Maria Cameron

1. Gaussian quadrature

The key idea behind the Gaussian quadrature is a clever choice of interpolation nodes in $[a, b]$ that maximizes the degree of exactness of the quadrature rule. Gaussian quadrature is designed for computing integrals of the form

$$I(f) = \int_a^b f(x)w(x)dx. \tag{1}$$

Here $w(x)$ is a weight function on $[a, b]$ which means that it is nonnegative in any open interval in $[a, b]$ ($a = -\infty$ and $b = \infty$ are accepted) and

$$\int_a^b w(x)|x|^n dx < \infty, \quad , n = 0, 1, 2, \dots.$$

1.1. Maximizing the degree of exactness.

As soon as the interpolation nodes $x_1 < x_2 < \dots x_n$, $x_j \in [a, b]$, $j = 1, 2, \dots, n$ are picked one can approximate the integral using the standard recipe for interpolating quadrature rules:

$$I(f) = \int_a^b f(x)w(x)dx \approx Q(f) = \int_a^b w(x) \sum_{j=1}^n f(x_j)L_j(x) = \sum_{j=1}^n w_j f(x_j) \tag{2}$$

where

$$w_j = \int_a^b L_j(x)w(x)dx, \quad L_j(x) = \prod_{k=1,\ k\neq j}^n \frac{(x - x_k)}{(x_j - x_k)}. \tag{3}$$

Independent of the choice of nodes such a rule is exact for polynomials $f(x)$ of degree $\leq\ n - 1$.

Definition 1. *We say a quadrature rule of the form of Eq.* (2) *has degree of exactness m if it is exact for all polynomials $f(x)$ of degree $\leq\ m$ but not exact for all polynomials of degree $m + 1$.*

The quadrature rule of the form (2) has $2n$ parameters: n nodes and n weights. Hence we can hope to make it exact for all polynomials of degree $2n - 1$ that have $2n$ coefficients. The nodes and weights for such a rule will be the solution of the system of $2n - 1$ equations

$$\sum_{j=1}^n w_j x_j^k = \int_a^b x_k w(x)dx, \quad k = 0, 1, \dots 2n - 1.$$

However, this system is hard to solve. It is nonlinear and badly conditioned. Therefore, this approach is not practical. A much better idea is to generate a set of orthogonal polynomials $\{p_k(x)\}_{k=0}^n$ on $[a, b]$ with respect to the weight function $w(x)$ using the three-term recurrence relations (TTRR). Then nodes coinciding with the zeros of the polynomial $p_n(x)$ for an interpolatory quadrature rule will render it exact for all polynomials of degree up to $2n - 1$.

Theorem 1. *Let $\{p_k(x)\}_{k=0}^n$ be a set of orthogonal polynomials on $[a,b]$ with respect to the inner product*

$$(f,g) = \int_a^b f(x)g(x)w(x)dx.$$

Let x_j, $j = 1, 2, \ldots, n$ be zeros of the polynomial $p_n(x)$. Then the quadrature rule given by Eqs. (2) and (3) has degree of exactness $2n-1$.

Proof. The key idea in the proof of this fact is division of the polynomial $f(x)$ by the polynomial $p_n(x)$. Let $f(x)$ be a polynomial of degree $\leq 2n-1$. Then

$$f(x) = p_n(x)q(x) + r(x), \quad \text{where} \quad q(x), r(x) \in \mathbb{P}_{n-1},$$

i.e, $q(x)$ and $r(x)$ are polynomials of degree $\leq n-1$. Since the rule (2), (3) is interpolatory, it is exact for $r(x)$. Then we have

$$I(f) = I(qp_n + r) = \int_a^b q(x)p_n(x)w(x)dx + \int_a^b r(x)w(x)dx = \int_a^b r(x)w(x)dx = Q(r).$$

Here we have used the fact that the polynomial $p_n(x)$ is orthogonal to all polynomials of degree $\leq n-1$. We continue:

$$I(f) = Q(r) = \sum_{j=1}^n w_j r(x_j) = \sum_{j=1}^n w_j (p_n(x_j)q(x_j) + r(x_j)) = Q(f).$$

Here we have used the fact that x_j's are the zeros of $p_n(x)$.

It remains to prove that the rule (2), (3) is not exact for all polynomials of degree $2n$. Let us make the polynomial $p_n(x)$ of norm 1. Let $f(x)$ be a polynomial of degree $2n$. Then $q(x)$ is of degree n and $r(x)$ is of degree $\leq n-1$. On one hand,

$$I(f) = I(qp_n + r) = \int_a^b q(x)p_n(x)w(x)dx + \int_a^b r(x)w(x)dx = Q(r) + (q, p_n).$$

Note that $(q, p_n) \neq 0$. On the other hand,

$$Q(f) = Q(p_n q + r) = Q(r),$$

as x_j's are the zeros of $p_n(x)$. Hence $I(f) \neq Q(f)$ for all polynomials f of degree $2n$. □

1.2. **Error estimate.** Next we will obtain an error estimate for the Gaussian quadrature.

Theorem 2. *Let $w(x)$ be a weight function and $p_n(x)$ be the monic polynomial of degree n orthogonal to all polynomials of smaller degrees. Let x_j, $j = 1, 2, \ldots, n$ be zeros of $p_n(x)$. Let $Q(f)$ be the quadrature rule defined in Eqs. (2) and (3). Suppose $f \in C^{2n}[a,b]$. Then one can find $\lambda \in (a,b)$ such that*

$$\int_a^b f(x)w(x)dx = Q(f) + \gamma_n \frac{f^{(2n)}(\lambda)}{(2n)!}, \quad \text{where} \quad \gamma_n = \int_a^b p_n^2(x)w(x)dx. \tag{4}$$

Proof. We use the Hermite interpolation to prove the error estimate. There exists a unique polynomial $h_{2n-1}(x)$ of degree $2n-1$ such that

$$h_{2n-1}(x_j) = f(x_j) \quad \text{and} \quad h'_{2n-1}(x_j) = f'(x_j).$$

In addition, there exists $\zeta \in (a,b)$ depending of x such that

$$f(x) = h_{2n-1}(x) + \frac{f^{(2n)}(\zeta)}{(2n)!}(x-x_1)^2 \dots (x-x_n)^2.$$

Note that $(x-x_1)\dots(x-x_n) = p_n(x)$ as p_n is monic and x_j, $j = 1,\dots,n$ are its roots. Therefore

$$\int_a^b f(x)w(x)dx = \int_a^b h_{2n-1}(x)w(x)dx + \int_a^b \frac{f^{(2n)}(\zeta)}{(2n)!}p_n^2(x)w(x)dx.$$

Since the quadrature is exact for polynomials of degree $2n-1$, it is exact for $h_{2n-1}(x)$. Hence

$$\int_a^b h_{2n-1}(x)w(x)dx = Q(h_{2n-1}) = \sum_{j=1}^n h_{2n-1}(x_j)w_j = \sum_{j=1}^n f(x_j)w_j = Q(f).$$

On the other hand, because $p_n^2(x)w(x) \geq 0$ on $[a,b]$ we can apply the mean value theorem and get

$$\int_a^b \frac{f^{(2n)}(\zeta)}{(2n)!}p_n^2(x)dx = \frac{f^{(2n)}(\lambda)}{(2n)!}\int_a^b p_n^2(x)w(x)dx$$

for some $\lambda \in (a,b)$. This completes the proof. □

1.3. **TTRR.** As we know from the beginning of the course, given a weight function $w(x)$ and an interval $[a,b]$, the set of monic orthogonal polynomials $\{p_k\}$ associated with the inner product

$$(f,g) = \int_a^b f(x)g(x)w(x)dx$$

satisfies the TTRR

$$\begin{aligned} p_0(x) &= 1, \\ p_1(x) &= (x-B_0)p_0(x), \\ p_{k+1}(x) &= (x-B_k)p_k(x) - A_k p_{k-1}(x), \quad k = 1,2,\dots, \quad \text{where} \\ A_k &= \frac{\|p_k\|^2}{\|p_{k-1}\|^2},\ k \geq 1, \quad B_k = \frac{(xp_k,p_k)}{\|p_k\|^2},\ k \geq 0. \end{aligned} \tag{5}$$

For the orthonormal polynomials $\{P_n\}$ the TTRR (5) can be rewritten as

$$\alpha_1 P_1(x) + \beta_0 P_0(x) = xP_0(x)$$

$$\alpha_{k+1}P_{k+1}(x) + \beta_k P_k(x) + \alpha_k P_{k-1}(x) = xP_k(x),\ k = 1, 2, \ldots, \text{ with} \tag{6}$$

$$\alpha_k = \sqrt{A_k} = \frac{\|p_k\|}{\|p_{k-1}\|} = \frac{a_{k-1}}{a_k},\ k \geq 1,$$

$$\beta_k = B_k = (xP_k, P_k),\ k \geq 0, \tag{7}$$

where a_k's are the leading coefficients of P_k's, since

$$F_k = \frac{1}{\|p_k\|}.$$

Exercise Prove TTRR (6).

Unfortunately, the integrals in TTRR (5) are the very kind of integrals we would like to compute using the Gaussian quadrature. However, there are three classic cases where the coefficients are analytic. These cases are listed below.

1.3.1. *Jacobi Polynomials.* Notation: $P_n^{(\alpha,\beta)}(x)$.
Interval: $[-1, 1]$.
Weight function: $w(x) = (1-x)^\alpha(1+x)^\beta,\ \alpha, \beta > -1$.
TTRR:

$$a_0 P_1(x) + b_0 P_0(x) = xP_0(x),$$
$$a_n P_{n+1}(x) + b_n P_n(x) + c_n P_{n-1}(x) = xP_n(x),$$

where

$$a_0 = \frac{2}{\alpha+\beta+2}, \quad b_0 = \frac{\beta-\alpha}{\alpha+\beta+2},$$
$$a_n = \frac{2(n+1)(n+\alpha+\beta+1)}{(L_n+1)(L_n+2)},$$
$$b_n = \frac{\beta^2-\alpha^2}{L_n(L_n+2)},$$
$$c_n = \frac{2(n+\alpha)(n+\beta)}{L_n(L_n+1)}, \quad n \geq 1,$$
$$L_n = 2n+\alpha+\beta.$$

1.3.2. *Generalized Laguerre polynomials.* Notation: $L_n^{(\alpha)}(x)$.
Interval: $[0, \infty)$.
Weight function: $w(x) = x^\alpha e^{-x},\ \alpha > -1$.

TTRR:

$$-L_1^{(\alpha)}(x) + (\alpha+1)L_0^{(\alpha)}(x) = xL_0^{(\alpha)}(x),$$
$$-(n+1)L_{n+1}^{(\alpha)}(x) + (2n+\alpha+1)L_n^{(\alpha)}(x) - (n+\alpha)L_{n-1}^{(\alpha)}(x) = xL_n^{(\alpha)}(x).$$

1.3.3. *Hermite polynomials.* Notation: $H_n(x)$.

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

Weight function: $w(x) = e^{-x^2}$.

TTRR:

$$\tfrac{1}{2}H_1(x) = xH_0(x),$$
$$\tfrac{1}{2}H_{n+1}(x) + nH_{n-1}(x) = xH_n(x).$$

1.4. **Golub-Welsch algorithm for finding nodes and weights.** The starting point for computing nodes and weights of a Gaussian n-point rule is TTRR (6). Let x_j be the one of the nodes of the rule. Then

$$\begin{aligned}
&\alpha_1 P_1(x_j) + \beta_0 P_0(x_j) && = x_j P_0(x_j)\\
&\alpha_2 P_2(x_j) + \beta_1 P_1(x_j) + \alpha_1 P_0 && = x_j P_1(x_j)\\
&\vdots\\
&\beta_{n-1} P_{n-1}(x_j) + \alpha_{n-1} P_{n-2}(x_j) && = x_j P_{n-1}(x_j).
\end{aligned} \tag{8}$$

In the matrix form Eq. (8) is

$$\begin{bmatrix} \beta_0 & \alpha_1 & 0 & \dots & 0 \\ \alpha_1 & \beta_1 & \alpha_2 & & \\ 0 & \alpha_2 & \beta_2 & & \\ \vdots & & & & \\ 0 & \dots & 0 & \alpha_{n-1} & \beta_{n-1} \end{bmatrix} \begin{bmatrix} P_0(x_j) \\ P_1(x_j) \\ P_2(x_j) \\ \vdots \\ P_{n-1}(x_j) \end{bmatrix} = x_j \begin{bmatrix} P_0(x_j) \\ P_1(x_j) \\ P_2(x_j) \\ \vdots \\ P_{n-1}(x_j) \end{bmatrix}, \tag{9}$$

or

$$JP(x_j) = x_j P(x_j), \tag{10}$$

where $P(x_j) = [P_0(x_j), \dots, P_{n-1}(x_j)]$. Therefore,**the nodes that we need for the n-point Gaussian quadrature are the eigenvalues of J!**

Now we need to find a way to obtain the weights. We will use the facts that (i) the Gaussian quadrature is exact for P_0, P_1,... ,P_{n-1}, (ii) $P(x_j)$ is the eigenvector for x_j, and (iii) P_0 is constant and we can easily find it and set the proper scaling for the rest of the entries of P.

Since

$$(P_i, P_k) = \int_a^b P_i(x)P_k(x)w(x)dx = \delta_{ik}, \quad 0 \le i, k \le n-1,$$

and the Gaussian quadrature is exact on these polynomials, we have

$$\delta_{ik} = (P_i, P_k) = \sum_{j=1}^{n} P_i(x_j) P_k(x_j) w_j.$$

In the matrix form this expression is

$$PWP^T = I, \tag{11}$$

where $W = \mathrm{diag}(w_1, w_2, \ldots, w_n)$ and $P = [P(x_1), P(x_2), \ldots, P(x_n)]$, i.e., P is the matrix whose columns are the properly scaled eigenvectors of J.

It follows from Eq. (11) that P is invertible, hence

$$W = P^{-1}P^{-T} = (P^T P)^{-1}.$$

Thus,

$$W^{-1} = P^T P,$$

which means that

$$\frac{1}{w_j} = \sum_{k=0}^{n-1} (P_k(x_j))^2 = \|P(x_j)\|^2, \tag{12}$$

where $\|\cdot\|$ is the Euclidean norm.

On the other hand, given an eigenvector $v^{(j)} = [v_1^{(j)}, \ldots, v_n^{(j)}]$ of the matrix J there exists a constant C such that

$$v_j = CP(x_j) = C[P_0(x_j), P_1(x_j), \ldots, P_{n-1}(x_j)].$$

The value of C can be obtained by considering

$$1 = (P_0, P_0) = P_0^2 \int_a^b w(x)dx = P_0^2 \mu_0.$$

It follows that

$$P_0 = \frac{1}{\sqrt{\mu_0}}.$$

Hence

$$v_1^{(j)} = CP_0 = \frac{C}{\sqrt{\mu_0}}.$$

Then

$$C = v_1^{(j)} \sqrt{\mu_0}.$$

Therefore,

$$P(x_j) = \frac{1}{C} v^{(j)} = \frac{1}{v_1^{(j)} \sqrt{\mu_0}} v^{(j)}.$$

Finally, we obtain the weight w_j associated with the node x_j:

$$w_j = \frac{1}{\|P(x_j)\|^2} = \mu_0 \frac{(v_1^{(j)})^2}{\|v^{(j)}\|^2}. \tag{13}$$

1.4.1. *Nonorthonormal polynomials.* The TTRR's for classic orthogonal polynomials are traditionally given in the form

$$xp_k(x) = a_k p_{k+1} + b_k p_k(x) + c_k p_{k-1}(x), \quad c_0 p_{-1}(x) = 0 \tag{14}$$

that leads to nonorthonormal families. We need to convert TTRR (16) to the TTRR for the orthonormal family. Since the orthonormal polynomials $P_k(x)$ and the orthogonal polynomials $p_k(x)$ relate via

$$p_j(x) = \lambda_j P_j(x),$$

we have

$$x\lambda_k P_k(x) = a_k \lambda_{k+1} \lambda P_{k+1} + b_k \lambda_k P_k(x) + c_k \lambda_{k-1} P_{k-1}(x). \tag{15}$$

Hence

$$\alpha_{k+1} = a_k \frac{\lambda_{k+1}}{\lambda_k}, \quad \beta_k = b_k, \quad \alpha_k = c_k \frac{\lambda_{k-1}}{\lambda_k}.$$

Therefore,

$$\alpha_k = a_{k-1} \frac{\lambda_k}{\lambda_{k-1}} = c_k \frac{\lambda_{k-1}}{\lambda_k} = \sqrt{a_{k-1} c_k},$$

To summarize,

$$\alpha_k = \sqrt{a_{k-1} c_k}, \quad \beta_k = b_k. \tag{16}$$

1.4.2. *The Golub-Welsch algorithm. Input:* $\mu_0 = \int_a^b w(x)dx$
Output: $x_1,,\ldots, x_n$*;* $w_1, \ldots, w_n$.

- Build J from α_1, ..., α_{n-1}, $\beta_0,,$ ..., β_{n-1}.
- Compute eigenvalues ρ_1, ..., ρ_n and eigenvectors v_1, ..., v_n of J.
- for $i = 1 : n$
  $x_i = \rho_i$,
  $w_i = \mu_0 \frac{v_i(1)^2}{\|v_i\|^2}$.
  end

The following subroutine implements the Golub-Welsch algorithm in matlab for computing the Gamma function. It tests that $\Gamma(5) = \int_0^\infty x^{5-1} e^{-x} dx = 4!$.

```
function golubwelsch()
n=20;
b=zeros(n,1);
a=zeros(n,1);
b(1)=1;
for j=2:n
    k=j-1;
    b(j)=b(k)+2;
    a(j)=k;
end
mu0=1;
%% form J
```

```
J=zeros(n);
J(1,1)=b(1);
J(1,2)=a(2);
for j=2:n-1
    J(j,j)=b(j);
    J(j,j-1)=a(j);
    J(j,j+1)=a(j+1);
end
J(n,n)=b(n);
J(n,n-1)=a(n);

%% find eigenvalues and eigenvectors
[V E]=eig(J);

%% find weigths
x=zeros(n,1);
w=zeros(n,1);
for j=1:n
    x(j)=E(j,j);
    w(j)=mu0*V(1,j)^2/norm(V(:,j))^2;
end

%% compute integral

I=sum(myf(x).*w);
fprintf('I = %.14e\n',I);

%%
function y=myf(x)
p=zeros(5,1); % polynomial of 4-th degree
p(1)=1; % p(x)= x^4
y=polyval(p,x);
```

1.5. **Example: Gauss-Chebyshev quadrature.** For the Gauss-Chebyshev quadrature nodes, i.e. zeros of $T_n(x)$, are given by

$$x_j = \cos\left(\frac{\pi(j-\frac{1}{2})}{n}\right), \quad j = 1, \ldots, n,$$

and weights can be found easily.

As we know, the $n-1$-st degree Chebyshev interpolant of $f(x)$ is

$$f_{n-1} = \sum_{j=0}^{n-1}{}' c_i T_i(x),$$

($\sum\!\!'$ means that the first term is divided by 2), where

$$c_i = \frac{2}{n}\sum_{k=1}^{n} f(x_k)T_i(x_k), \quad x_k = \cos\left(\frac{\pi(k-\frac{1}{2})}{n}\right), \quad k = 1, \ldots, n.$$

The Gauss-Chebyshev rule consists in the approximation

$$I(f) = \int_{-1}^{1} \frac{f(x)}{\sqrt{1-x^2}}dx \approx Q(f) = \int_{-1}^{1} \frac{f_{n-1}(x)}{\sqrt{1-x^2}}dx.$$

Plugging in the expression for $f_{n-1}(x)$ and using the facts that $(T_i, T_k) = 0$ for $i \neq k$, $T_0 = 1$ and $(T_0, T_0) = \pi$ we get

$$Q(f) = \int_{-1}^{1} \frac{1}{\sqrt{1-x^2}} \sum_{j=0}^{n-1}{}'c_i T_i(x)dx = \frac{c_0}{2}(T_0, T_0) = \frac{\pi c_0}{2}.$$

Now, using the expression for c_0 we obtain

$$Q(f) = \frac{\pi c_0}{2} = \frac{\pi}{2}\frac{2}{n}\sum_{k=1}^{n} f(x_k) = \frac{\pi}{n}\sum_{k=1}^{n} f(x_k).$$

Therefore, the weights for the Gauss-Chebyshev quadrature are all equal to π/n:

$$w_i = \frac{\pi}{n}, \quad i = 1, \ldots, n.$$

References

[1] A. Gil, J. Segura, N. Temme, Numerical Methods for Special Functions, SIAM, 2007