07.03.2017

Computational methods for Integral Equations

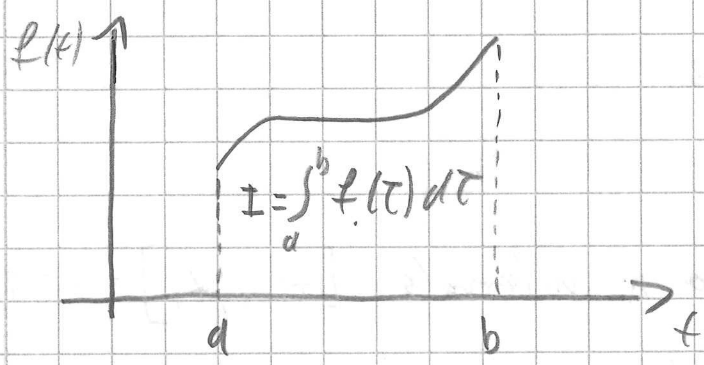
I Numerical Integration

We seek to compute the Riemann Integral

$$I(f) := I_a^b(f) := \int_a^b f(t)\,dt$$

Where f is ~~piecewise~~ continuous on $[a,b] \subset \mathbb{R}$.

$$f \in C([a,b])$$



The procedure of ~~numerical~~ integration is refered to as <u>numerical quadrature</u>.

1) Quadrature formulas

Let $I_a^b = I : C([a,b]) \longrightarrow \mathbb{R}$, $f \longmapsto \int_a^b f(x)\,dx$, with $a < b$.

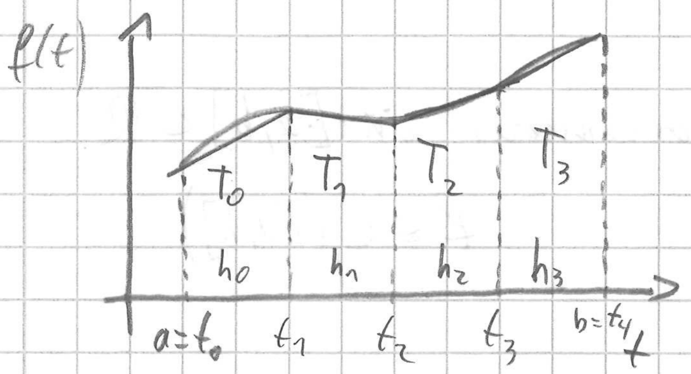
I is linear and positive, i.e.

$$f \geq 0 \Rightarrow I(f) \geq 0.$$

In the following we construct a positive linear functional $\hat{I} : C([a,b]) \to \mathbb{R}$, $f \mapsto \hat{I}(f)$

such that $\hat{I}(f) - I(f)$ is "small".

Example: The trapezoidal rule.



We divide $[a,b]$ into n intervals $[t_{i-1}, t_i]$, $i = 1, \dots n$, with length $h_i := t_i - t_{i-1}$ and

$$a = t_0 < t_1 \dots < t_n = b.$$

We approximate $I(f)$ by the sum

$$T^{(n)} := \sum_{i=1}^{n} T_i \quad , \quad T_i = \frac{h_i}{2}\left(f(t_{i-1}) + f(t_i) \right).$$

Compare to the Riemann sums (upper and lower)

$$R_u^{(n)} = \sum_{i=1}^{n} h_i \min_{t \in [t_{i-1}, t_i]} f(t) \quad \text{and}$$

$$R_o^{(n)} = \sum_{i=1}^{n} h_i \max_{t \in [t_{i-1}, t_i]} f(t) .$$

Then $R_u^{(n)} \le T^{(n)} \le R_o^{(n)}$, convergence of $T^{(n)}$ follows for $f \in C([a,b])$ and $n \to \infty$.

Definition 1

We denote the sum
$$\hat{I}(f) = (b-a)\sum_{i=0}^{n} \lambda_i f(t_i)$$
for computing an integral as <u>quadrature formula</u>.

With the nodes $t_0, \dots, t_n$ and weigths $\lambda_0, \dots, \lambda_n$ such that $\sum_{i=0}^{n} \lambda_i = 1$.

Note that $\hat{I}$ positive $\Leftrightarrow \lambda_i \geq 0 \quad \forall i = 0, \dots, n$

2. Newton-Cotes formulas

We first replace f by an approximation $\hat{f}$ and seek to find $\hat{I}$, such that
$$\hat{I}(f) = I(\hat{f}).$$

Given ~~let~~ ~~for~~ $t_0, \dots, t_n$, then we set the interpolation polynomial
$$\hat{f}(t) := \sum_{i=0}^{n} f(t_i) L_{i,n}(t).$$

Where $L_{i,n} \in \mathbb{P}_n$ is the ith Lagrange polynomial for the nodes t_j, i.e. $L_{i,n}(t_j) = \delta_{i,j}$

$\mathbb{P}_n := \{ p : [a,b] \to \mathbb{R} : p(x) = \sum_{i=0}^{n} a_i x^i, \ a_i \in \mathbb{R} \ \forall i = 0, \dots, n \}$

Space of polynomials of order n

Then it follows

$$\hat{I}(f) = (b-a) \sum_{i=0}^{n} \lambda_{i,n} f(t_i)$$

with

$$\lambda_{i,n} = \frac{1}{b-a} \int_a^b L_{i,n}(t)\, dt$$ depending only on $t_0, \dots, t_n$

Lemma 2

Given $n+1$ pairwise different nodes $t_0, \dots, t_n$, then there is exactly one quadrature formula

$$\hat{I}(f) = (b-a) \sum_{i=0}^{n} \lambda_i f(t_i) \qquad (*)$$

that is exact for all polynomials $p \in \mathbb{P}_n$

Proof:

For the Lagrange polynomials $L_{i,n} \in \mathbb{P}_n$ given for the nodes t_i we have by definition

optional

$$I(L_{i,n}) = \hat{I}(L_{i,n}) = (b-a) \sum_{j=0}^{n} \lambda_{j,n} L_{i,n}(t_j)$$

$$((*) \Rightarrow) \qquad = (b-a) \sum_{j=0}^{n} \lambda_j \delta_{i,j} = (b-a) \lambda_j$$

$$\Rightarrow \lambda_i = I(L_{i,n}) / (b-a) = \lambda_{i,n}$$

$\square$

Comp. meth. Int. eq

For the special case of equidistant nodes,

$$h_i = h = \frac{b-a}{n}, \quad t_i = a + i \cdot h, \quad i = 0, \dots, n$$

we get the Newton-Cotes formulas.

The weight can be computed independent of the interval by substituting $s := \frac{(t-a)}{h}$ as follows

$$\lambda_{i,n} = \frac{1}{b-a} \int_a^b \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{t-t_j}{t_i - t_j} = \frac{1}{n} \int_0^n \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{s-j}{i-j} \, ds$$

We get the following common formulas

n	$\lambda_{0,n} \dots \lambda_{n,n}$	Error	Name
1	$\frac{1}{2} \quad \frac{1}{2}$	$\frac{h^3}{12} f''(\tau)$	Trapezoidal rule
2	$\frac{1}{6} \quad \frac{4}{6} \quad \frac{1}{6}$	$\frac{h^5}{90} f^{(4)}(\tau)$	Simpson's rule
3	$\frac{1}{8} \quad \frac{3}{8} \quad \frac{3}{8} \quad \frac{1}{8}$	$\frac{3h^5}{80} f^{(4)}(\tau)$	Simpsons 3/8 rule
4	$\frac{7}{90} \quad \frac{32}{90} \quad \frac{12}{90} \quad \frac{32}{90} \quad \frac{7}{90}$	$\frac{8h^7}{945} f^{(6)}(\tau)$	Boole's rule

optional

3. Gaussian quadrature (short)

Compute $I(f) := \int_a^b w(t) f(t)\, dt$ with a positive weight function $w(t)$.

~~We~~ Construct quadrature formula that is exact; i.e.

$$\hat{I}_n(p) = I(p), \text{ for } p \in \mathbb{P}_{2n+1}$$

$\Rightarrow$ nodes x_i need to be chosen specifically

Gauss-Legendre: $w(x) = 1$

$[a,b] = [-1, 1]$

Chebyshev-Gauss: $w(x) = \frac{1}{\sqrt{1-x^2}}$

Interval: $(-1, 1)$

Gauss-Hermite: $\cancel{e^{-x^2}}$ $w(x) = e^{-x^2}$

Interval: $(-\infty, \infty)$

The approximation error is generally given by

$$\int_a^b wf - \hat{I}_n(f) = \frac{f^{(2n+2)}(\tau)}{(2n+2)!} (P_{n+1}, P_{n+1})$$

P_n is the monic orthogonal polynomial (associated with w) of order n.

II The Nyström method

We want to solve the Fredholm integral equation of the second kind,

$$(1)\quad \lambda x(t) - \int_a^b K(t,s)\,x(s)\,ds = y(t), \quad t \in [a,b]$$

by ~~numer~~ applying numerical quadrature.

Let $K(t,s)$ be continuous for all $t,s \in [a,b]$.

First divide $[a,b]$ into n intervals:

~~$t_0 = a, t_1, t_n = b$~~ $a = t_0, t_1, \dots, t_n = b$

~~$s_0 = a, s_0, s_n = b$~~ $a = s_0, s_1, \dots, s_n = b$ (with $t_0 = s_0$, $t_1 = s_1$, $t_n = s_n$)

Then we discretize the integral in (1) and get

$$\lambda x(t) - \sum_{j=0}^{n} w_j K(t, s_j)\, x(s_j) = y(t).$$

Second we divide this into $n+1$ equations for $t_0, \dots, t_n$

$$(2)\quad \Rightarrow \lambda x(t_i) - \sum_{j=0}^{n} w_j K(t_i, s_j)\, x(s_j) = y(t_i), \quad i = 0, \dots, n$$

$\Rightarrow$ $n+1$ linear equations for $n+1$ unknown.

Compute $\bar{x} = [x(t_0), x(t_1), \dots, x(t_n)]$

Originally this is understood as an interpolation.

~~The Define~~

Given a solution $\bar{z}_n$ to (2), then define

$$z(t) = \frac{1}{\lambda}\left(y(t) + \sum_{j=0}^{n} w_j K(t, s_j) z_j\right),$$

~~with~~ which is an interpolation formula at the node points t_j

1. Convergence of the Nyström method

optional The error is governed by the integration error of the chosen quadrature formula and can be expressed as

$$\|(\mathcal{K} - \mathcal{K}_n)x\|_\infty = \max_{t \in [a,b]} \left| \int_a^b K(t,s)\, x(s)\, ds - \sum_{j=0}^{n} w_j K(t, s_j) x(s_j) \right|$$

Examples

For the trapezoidal rule we have

$$(\mathcal{K} - \mathcal{K}_n) x(s) = -\frac{(b-a)^3}{12} \left(\frac{\partial^2 K(t,\xi)\, x(\xi)}{\partial s^2} \right), \quad \tau \in [a,b]$$

2. Algorithmic implementation

We want to represent (1) ~~in~~ as discrete matrix vector equation:

(3) $(\lambda I - \bar{K})\bar{x} = \bar{y}$, with I the ~~unit~~ identity matrix.

and $\bar{K} \in \mathbb{R}^{(n+1)\times(n+1)}$, $\bar{x} \in \mathbb{R}^{n+1}$, $\bar{y} \in \mathbb{R}^{n+1}$.

The main task is to build $\bar{K}$ as follows.

1.) Set value of n

2.) Partition interval $[a,b]$ into n Intervals, compute ω.

3.) for $i = 0:n$

4.) for $j = 0:n$

5.) Set $\bar{K}_{i+1,j+1} = K(t_i, s_j)\cdot\omega_j$

~~Set y~~

6.) end

7.) Set $y_j = y(t_i)$

8.) end

And we are left to solve the matrix vector equation (3). In MATLAB by

$$\bar{x} = (\lambda I - \bar{K}) \setminus \bar{y}$$