ETH Lecture 401-0663-00L Numerical Methods for CSE

Numerical Methods for Computational Science and Engineering

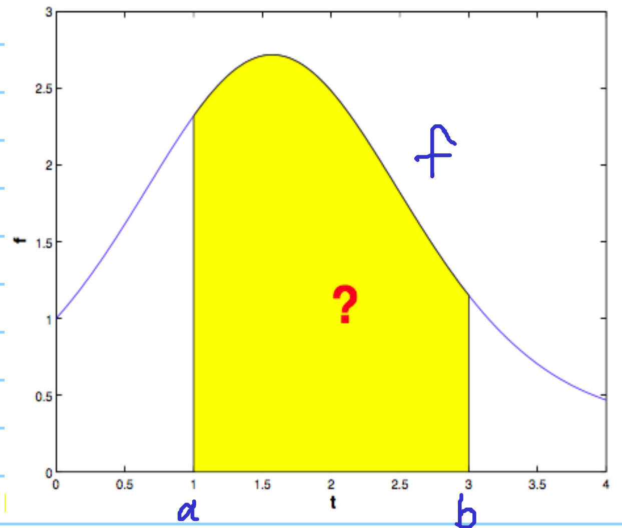
Prof. R. Hiptmair, SAM, ETH Zurich

(with contributions from Prof. P. Arbenz and Dr. V. Gradinaru)

Autumn Term 2015

URL: http://www.sam.math.ethz.ch/~hiptmair/tmp/NumCSE/NumCSE15.pdf

V. Numerical Quadrature



Task: Compute $\int_a^b f(t)\,dt$

f given in procedural form

[only point evaluations possible]

$\#\{f\text{-eval}\} \hat{=}$ measure of cost

Typical application:

Now assume time-harmonic periodic excitation with period $T > 0$.

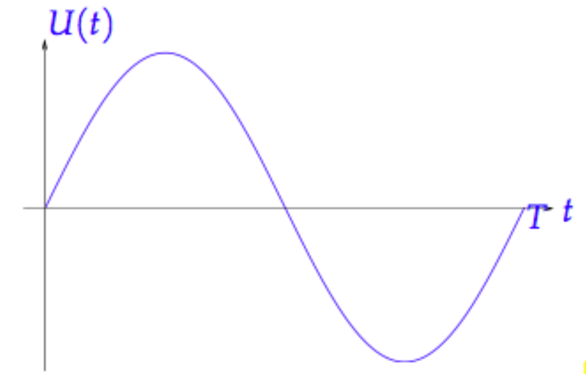


Fig. 165

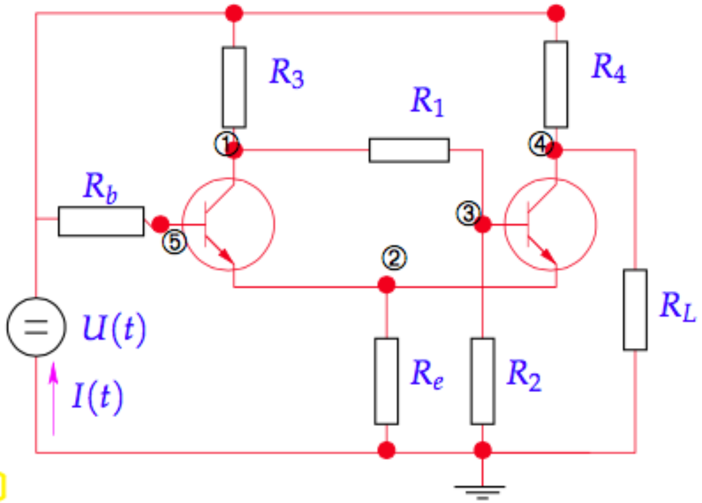


Fig. 166

Compute: $P = \int_0^T I(t)\,U(t)\,dt$

Circuit analysis: $I = I(U)$

$\hookrightarrow$ by solving a non-linear system!

5.1. Quadrature Formulas

Definition 5.1.1. 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:

$$\int_a^b f(t)\,\mathrm{d}t \approx Q_n(f) := \sum_{j=1}^{n} w_j^n f(c_j^n)\,. \quad (5.1.2)$$

Terminology: w_j^n : quadrature weights $\in \mathbb{R}$; c_j^n : quadrature nodes $\in [a,b]$: $n \sim$ cost

Affine transformation of quadrature formula:

$\phi : [-1,1] \to [a,b]$, $\phi(\tau) = a + \frac{1}{2}(b-a)(\tau+1)$

reference interval

↳ quadrature formula given there: $\hat{Q}_n(f) = \sum_{j=1}^{n} \hat{w}_j f(\hat{c}_j)$

$$\int_a^b f(t)\,dt = \int_{-1}^{1} f(\phi(\tau))\,\tfrac{1}{2}(b-a)\,d\tau$$

$$\approx \tfrac{1}{2}(b-a) \sum_{j=1}^{n} f(\phi(\hat{c}_j))\,\hat{w}_j$$

▷ $w_j = \frac{1}{2}(b-a)\hat{w}_j$ $c_j = \phi(\hat{c}_j)$

▷ In codes: tabulated quadrature rules on reference intervals

```
struct QuadTab {
  template <typename VecType>
    static void getrule(int n,VecType &c,VecType &w,double
      a=-1.0,double b=1.0);
}
```

Quadrature by interpolation & approximation

↳ scheme $A : C^0([a,b]) \to V$
$V \triangleq$ space of simple functions

$$\Rightarrow \quad Q_n(f) := \int_a^b (Af)(t)\,dt$$

Interpolation operator

$$I : \begin{cases} \mathbb{R}^n \times \mathbb{R}^n & \longrightarrow V \\ ([c_j]_j, [y_j]_j) & \longrightarrow I_J[y] \end{cases}$$

$$\int_a^b f(t)\,dt \approx \int_a^b I_J([f(c_j)]_j)(t)\,dt$$

Assumption: I_J is linear

$$[f(c_j)]_j = \sum_{j=1}^{n} f(c_j)\,e_j\,, \quad e_j = \begin{array}{c} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \end{array} \leftarrow j$$

$$▷ \int_a^b f(t)\,dt \approx \sum_{j=1}^{n} f(c_j) \underbrace{\int_a^b (I_J(e_j))(t)\,dt}_{= w_j\ !}$$

Here: $Af := I_J[f(c_j)]$

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

$$\leq |b-a|\, \|f - Af\|_{L^\infty([a,b])}$$

③ 5.2. Polynomial quadrature formulas

Now: $I_S \;\hat{=}\;$ Lagrange interpolation in $c_1, \dots, c_n$

$$\Rightarrow \quad Q_n(f) = \sum f(c_j)\, w_j \,, \quad w_j := \int_a^b L_{j-1}(t)\,dt$$

$L_j \;\hat{=}\;$ j-th Lagrange polynomial for node set $\{c_1, \dots, c_n\}$

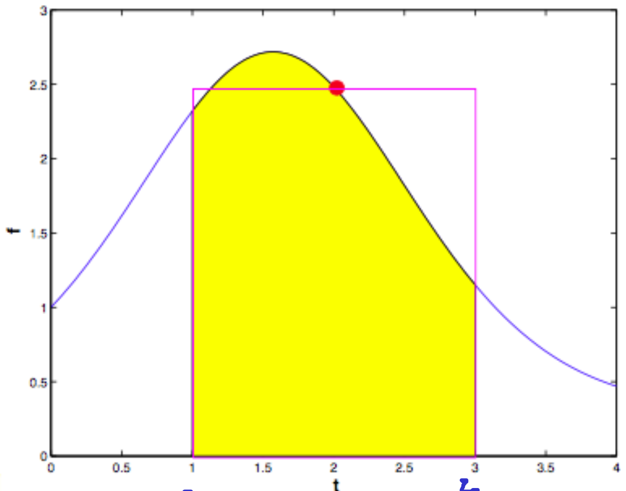
Examples: Midpoint rule

$n = 1$ (polynomial degree)

$c_1 = \frac{1}{2}(a+b)$

$w_1 = b - a$

order ≥ 1

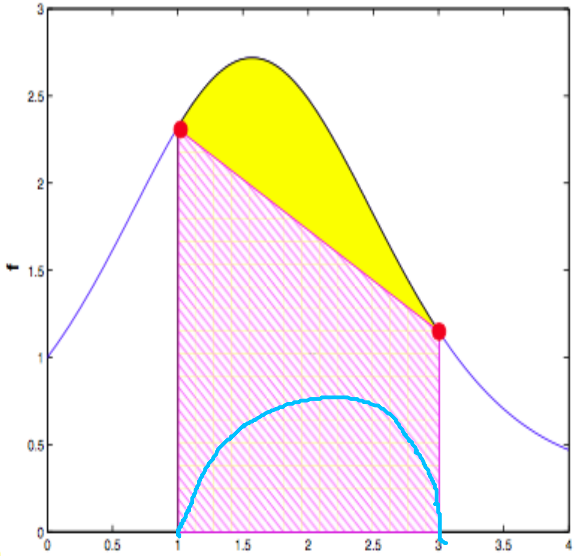


Example: Trapezoidal rule

$c_1 = a \,, \quad c_2 = b$

$w_1 = w_2 = \frac{1}{2}(b-a)$

$\Rightarrow$ order $= 2$



General: Newton-Cotes formulas

$\rightarrow$ Equidistant nodes: dangerous for $n \gg 1$

Do not use them!

Much better: Chebychev quadrature nodes

Clenshaw-Curtis rules

$\rightarrow$ positive weights throughout

5.3. Gauss quadrature

Quality criterion for a quadrature rule:

Definition 5.3.1. Order of a quadrature rule

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

$$\operatorname{order}(Q_n) := \max\{q \in \mathbb{N}_0 :\; Q_n(p) = \int_a^b p(t)\,\mathrm{d}t \;\; \forall p \in \mathcal{P}_q\} + 1 \,, \tag{5.3.2}$$

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

▽ $\mathcal{P}_n$ are invariant under affine pullback

$\rightarrow$ order of a Q.R. is not affected by transformation

Theorem 0.3.5. Sufficient order conditions for quadrature rules

An n-point quadrature rule on $[a,b]$ (→ Def. 5.1.1)

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

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 (3.2.11) associated with the ordered node set $\{t_1,t_2,\dots,t_n\}$.

eq:lagrangepol

Proof: $\{L_0,\dots,L_{n-1}\}$ is a basis of $\mathcal{P}_{n-1}$

$$Q_n(L_j) = W_{j+1} \underset{\text{exact!}}{\overset{\uparrow}{=}} \int_a^b L_j(t)\,dt \qquad \square$$

n-pt. Q.R. with order $> n$?

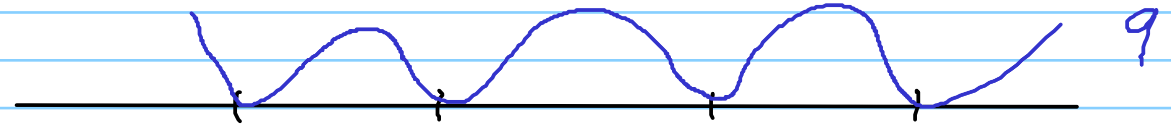
Theorem 5.3.9. Maximal order of n-point quadrature rule

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

Proof: (indirect) Assume $Q_n(f) = \sum_{j=1}^{n} W_j f(c_j)$ has order $2n+1$ $\Leftrightarrow$ exact $\forall q \in \mathcal{P}_{2n}$

$$q(t) = \prod_{j=1}^{n} (t-c_j)^2 \in \mathcal{P}_{2n}$$

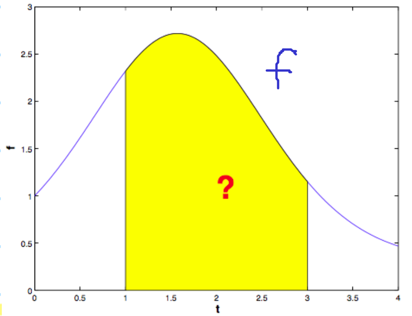
$$Q_n(q) = 0 \quad \longleftrightarrow \quad \int_a^b q(t)\,dt > 0$$



→ q cannot be integrated exactly $\square$

Warning + promise

Theory ahead !



$$\int_a^b f(t)\,dt \approx \sum_{j=1}^{n} w_j f(c_j) =: Q_n(f)$$

Definition 5.3.1. Order of a quadrature rule

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

$$\text{order}(Q_n) := \max\{q \in \mathbb{N}_0:\ Q_n(p) = \int_a^b p(t)\,dt \quad \forall q \in \mathcal{P}_n\}+1\,, \qquad (5.3.2)$$

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

Theorem 5.3.9. Maximal order of n-point quadrature rule

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

n-pt. quadrature rules of order $2n$? $\quad n=2$ ✓

"Counting argument": n-pt. rule $\longrightarrow$ $2n$ free parameters

$\Updownarrow$ match

$\dim \mathcal{P}_{2n-1} = 2n \longrightarrow 2n$ equations

$\left[\ Q_n(b_j) = \int_a^b b_j(t)\,dt\,, \quad \{b_0,\dots,b_{2n-1}\}\right.$ basis of $\mathcal{P}_{2n-1}$

Example 5.3.10 (2-point quadrature rule of order 4)

Necessary & sufficient conditions for order 4 , *cf.* (5.4.26):

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

4 equations for weights w_j and nodes c_j, $j=1,2$ $\quad (a=-1, b=1)$, *cf.* Rem. 5.4.24

non-linear equ.

$$\int_{-1}^{1} 1\,dt = 2 = 1w_1 + 1w_2 \ , \quad \int_{-1}^{1} t\,dt = 0 = c_1w_1 + c_2w_2$$
$$\int_{-1}^{1} t^2\,dt = \frac{2}{3} = c_1^2w_1 + c_2^2w_2 \ , \quad \int_{-1}^{1} t^3\,dt = 0 = c_1^3w_1 + c_2^3w_2\,.$$

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

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

Necessary conditions on quadrature rule of order $2n$:

Assume $\quad Q_n(f) = \sum_{j=1}^{n} w_j f(c_j)$ has order $2n$

$\bar{P}_n(t) = \prod_{j=1}^{n}(t-c_j) \in \mathcal{P}_n$, leading coeff. $=1$

$\forall\ q \in \mathcal{P}_{n-1}$: $\displaystyle\int_{-1}^{1} \underbrace{q\bar{P}_n}_{\in \mathcal{P}_{2n-1}}\,dt \underset{\text{order}=2n}{=} \sum_{j=1}^{n} w_j (q\bar{P}_n)(c_j) = 0$

$\Rightarrow \quad q \perp \bar{P}_n$ w.r.t. $L^2([-1,1])$-inner product

$\Rightarrow \quad \bar{P}_n \perp \mathcal{P}_{n-1} \Rightarrow \bar{P}_n$ unique (up to sign), because $\dim \mathcal{P}_n - \dim \mathcal{P}_{n-1} = 1$

$\rightarrow$ Nodes c_j are zeros of $\bar{P}_n$, thus fixed

Sufficient conditions :

Theorem 5.3.18. Existence of n-point quadrature formulas of order $2n$

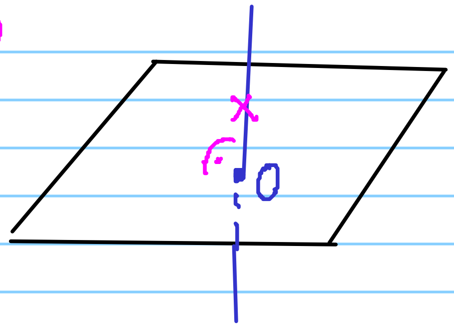
Let $\{\bar{P}_n\}_{n\in\mathbb{N}_0}$ be a family of non-zero polynomials that satisfies

- $\bar{P}_n \in \mathcal{P}_n$,
- $\int_{-1}^{1} q(t)\bar{P}_n(t)\,dt = 0$ *for all* $q \in \mathcal{P}_{n-1}$ $\qquad$ ($L^2([-1,1])$-*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

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

with weights chosen according to Thm. 5.3.5 provides a quadrature formula of order $2n$ on $[-1,1]$.



◁ Uniqueness of $\overline{P}_n$

Proof :

$$p(c_j) = r(c_j)$$

$p \in \mathcal{P}_{2n-1}$: $p = h\overline{P}_n + r$ with unique $h, r \in \mathcal{P}_{n-1}$

by, polynomial division,

$$\int_{-1}^{1} p(t)\,dt = \underbrace{\int_{-1}^{1} (h\overline{P}_n)(t)\,dt}_{=0 \text{ by orthogonality}} + \int_{-1}^{1} r(t)\,dt = \sum_{j=1}^{n} w_j \underbrace{r(c_j)}_{=p(c_j)}$$

order $\geq n$! □

Definition 5.3.23. Legendre polynomials

The n-th Legendre polynomial P_n is defined by

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

Legendre polynomials $P_0, \dots, P_5$ ➤

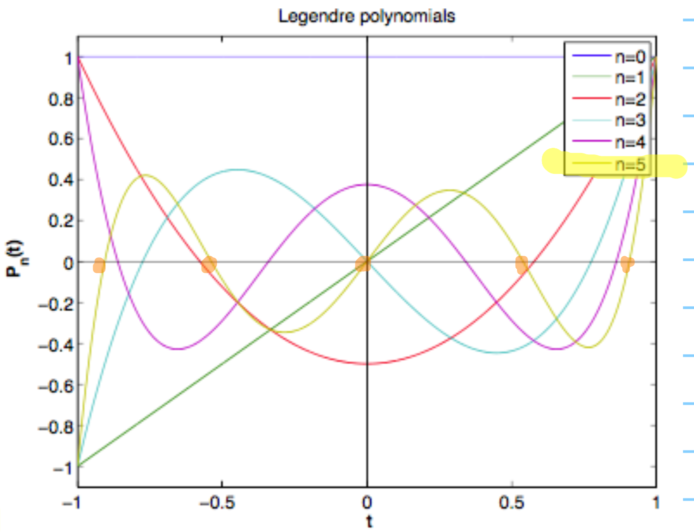
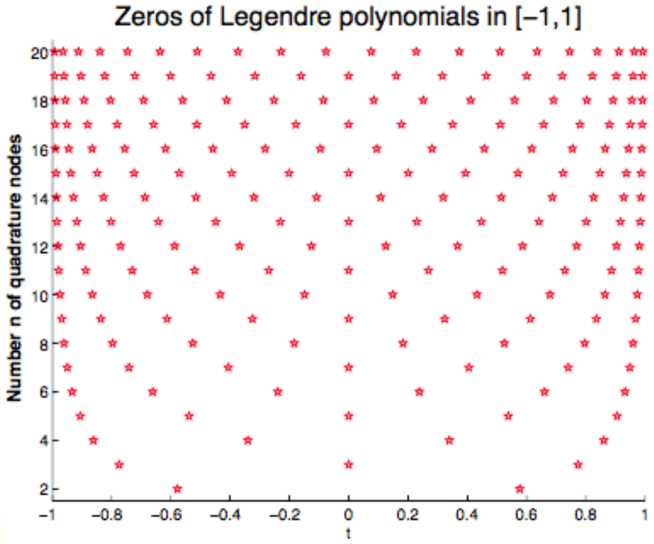


Fig. 169



◁ Obviously:

Lemma 5.3.24. Zeros of Legendre polynomials

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

Zeros of Legendre polynomials = Gauss points

Proof : (indirect)

(i) Assume P_n has a double zero in ξ, other single zeros $\xi_1, \dots, \xi_{n-2}$

Then qP_n with $q(t) := \prod_{\ell=1}^{n-2} (t-\xi_\ell)$ does not change sign in $[-1,1]$

(ii) Assume that P_n has only $m < n$ zeros in $[-1,1]$: $z_1, \dots, z_m \in]-1,1[$

$q(t) := \prod_{\ell=1}^{m} (t - z_\ell) \in \mathcal{P}_{n-1} \Rightarrow P_n q$ has fixed sign on $[-1,1]$

↳ changes sign exactly where P_n changes sign ! $\Rightarrow \int_{-1}^{1} P_n q\,dt \neq 0$ ↯

□

▷ Gauss - Legendre quadrature rules

Notation : ξ_j^n, $j = 1, \dots, n$ $\hat{=}$ zeros of P_n $\hat{=}$ Gauss nodes

Obviously ▷

Lemma 5.3.26. Positivity of Gauss-Legendre quadrature weights

The weights of Gauss-Legendre quadrature formulas are positive.

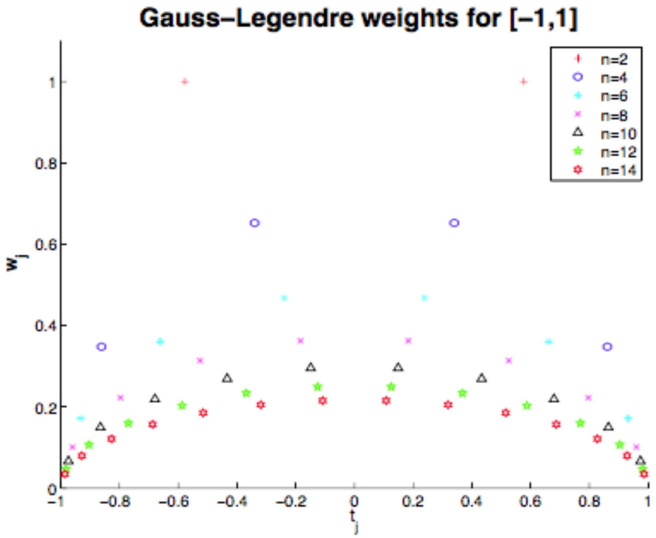


Fig. 171

Proof: Fix n, pick $j \in \{1,\dots,n\}$

$$q(t) = \prod_{\substack{\ell=1\\ \ell\neq j}}^{n} (t-\xi_\ell^n)^2 \in \mathcal{P}_{2n-2}$$

$$0 < \int_{-1}^{1} q(t)\,dt = \sum_{\ell=1}^{n} w_\ell\, q(\xi_\ell^n) = w_j \underbrace{q(\xi_j^n)}_{>0} \qquad \square$$

Order $2n$!

Computation of Gauss nodes:

3-recursion for Legendre polynomials:

$$\boxed{P_{n+1}(t) := \frac{2n+1}{n+1}\, t P_n(t) - \frac{n}{n+1} P_{n-1}(t)\ ,\quad P_0 := 1\ ,\quad P_1(t) := t}$$

▷ $\tilde{P}_n := \frac{1}{\sqrt{n+1/2}} P_n$ satisfy , $\tilde{P}_{-1} := 0$

$$t\tilde{P}_n(t) = \underbrace{\frac{n}{\sqrt{4n^2-1}}}_{=:\beta_n} \tilde{P}_{n-1}(t) + \underbrace{\frac{n+1}{\sqrt{4(n+1)^2-1}}}_{=:\beta_{n+1}} \tilde{P}_{n+1}(t)\ . \qquad (*)$$

fix $t \in \mathbb{R}$: $p(t) := [\tilde{P}_j(t)]_{j=0}^{n-1} \in \mathbb{R}^n$

$$t \underbrace{\begin{bmatrix} \tilde{P}_0(t) \\ \tilde{P}_1(t) \\ \vdots \\ \tilde{P}_{n-1}(t) \end{bmatrix}}_{=:\mathbf{p}(t)\in\mathbb{R}^n} = \underbrace{\begin{bmatrix} 0 & \beta_1 & & & & \\ \beta_1 & 0 & \beta_2 & & & \\ & \beta_2 & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & & 0 & \beta_{n-1} \\ & & & & \beta_{n-1} & 0 \end{bmatrix}}_{=:\mathbf{J}_n\in\mathbb{R}^{n,n}} \begin{bmatrix} \tilde{P}_0(t) \\ \tilde{P}_1(t) \\ \vdots \\ \tilde{P}_{n-1}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \beta_n \tilde{P}_n(t) \end{bmatrix}$$

If $\tilde{P}_n(\xi) = 0 \Rightarrow$ $\boxed{\xi\, p(\xi) = J_n\, p(\xi)}$

$\Leftrightarrow$ ξ is an eigenvalue of J_n

Effort $O(n^2)$
(= asymp. complexity of eig())

MATLAB-code 5.3.32: Golub-Welsch algorithm

```
function [x,w]=gaussquad(n)
% Computation of weights and nodes of n-point
% Gaussian quadrature rule on [-1,1].
if (n==1), x = 0; w = 2;
else
  b = zeros(n-1,1);
  for i=1:(n-1), b(i)=i/sqrt(4*i*i-1); end
  J=diag(b,-1)+diag(b,1); [ev,ew]=eig(J);
  x=diag(ew); w=(2*(ev(1,:).*ev(1,:)))';
end
```

Quadrature error $\leftrightarrow$ best approximation error

$\underbrace{\text{Q.R. on } [a,b] \text{ of order } q \geq 1}_{\text{by linearity}}$: $E_n(f) \triangleq$ quadrature error

$\hookrightarrow \quad E_n(f-p) = E_n(f) \qquad p \in \mathcal{P}_{q-1}$

$$E_n(f) = E_n(f-p) = \left| \int_a^b (f-p)(t)\,dt - \sum_{j=1}^{n} w_j (f-p)(c_j) \right|$$

$$\leq \int_a^b |f-p|(t)\,dt + \sum_{j=1}^{n} |w_j|\,|f-p|(c_j)$$

$$\leq \|f-p\|_{L^\infty([a,b])} \left(|b-a| + \sum_{j=1}^{n} |w_j| \right)$$

$\underbrace{\text{If } w_j \geq 0}$, use $\sum_{j=1}^{n} w_j = b-a$

$\hookrightarrow \quad E_n(f) \leq 2|b-a|\,\|f-p\|_{L^\infty} \quad \forall p \in \mathcal{P}_{q-1}$

Theorem 5.3.35. Quadrature error estimate for quadrature rules with positive weights

For every n-point quadrature rule Q_n as in (5.1.2) of order $q \in \mathbb{N}$ with weights $w_j \geq 0$, $j = 1,\dots,n$ the quadrature error satisfies

$$E_n(f) := \left| \int_a^b f(t)\,dt - Q_n(f) \right| \leq 2|b-a| \underbrace{\inf_{p \in \mathcal{P}_{q-1}} \|f-p\|_{L^\infty([a,b])}}_{\text{best approximation error}} \quad \forall f \in C^0([a,b]) \,. \qquad (5.3.36)$$

▷ Convergence analysis of G.-L.-Q.R.* through best approximation estimates for polynomials:

$f \in C^r \overset{\text{Thm. 4.1.11}}{\Longrightarrow} |E_n(f)| = O(n^{-r})$ [alg. cvg.]

f has analytic ext. to $\mathbb{C}$-neighborhood of $[a,b]$ $\Longrightarrow$ $|E(f)| = O(q^n)$, $0 \leq q < 1$ [exp. cvg.]

↑ Chebychev intp. est.

* $q = 2n$!

Example :

blow-up !

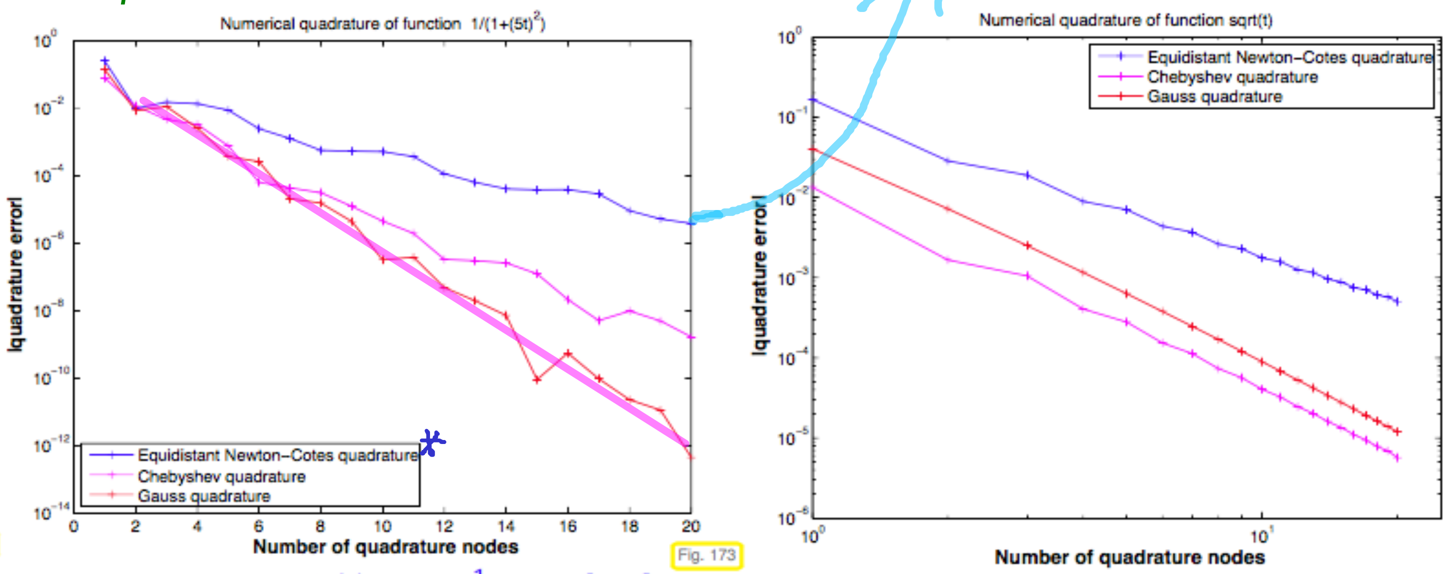


Fig. 173

quadrature error, $f_1(t) := \frac{1}{1+(5t)^2}$ on $[0,1]$ — quadrature error, $f_2(t) := \sqrt{t}$ on $[0,1]$

↓ has analytic extension — $f_2 \notin C^1([0,1])$

* negative weight — merely alg. cvg.

▷ Smoothness of integrand is crucial for fast cvg.

Smoothing integrands by transformation

Ex: $\displaystyle\int_0^b \underbrace{\sqrt{t}\, f(t)}\, dt$, where $f:[0,b]\to\mathbb{R}$ has analytic extension, given only in procedural form

↓

no-smooth → slow cvg. of G.-L.

Idea : Substitution $s := \sqrt{t} \Rightarrow \frac{ds}{dt} = \frac{1}{2\sqrt{t}} \Rightarrow dt = 2\sqrt{t}\,ds$

$$\int_0^b \sqrt{t}\, f(t)\, dt = \int_0^{\sqrt{b}} \underbrace{s\, f(s^2)\, 2s}_{\text{analytic integrand}}\, ds \overset{*}{=} \int_0^b \frac{2\tau^2}{b} f\left(\frac{\tau^2}{b}\right) \frac{1}{\sqrt{b}}\, d\tau$$

analytic integrand → Exp. cvg. of G.L.

* transformation $\tau = \sqrt{b}\, s$

G.L. q.r. on $[0,b]$: $Q_n^{GL}(f) \approx \sum_{j=1}^n \hat{w}_j f(\hat{c}_j)$

$$\int_0^b \ldots d\tau \approx \sum_{j=1}^n \hat{w}_j \frac{2}{b\sqrt{b}} \hat{c}_j^2 f\left(\frac{\hat{c}_j^2}{b}\right) = \sum_{j=1}^n w_j f(c_j)$$

with $w_j =$ ↓ , $c_j =$ ↓

The message of asymptotic convergence

Alg. cvg :

$E_n(f) = O(n^{-r}) \Rightarrow E_n(f) \approx C n^{-r}$ for n large

quadrature error ... Ass: estimate is sharp ... unknown

$\Rightarrow$ No information about $E_n(f)$ for given n

↳ tells us what additional effort ($\sim$ # f-eval) is needed to reduce quad. error by a factor of $\rho > 1$

$$\frac{C n_1^{-r}}{C n_0^{-r}} \approx \frac{1}{\rho} \Rightarrow n_1 : n_0 = \sqrt[r]{\rho}$$

Bigger rate r $\Rightarrow$ less additional effort

Measure for additional effort (asymptotically)

Exp. cvg :

$E_n(f) = O(q^n) \Rightarrow E_n(f) \approx C q^n$ for large n $[0 \le q < 1]$

We aim for error reduction by a factor of ρ

$$\frac{C q^{n_1}}{C q^{n_0}} \approx \frac{1}{\rho} \Rightarrow q^{n_1 - n_0} = \frac{1}{\rho} \Rightarrow n_1 = n_0 - \frac{\log \rho}{\log q}$$

→ Fixed additional no. of quad. nodes gain factor ρ in accuracy !

5.4. Composite quadrature

mesh $\mathcal{M} = \{a = x_0 < x_1 < \dots < x_m = b\}$

$$\int_a^b f(t)\,dt = \sum_{\ell=1}^{m} \underbrace{\int_{x_{\ell-1}}^{x_\ell} f(t)\,dt}$$

↳ apply q.r. here!

General construction of composite quadrature rules

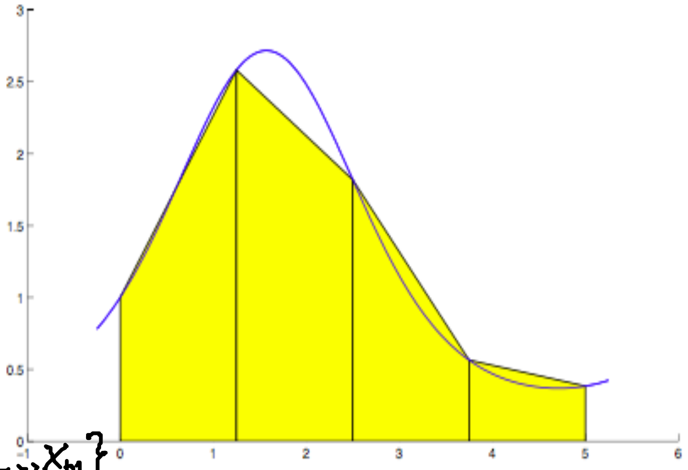
Idea:
- Partition integration domain $[a,b]$ by a mesh/grid (→ Section 4.5)
  $$\mathcal{M} := \{a = x_0 < x_1 < \ldots < x_m = b\}$$
- Apply quadrature formulas from Section 5.2, Section 5.3 locally on mesh intervals $I_j := [x_{j-1}, x_j]$, $j = 1, \ldots, m$, and sum up.

▶ composite quadrature rule

$$\#\{f\text{-eval}\} = \sum_{\ell=1}^{m} n_\ell\,, \quad \text{local } n_\ell\text{-pt. q.r.}$$

Composite trapezoidal rule, *cf.* (5.2.5)

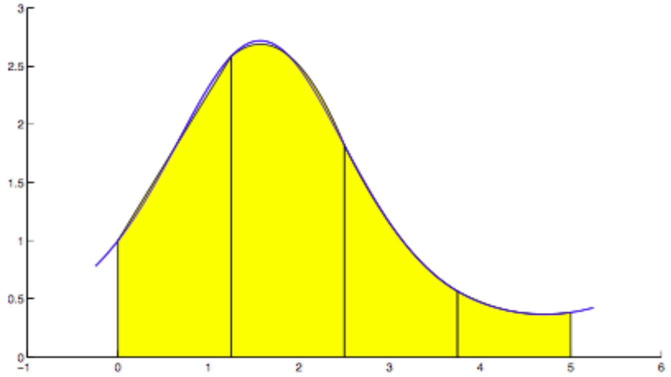
$$\int_a^b f(t)\mathrm{d}t = \tfrac{1}{2}(x_1 - x_0)f(a) + \sum_{j=1}^{m-1} \tfrac{1}{2}(x_{j+1} - x_{j-1})f(x_j) + \tfrac{1}{2}(x_m - x_{m-1})f(b)\,. \tag{5.4.4}$$



≙ integrate linear interpolant for nodes $\{x_0, \dots, x_m\}$

Composite Simpson rule, *cf.* (5.2.6)

$$\int_a^b f(t)\mathrm{d}t = \tfrac{1}{6}(x_1 - x_0)f(a) + \sum_{j=1}^{m-1} \tfrac{1}{6}(x_{j+1} - x_{j-1})f(x_j) + \sum_{j=1}^{m} \tfrac{2}{3}(x_j - x_{j-1})f(\tfrac{1}{2}(x_j + x_{j-1})) + \tfrac{1}{6}(x_m - x_{m-1})f(b)\,. \tag{5.4.5}$$



Special case: all local quadrature rules from a single q.r. on reference interval by affine transformation.

Quadrature error estimates by adding local error contributions

$$f \in C^r([a,b]) \Rightarrow \left| \int_{x_{j-1}}^{x_j} f(t)\,\mathrm{d}t - \underbrace{Q^j_{n_j}(f)}_{\text{local q.r. in } [x_{j-1},x_j]} \right| \le \underbrace{C}_{\text{does not depend on } f,\, j} |\underbrace{x_j - x_{j-1}}_{=: h_j}|^{\overbrace{\min\{r,q_j\}}^{\text{local order}}+1} \left\| f^{(\min\{r,q_j\})} \right\|_{L^\infty([x_{j-1},x_j])}$$

If $q_j = q$ for all j

$$\Rightarrow \left| \int_{x_0}^{x_m} f(t)\,\mathrm{d}t - Q(f) \right| \le C h_{\mathcal{M}}^{\min\{q,r\}} |b-a| \left\| f^{(\min\{q,r\})} \right\|_{L^\infty([a,b])}\,, \qquad (*)$$

$h_{\mathcal{M}} := \max_j |x_j - x_{j-1}|$ meshwidth

11

by $\displaystyle\sum_{j=1}^{m} C h_j^{\min(r,q)+1} \|f^{(\cdot\cdot)}\|_{L^\infty([x_{j-1},x_j])}$

$\displaystyle\le C \|f\|_{L^\infty([a,b])}\, h^{\min(r,q)} \sum_j h_j$

(*) $\Rightarrow$ [n_j the same for all j]

$E_n(f) = O(n^{-\min(r,q)}) = O(h_M^{\min(r,q)})$

For $n \to \infty$ ($n \,\hat{=}\, \#\{f\text{-eval}\}$)

$\Rightarrow$ alg. cvg. with rate $\min\{r,q\}$ (as $h_M \to 0$)

Letting h_M & fixed local q.r. h-convergence

Comparison: Composite quad. $\longleftrightarrow$ G.L. quad.

$\downarrow$ fixed local q.r., order q, equidistant mesh

$f \in C^r$: $E_n^{CR}(f) \le C n^{-\min\{q,r\}}$ for large n

$E_n^{GL}(f) \le C n^{-r}$

$\Rightarrow$ In terms of rate: G.L. is at least as good as C.R.
G.L. will "auto-detect" best possible rate!

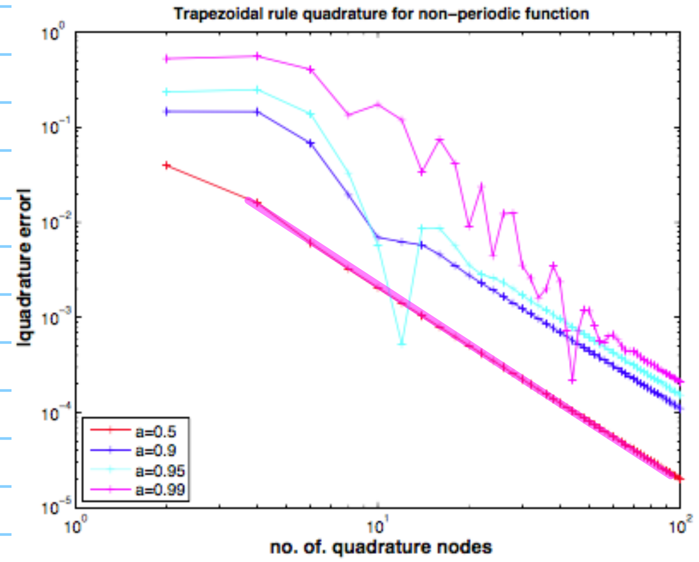
& if f analytic then exp. cvg. of G.L.

$\uparrow$ the clear winner

Equidistant trapezoidal rule

$$\int_a^b f(t)\,dt \approx T_m(f) := h\left(\tfrac{1}{2}f(a) + \sum_{k=1}^{m-1} f(kh) + \tfrac{1}{2}f(b)\right), \quad h := \frac{b-a}{m}.$$

Exp: $f(t) = \dfrac{1}{\sqrt{1 - a\sin(2\pi(t-1))}}$, $0 < a < 1$ — 1-periodic, C^∞



quadrature error for $T_n(f)$ on $[0, \frac{1}{2}]$

Alg. cvg. order 2

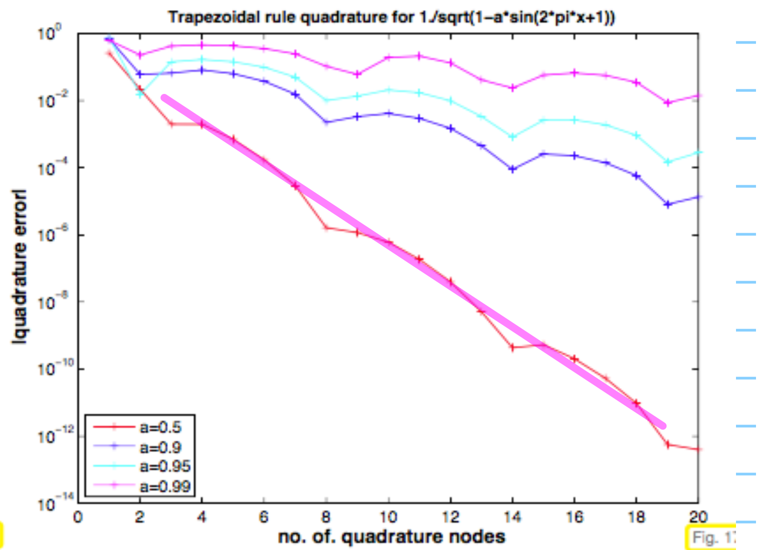
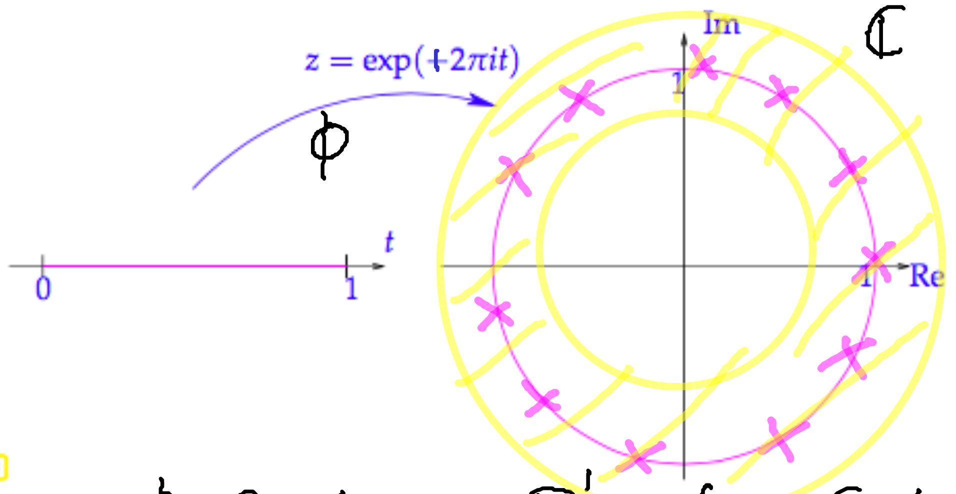


Fig. 1

quadrature error for $T_n(f)$ on $[0, 1]$

Exp. cvg.!



p. 151

$\Phi : [0,1[\;\longrightarrow\; \mathbb{S}^1 := \{ z \in \mathbb{C} : |z| = 1 \}$

Quadrature rule on $\mathbb{S}$

no path integral

$$Q_n^{\mathbb{S}}(g) = \sum_{j=1}^{n} w_j^{\mathbb{S}}\, g(z_j^{\mathbb{S}}) = \int_{\mathbb{S}} (L_z g)(\tau)\, dS(\tau)$$

↳ induced by equidistant Lagrange interpolation on $\mathbb{S}$

↳ nodes $z_j = \exp(2\pi i \frac{j}{n})$, $j = 0, \dots, n-1$

New: polynomial interpolation with complex nodes

⟶ same theory

Weights from perfect symmetry of nodes: $w_j^{\mathbb{S}} = \frac{2\pi}{n}$

$$\int_0^1 f(t)\,dt = \frac{1}{2\pi} \int_{\mathbb{S}} ((\Phi^{-1})^* f)(\tau)\, d\tau \approx \frac{1}{2\pi} \frac{2\pi}{n} \sum_{j=0}^{n-1} ((\Phi^{-1})^* f)(\exp(2\pi i \tfrac{j}{n}))$$

$$= \frac{1}{n} \sum_{j=0}^{n-1} f(\Phi^{-1}(\Phi(j/n))) = \frac{1}{n} \sum_{j=0}^{n-1} f(j/n)$$

= trapezoidal rule, if f 1-periodic!

▷ Equidistant T.R. $\hat{=}$ global polynomial quadrat. on $\mathbb{S}^1$

⇓

Error ~ approximation error of equidistant Lagr. interpolation on $\mathbb{S}^1$

⇓

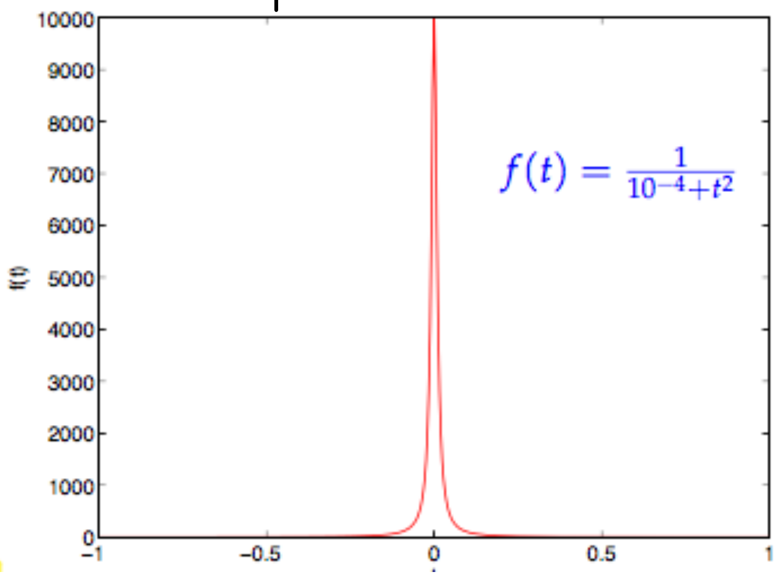
Exp. cvg. in n if $(\Phi^{-1})^* f$ analytic ext.

5.5. Adaptive Quadrature

→ rehabilitation of composite quadrature

So far: on fixed meshes (independent of f)

Example: "peak function"



◁ f flat in large parts of $[a,b]$

→ For C.Q.: use small cells close to 0, large cells far away from zero

Local quadrature estimate for trapezoidal rule

$$\left| \int_{x_{k-1}}^{x_k} f(t)\,dt - \tfrac{1}{2}h_k\left(f(x_{k-1})+f(x_k)\right) \right| \le \underbrace{\tfrac{1}{2}\|f''\|_{L^\infty([x_{k-1},x_k])}}_{=:\,B_k} \cdot h_k^3$$

B_k depends on f

▷ Bound for quadrature error: $E(f) \le \underbrace{\sum_k B_k h_k^3}$

Goal: Minimize this under constraint $\sum_k h_k = b-a$

Focus on two cells: minimize their combined error contribution

$$B_k(h_k+\delta)^3 + B_\ell(h_\ell-\delta)^3 \longrightarrow \min$$

Minimizer δ^* satisfies: $B_k(h_k+\delta^*)^3 = B_\ell(h_\ell-\delta^*)^3$

→ ⊕ Optimal cell size distribution: all cells make the same contribution to the error bound.

Error equidistribution principle

The mesh for *a posteriori adaptive composite numerical quadrature* should be chosen to achieve equal contributions of all mesh intervals to the quadrature error

→ This mesh will depend on f: "adapted to f"

Adaptive numerical quadrature

The policy of adaptive quadrature approximates $\int_a^b f(t)\,dt$ by a quadrature formula (5.1.2), whose nodes c_j^n are chosen depending on the integrand f.

We distinguish

(I) a priori adaptive quadrature: the nodes a *fixed* before the evaluation of the quadrature formula, taking into account external information about f, and

(II) a posteriori adaptive quadrature: the node positions are chosen or improved based on information gleaned *during the computation* inside a loop. It terminates when sufficient accuracy has been reached.

iterative improvement of mesh

move nodes

add nodes

"mesh refinement"

(14)

Adaptation loop for numerical quadrature

(1) ESTIMATE: based on available information compute an approximation for the quadrature error on every mesh interval.
(2) CHECK TERMINATION: if total error sufficient small → **STOP**
(3) MARK: single out mesh intervals with the largest or above average error contributions.
(4) REFINE: add node(s) inside the marked mesh intervals.

MATLAB-code 5.5.15: h-adaptive numerical quadrature

```
function I = adaptquad(f,M,rtol,abstol)
% adaptive numerical quadrature:  f is a function handle to integrand
h = diff(M);                       % distances of quadrature nodes
mp = 0.5*(M(1:end-1)+M(2:end));    % positions of midpoints
fx = f(M); fm = f(mp);             %
trp_loc  = h.*(fx(1:end-1)+2*fm+fx(2:end))/4; % trapezoidal rule (5.4.4)
simp_loc = h.*(fx(1:end-1)+4*fm+fx(2:end))/6; % Simpson rule (5.4.5)
I = sum(simp_loc);                 % Simpson approximation of integral value
est_loc  = abs(simp_loc -trp_loc);   % local error estimate (5.5.11)
err_tot  = sum(est_loc);             % estimate for quadrature error
% Termination based on (5.5.12)
if ((err_tot > rtol*abs(I)) && (err_tot > abstol)) %
  refcells = find(est_loc > 0.9*sum(est_loc)/length(h));
  I = adaptquad(f,sort([M,mp(refcells)]),rtol,abstol); %
end
```

Order 2 (trapezoidal rule)
Order 4 (Simpson rule)
→ TERMINATION
→ MARKING*
→ REFINE+

ESTIMATE:

Idea: local error estimation by comparing local results of two quadrature formulas Q_1, Q_2 of *different* order → local error estimates

heuristics: $\text{error}(Q_2) \ll \text{error}(Q_1) \Rightarrow \text{error}(Q_1) \approx Q_2(f) - Q_1(f)$.

Here: Q_1 = trapezoidal rule (order 2) ↔ Q_2 = Simpson rule (order 4)

Above: $Q_1 \Leftrightarrow$ trapezoidal rule $O(h_k^3)$, $Q_2 \Leftrightarrow$ Simpson rule $O(h_k^5)$

NCSE15

Objection: We estimate the error for trapezoidal rule, but use Simpson rule for quadrature!

→ est_loc still useful for refinement

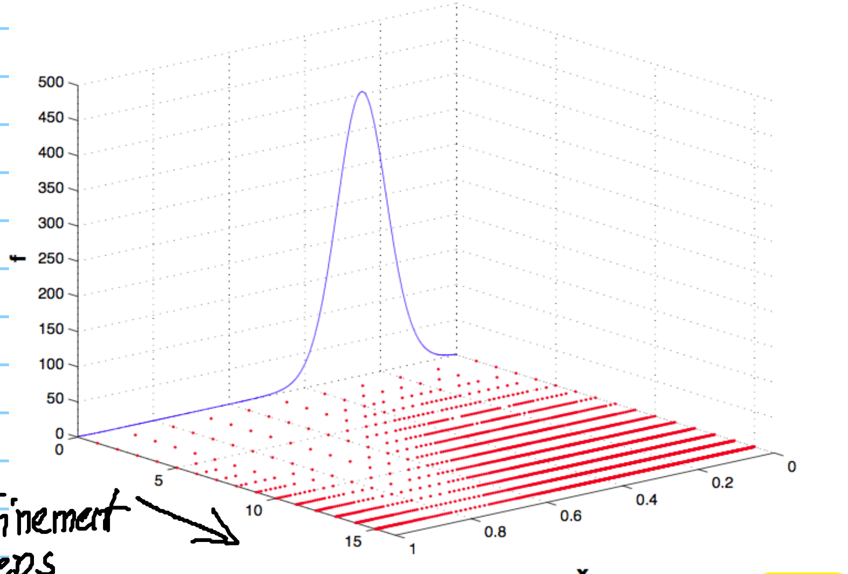
→ err_tot is a very crude upper bound for the quadrature error

* Marked intervals: $\mathcal{S} := \{k \in \{1,\dots,m\}: \text{EST}_k \geq \eta \cdot \frac{1}{m}\sum_{j=1}^{m} \text{EST}_j\}$, $\eta \approx 0.9$.

+ REFINE: new mesh: $\mathcal{M}^* := \mathcal{M} \cup \{p_k := \frac{1}{2}(x_{k-1}+x_k): k \in \mathcal{S}\}$.

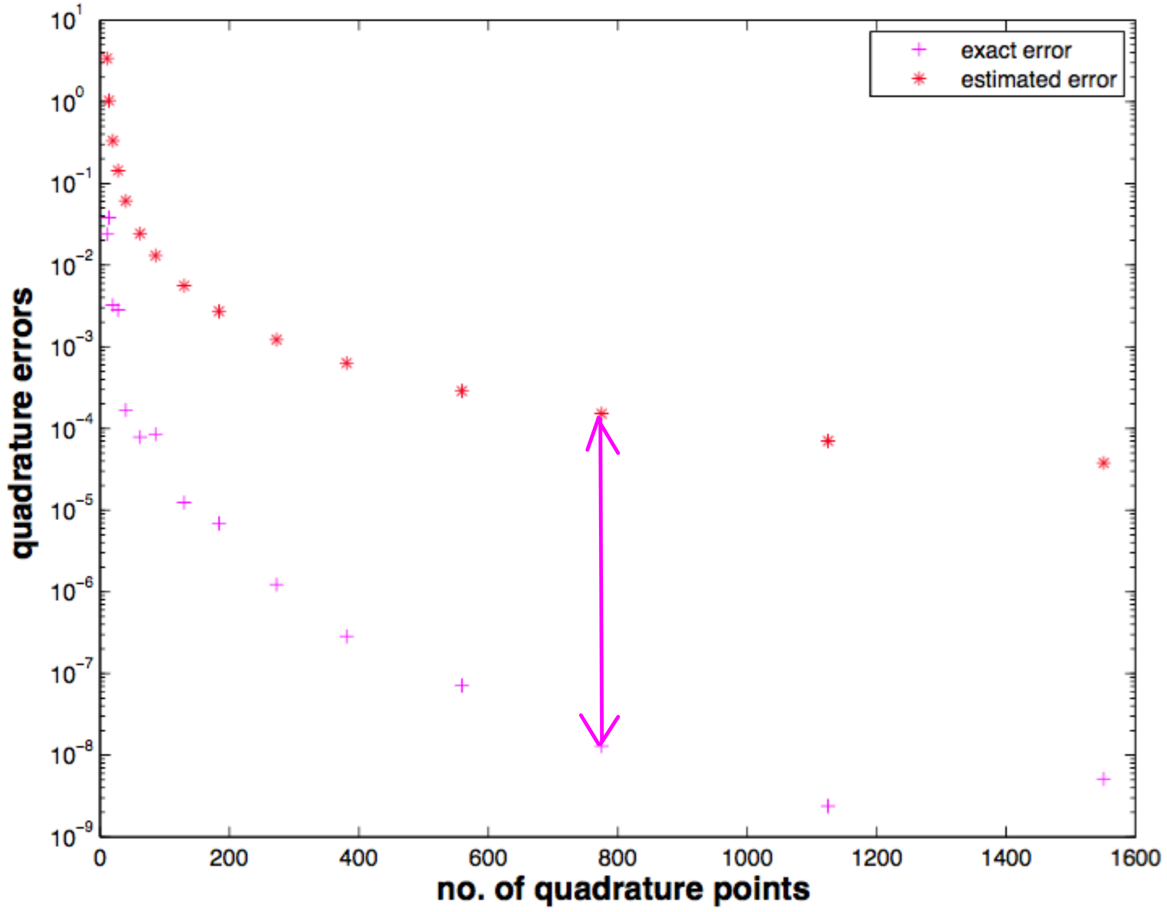
Example:

① $f(t) = e^{6\sin(2\pi t)}$ on $[0,1]$



refinement steps

Fig. 183



Gross overestimation of error by err_tot

→ termination at least reliable (maybe not efficient)