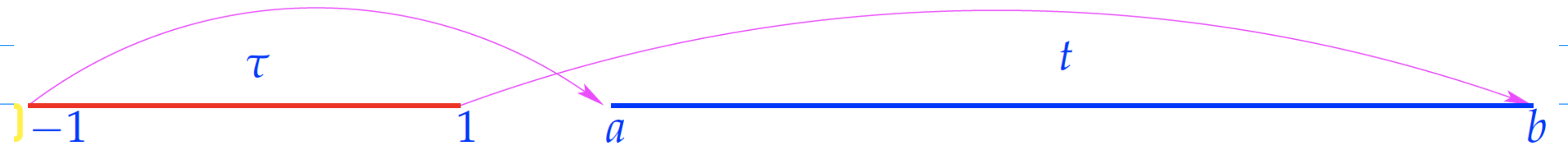
ref. interval

$\int_a^b f(t)\,dt$ can be transformed to $[-1,1]$:

$$\int_a^b f(t)\,dt = \int_{-1}^{1} f(\Phi(\tau))\,\Phi'(\tau)\,d\tau = \frac{b-a}{2}\int_{-1}^{1} \hat{f}(\tau)\,d\tau$$

$$\Phi(\tau) = \frac{1}{2}(1-\tau)a + \frac{1}{2}(1+\tau)b$$

i.e. $\hat{f}$ is the affine pullback $\Phi^* f$ of f to $[-1,1]$



▶ quadrature formula for general interval $[a,b]$, $a,b\in\mathbb{R}$:

$$\boxed{\int_a^b f(t)\,\mathrm{d}t \approx \tfrac{1}{2}(b-a)\sum_{j=1}^{n}\widehat{w}_j \widehat{f}(\widehat{c}_j) = \sum_{j=1}^{n} w_j f(c_j) \quad \text{with} \quad \begin{array}{l} c_j = \tfrac{1}{2}(1-\widehat{c}_j)a + \tfrac{1}{2}(1+\widehat{c}_j)b\,, \\ w_j = \tfrac{1}{2}(b-a)\widehat{w}_j\,. \end{array}}$$

Need: $\displaystyle\int_a^b f(t)\,dt = \sum_{j=1}^{n} \omega_j f(c_j)$

$$= \frac{1}{2}(b-a)\sum_{j=1}^{n} \hat{\omega}_j \hat{f}(\hat{c}_j)$$

$$c_j = \Phi(\hat{c}_j)$$

$$\omega_j = \frac{|[a,b]|}{|[-1,1]|}\,\hat{\omega}_j$$

Quadrature by approximation schemes

Given approximation scheme $A : C^0([a,b]) \to V$

where V is a space of "simple" functions on $[a,b]$,

we can find a numerical integration method

$$\int_a^b f(t)\,dt \approx \int_a^b (Af)(t)\,dt =: Q_A(f)$$

Recall: every interpolation scheme induces an

approximation scheme

Interpolation scheme $I_\mathcal{T}$ with node set $\mathcal{T} = \{t_1, \dots, t_n\}$

$\subset [a,b]$

$$\int_a^b f(t) \approx \int_a^b I_\mathcal{T}\,[f(t_1), \dots, f(t_n)]^T(t)\,dt \qquad (*)$$

If $I_\mathcal{T}$ is a linear interpolation operator, then

$(*)$ yields a QF.

$$\int_a^b I_\mathcal{T}\left[\left[f(t_i)\right]_{i=1}^{n}\right]^T(t)\,dt = \underline{\underline{\sum_j \omega_j f(c_j)}}$$

Why?

$$\int_a^b I_\mathcal{T}\,[f(t_1), \dots, f(t_n)]^T(t)\,dt = \int_a^b I_\mathcal{T}\left[\sum_{j=1}^n f(t_j)\cdot e_j\right](t)\,dt$$

$$\underset{\text{linearity}}{=} \sum_{j=1}^n f(t_j) \underbrace{\int_a^b I_\mathcal{T}\,[e_j](t)\,dt}_{=:\,\omega_j^n} = \sum_{j=1}^n \omega_j^n f(t_j)$$

interpolation schemes $\longrightarrow$ approximation schemes $\longrightarrow$ quadrature schemes

Quadrature error: $E_n(f) = \left| \int_a^b f(t)\,dt - Q_n(f) \right|$

Asymptotic behavior of $E_n(f)$ as $n \to \infty$

Simple estimate if QF is induced by interp. scheme $I_{\mathcal{T}}$:

$$E_n(f) = \left| \int_a^b \left(f(t) - I_{\mathcal{T}}[f(t_1), \dots, f(t_n)]^T(t) \right) dt \right|$$

$$\leq |b-a| \cdot \underbrace{\| f - I_{\mathcal{T}}[f(t_1), \dots, f(t_n)]^T \|_{L^\infty([a,b])}}_{\substack{\text{interpolation error} \\ \text{(Chapter 6)}}}$$

7.2. Polynomial Quadrature Formulas

QF induced by Lagrange interpolation scheme $I_{\mathcal{T}}$

Idea: replace integrand f with $p_{n-1} := \mathsf{I}_{\mathcal{T}} \in \mathcal{P}_{n-1}$ = polynomial Lagrange interpolant of f ($\rightarrow$ Cor. 5.2.15) for given node set $\mathcal{T} := \{t_0, \dots, t_{n-1}\} \subset [a,b]$

$$\int_a^b f(t)\,dt \approx Q_n(f) := \int_a^b p_{n-1}(t)\,dt\,. \qquad (7.2.1)$$

Lagrange interpolant $p_{n-1}(t) = \sum_{i=0}^{n-1} f(t_i) L_i(t)$

Lagrange poly. $L_i(t) = \prod_{\substack{j=0 \\ j\neq i}}^{n-1} \frac{(t-t_j)}{(t_i - t_j)}$

QF: $\int_a^b p_{n-1}(t)\,dt = \sum_{i=0}^{n-1} f(t_i) \int_a^b L_i(t)\,dt$

$$= \sum_{i=1}^{n} f(t_{i-1}) \int_a^b L_{i-1}(t)\,dt$$

weights $\quad w_i := \int_a^b L_{i-1}(t)\,dt \qquad i = 1, \dots, n$

nodes $\quad c_i := t_{i-1}$

Examples: Midpoint rule: $n = 1 \qquad t_0 = \frac{1}{2}(a+b)$

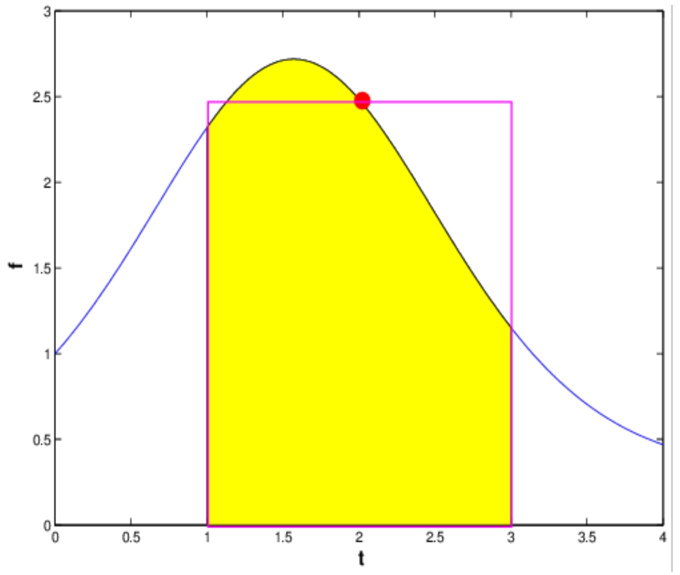


Fig. 263

The midpoint rule is (7.2.2) for $n = 1$ and $t_0 = \frac{1}{2}(a+b)$. It leads to the 1-point quadrature formula

$$\int_a^b f(t)\,dt \approx Q_{mp}(f) = (b-a) f(\tfrac{1}{2}(a+b)) .$$

"midpoint"

◁ the area under the graph of f is approximated by the area of a rectangle.

② Newton-Cotes formulas

n-point Newton-Cotes formula

Lagrange interpolation with equidistant nodes

$$t_j := a + \frac{b-a}{n-1} j \qquad j = 0, \dots, n-1$$

$n=2$: Trapezoidal rule

```
> trapez := newtoncotes(1);
```

$$\widehat{Q}_{trp}(f) := \frac{1}{2}(f(0) + f(1)) \qquad (7.2.5)$$

$$\left(\int_a^b f(t)\,dt \approx \frac{b-a}{2}(f(a) + f(b)) \right)$$

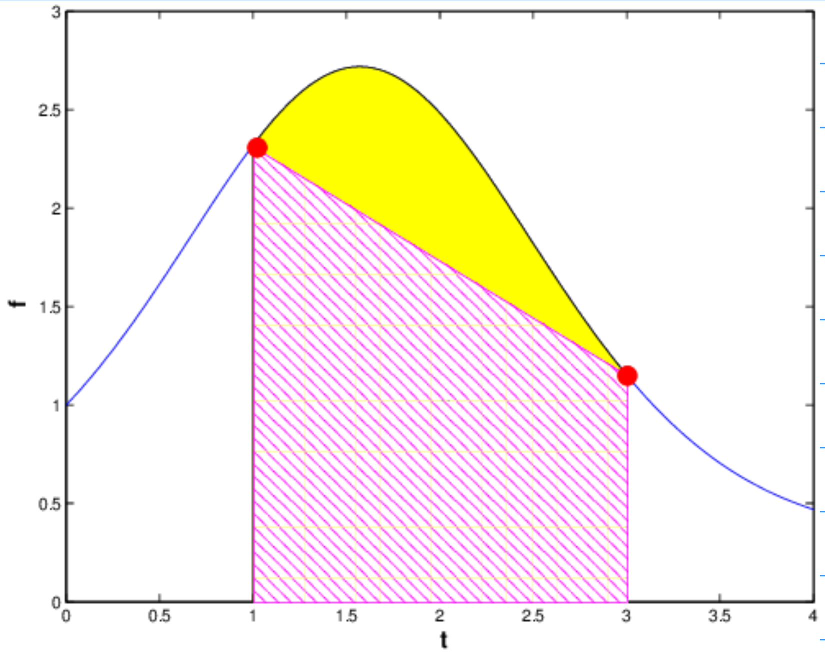


Fig. 264

$$w_i = \int_a^b L_{i-1}(t)\,dt$$

$$w_1 = \int_a^b L_0(t)\,dt = \int_a^b \frac{t-b}{a-b}\,dt = \frac{b-a}{2}$$

$$w_2 = \int_a^b L_1(t)\,dt = \int_a^b \frac{t-a}{b-a}\,dt = \frac{b-a}{2}$$

- $n = 3$: Simpson rule

```
> simpson := newtoncotes(2);
```

$$\frac{h}{6}\left(f(0)+4f\left(\tfrac{1}{2}\right)+f(1)\right) \quad \left(\int_a^b f(t)\,\mathrm{d}t \approx \frac{b-a}{6}\left(f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right)\right) \qquad (7.2.6)$$

Recall: Lagrange interpolation with equidistant nodes is unstable for large n!

Remedy: Chebychev nodes $\rightsquigarrow$ yield Clenshaw-Curtis QF

Next: In some sense "optimal" QFs

↑ depending on how quality of a QF is defined

7.3. Gauss Quadrature

Quality measure for QF (independent of a specific integrand)

Definition 7.3.1. Order of a quadrature rule

The order of quadrature rule $Q_n : C^0([a,b]) \to \mathbb{R}$ is defined as

$$\operatorname{order}(Q_n) := \max\left\{m \in \mathbb{N}_0 :\ Q_n(p) = \int_a^b p(t)\,\mathrm{d}t \quad \forall p \in \mathcal{P}_m\right\}+1\,, \qquad (7.3.2)$$

that is, as the maximal degree $+1$ of polynomials for which the quadrature rule is guaranteed to be exact.

Note: order of QF is invariant under affine transformations (such as pullback)

Example: Polynomial QF with n points:

(exact for $p \in \mathcal{P}_{n-1}$) order $\geq n$

Question: When does an n-point QF have order $\geq n$?

Theorem 7.3.5. Sufficient order conditions for quadrature rules

An n-point quadrature rule on $[a,b]$ ($\to$ Def. 7.1.1)

$$Q_n(f) := \sum_{j=1}^{n} w_j f(t_j)\,, \quad f \in C^0([a,b])\,,$$

with nodes $t_j \in [a,b]$ and weights $w_j \in \mathbb{R}$, $j = 1,\dots,n$, has order $\geq n$, if and only if

$$w_j = \int_a^b L_{j-1}(t)\,dt\,, \quad j = 1,\dots,n\,,$$

where L_k, $k = 0,\dots,n-1$, is the k-th Lagrange polynomial (5.2.11) associated with the ordered node set $\{t_1,t_2,\dots,t_n\}$.

Note: for QF Q_n to have order $\geq n$ weights w_j only depend on node set $\mathcal{T} = \{t_1, \dots, t_n\}$

Proof:

Q_n has order $\geq n \iff Q_n(p) = \int_a^b p(t)\,dt \quad \forall p \in \mathcal{P}_{n-1}$

Note: $\mathcal{P}_{n-1} = \operatorname{span}\{L_0, \dots, L_{n-1}\}$

$$Q_n(p) = \int_a^b p(t)\,dt \quad \forall p \in \mathcal{P}_{n-1} \iff Q_n(L_{i-1}) = \int_a^b L_{i-1}(t)\,dt \quad \forall i \in \{1,\dots,n\}$$

$$\iff \underbrace{Q_n(L_{i-1})}_{} = \sum_{j=1}^{n} w_j \underbrace{L_{i-1}(t_j)}_{\delta_{i,j}} = \int_a^b L_{i-1}(t)\,dt \quad \forall i \in \{1,\dots,n\}$$

$$\omega_i = \int_a^b L_{i-1}(t)\,dt \,. \qquad \square$$

Next natural question: Existence of n-point QFs with order $> n$?

(We know n-point QFs order $\geq n$)

Theorem 7.3.12. Maximal order of n-point quadrature rule

The maximal order of an n-point quadrature rule is $2n$.

Proof: $Q_n(f) := \sum_{j=1}^{n} \omega_j^n f(c_j^n)$

and construct $q \in \mathcal{P}_{2n}$ s.t.

$$Q_n(q) \neq \int_a^b q(t)\,dt \,.$$

($\Rightarrow$ order $< 2n+1$)

$$q(t) := (t-c_1^n)^2 (t-c_2^n)^2 \cdots (t-c_n^n)^2 \in \mathcal{P}_{2n}$$

$q(t) > 0$ almost everywhere

$$\Rightarrow \int_a^b q(t)\,dt > 0$$

$$Q_n(q) = \sum_{j=1}^{n} \omega_j^n \underbrace{q(c_j^n)}_{=0} = 0 \,.$$

$$\Rightarrow 0 = Q_n(q) \neq \int_a^b q(t)\,dt > 0 \qquad \square$$

Example: 2-point QF Q_2 with order 4 (on $[-1,1]$)

$$Q_n(p) = \int_a^b p(t)\,dt \;\forall p \in \mathcal{P}_3 \;\Leftrightarrow\; Q_n(\{t \mapsto t^q\}) = \frac{1}{q+1}(b^{q+1} - a^{q+1}),\quad q = 0,1,2,3 \,.$$

check only on monomials (or any basis of $\mathcal{P}_3$)

4 equations for weights w_j and nodes c_j, $j = 1,2$ $\quad(a=-1, b=1)$, cf. Rem. 7.3.6

$$\begin{aligned} \int_{-1}^{1} 1\,\mathrm{d}t = 2 = 1w_1 + 1w_2 \quad &, \quad \int_{-1}^{1} t\,\mathrm{d}t = 0 = c_1 w_1 + c_2 w_2 \\ \int_{-1}^{1} t^2\,\mathrm{d}t = \frac{2}{3} = c_1^2 w_1 + c_2^2 w_2 \quad &, \quad \int_{-1}^{1} t^3\,\mathrm{d}t = 0 = c_1^3 w_1 + c_2^3 w_2 \,. \end{aligned} \tag{7.3.14}$$

4 (nonlinear) equations in 4 unknowns

➣ weights & nodes: $\{w_2 = 1, w_1 = 1, c_1 = 1/3\sqrt{3}, c_2 = -1/3\sqrt{3}\}$

▶ quadrature formula (order 4): $\int_{-1}^{1} f(x)\,dx \approx f\left(\frac{1}{\sqrt{3}}\right) + f\left(-\frac{1}{\sqrt{3}}\right)$ (7.3.15)

Question: Is there a family Q_n of QFs

s.t. Q_n is
- n point
- of order $2n$

Suppose this was the case!

Optimist's **assumption**: $\exists$ family of n-point quadrature formulas on $[-1,1]$

$$Q_n(f) := \sum_{j=1}^{n} w_j^n f(c_j^n) \approx \int_{-1}^{1} f(t)\,\mathrm{d}t\,, \quad w_j \in \mathbb{R}\,, n \in \mathbb{N}\,,$$

of order $2n$ $\Leftrightarrow$ exact for polynomials $\in \mathcal{P}_{2n-1}$. (7.3.17)

Define $\bar{P}_n(t) := (t - c_1^n) \cdot \dots \cdot (t - c_n^n)\,,\ t \in \mathbb{R} \Rightarrow \bar{P}_n \in \mathcal{P}_n$.

leading coeff. of $\bar{P}_n$ is 1.

$\forall q \in \mathcal{P}_{n-1}$: $q \cdot \bar{P}_n \in \mathcal{P}_{2n-1}$

$$\Rightarrow \underbrace{\int_{-1}^{1} q(t)\,\bar{P}_n(t)\,dt}_{\langle q, \bar{P}_n\rangle_{L^2([-1,1])}} \underset{\substack{\text{exact}\\ \text{QF}\\ \text{on } \mathcal{P}_{2n-1}}}{=} \sum_{j=1}^{n} w_j^n\, q(c_j^n) \underbrace{\bar{P}_n(c_j^n)}_{=0\ \forall j = \{1,\dots,n\}} = 0$$

$$\bar{P}_n \perp \mathcal{P}_{n-1} \quad \text{in } L^2([-1,1]) \qquad (*)$$

$$\bar{P}_n(t) = t^n + \alpha_{n-1} t^{n-1} + \dots + \alpha_1 t + \alpha_0 .$$

$\bar{P}_n(t)$ is determined by <u>n coefficients</u>

$$\alpha_0, \dots, \alpha_{n-1}$$

(*) n conditions [$\dim \mathcal{P}_{n-1} = n$

$$\int_{-1}^{1} \bar{P}_n(t)\, t^l \, dt = 0 \quad \forall l = 0, \dots, n-1]$$

$\Rightarrow$ $\bar{P}_n \in \mathcal{P}_n$ is unique [if it exists]

$\uparrow$

fulfilling (*) & having leading coeff. 1.

Find $\bar{P}_n$:

$$\int_{-1}^{1} q(t)\, \bar{P}_n(t)\, dt = 0 \qquad \forall q \in \mathcal{P}_{n-1}$$

$$\Leftrightarrow \int_{-1}^{1} \underset{\substack{\uparrow \\ \text{monomials}}}{t^l} \underbrace{\left(t^n + \sum_{j=0}^{n-1} \alpha_j t^j \right)}_{\bar{P}_n} dt = 0 \qquad \forall l = 0, \dots, n-1$$

$$\sum_{j=0}^{n-1} \alpha_j \int_{-1}^{1} t^l t^j \, dt = - \int_{-1}^{1} t^l t^n \, dt$$

Can be written as $\quad A \, [\alpha_j]_{j=0}^{n-1} = b$

$$A_{j,l} = \int_{-1}^{1} t^l t^j \, dt = \langle t^j, t^l \rangle_{L^2([-1,1])}$$

A is symmetric.

$$x^T A x = \sum_{\ell=0}^{n-1} x_\ell \left(\sum_{j=0}^{n-1} \int_{-1}^{1} t^j t^\ell \, dt \; x_j \right)$$

$$= \int_{-1}^{1} \left(\sum_{\ell=0}^{n-1} x_\ell t^\ell \right) \left(\sum_{j=0}^{n-1} x_j t^j \right) dt$$

$$= \int_{-1}^{1} \left(\sum_{j=0}^{n-1} x_j \, t^j \right)^2 dt > 0$$

if $x \neq 0$

$\Rightarrow$ A sym. positive definite

$\Rightarrow$ $[\alpha_j]_{j=0}^{n-1}$ exists & is unique.

Theorem 7.3.22. Existence of n-point quadrature formulas of order $2n$

Let $\{\bar{P}_n\}_{n\in\mathbb{N}_0}$ be a family of non-zero polynomials that satisfies

- $\bar{P}_n \in \mathcal{P}_n$,
- $\int_{-1}^{1} q(t)\bar{P}_n(t)\,dt = 0$ *for all* $q \in \mathcal{P}_{n-1}$ ($L^2([-1,1])$-*orthogonality*),
- *The set* $\{c_j^n\}_{j=1}^m$, $m \le n$, *of real zeros of* $\bar{P}_n$ *is contained in* $[-1,1]$.

Then the quadrature rule (→ Def. 7.1.1) $Q_n(f) := \sum_{j=1}^{m} w_j^n f(c_j^n)$

with weights chosen according to Thm. 7.3.5 provides a quadrature formula of order $2n$ on $[-1,1]$.

$\Rightarrow$ n point QF with order $2n$:

nodes will have to be zeros of $\bar{P}_n$.

▶ n-point quadrature formulas of order $2n$ are unique

Polynomials $\bar{P}_n$ are up to scaling factor the Legendre polynomials:

Definition 7.3.27. Legendre polynomials

The n-th Legendre polynomial P_n is defined by

- $P_n \in \mathcal{P}_n$,
- $\int_{-1}^{1} P_n(t) q(t) \, dt = 0 \quad \forall q \in \mathcal{P}_{n-1}$,
- $P_n(1) = 1$.

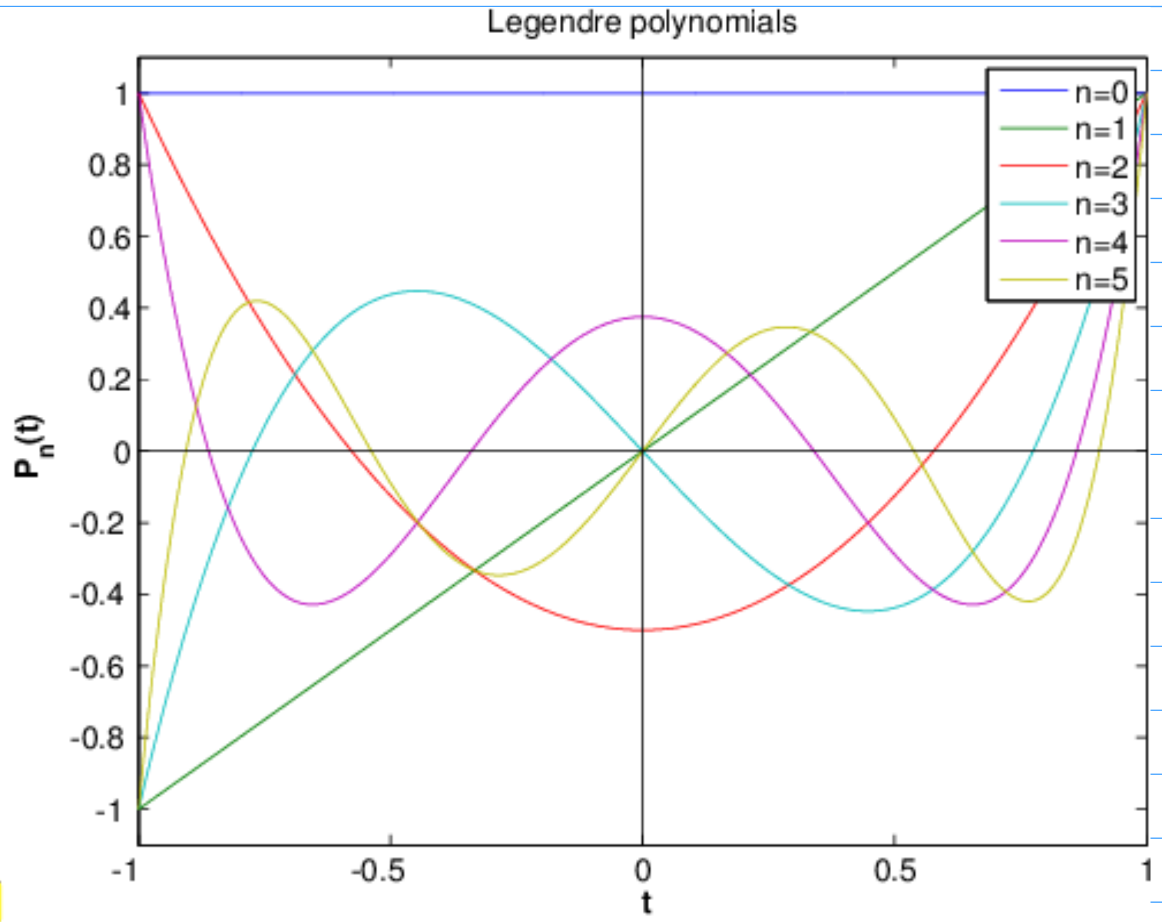


Fig. 265

One more thing:

Lemma 7.3.28. Zeros of Legendre polynomials

P_n has n distinct zeros in $]-1,1[$.

Zeros of Legendre polynomials = Gauss points

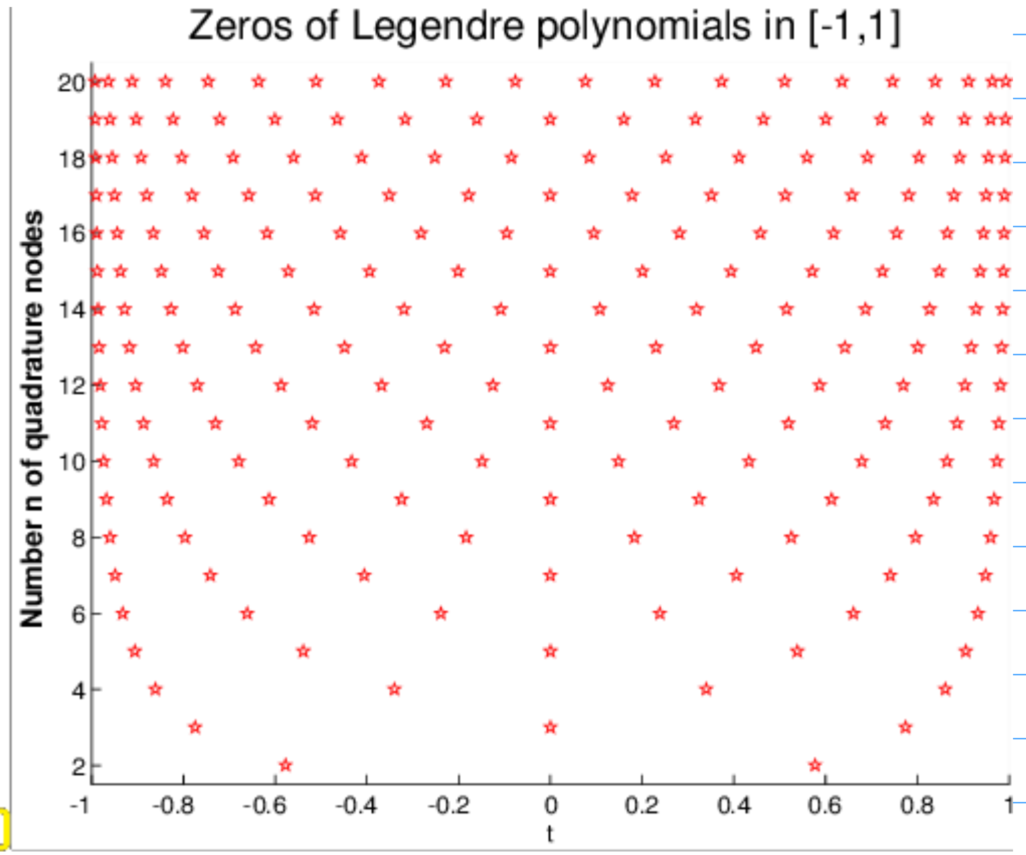


Fig. 266

Proof: Assume P_n has only $m<n$ zeros

$\xi_1, \dots, \xi_m \in (-1,1)$.

$\rightarrow P_n$ changes sign at $\xi_1, \dots, \xi_m$

Define $q(t) := \prod_{j=1}^{m} (t - \xi_j) \in \mathcal{P}_{m-1} \subset \mathcal{P}_{n-1}$

$\Rightarrow q$ changes sign at $\xi_1, \dots, \xi_m$

$\Rightarrow P_n \cdot q \geq 0$ on $(-1,1)$

or $P_n \cdot q \leq 0$ on $(-1,1)$

$$\Rightarrow \int_{-1}^{1} P_n(t) q(t)\, dt \neq 0 \qquad \lightning$$

$P_n \perp \mathcal{P}_{n-1}$ $\square$

Definition 7.3.29. Gauss-Legendre quadrature formulas

The n-point Quadrature formulas whose nodes, the Gauss points, are given by the zeros of the n-th Legendre polynomial ($\rightarrow$ Def. 7.3.27), and whose weights are chosen according to Thm. 7.3.5, are called Gauss-Legendre quadrature formulas.