Introduction to Numerical Integration
Part II

CS 375/Math 375

Brian T. Smith, UNM, CS Dept.

Spring, 1998

Intro to Gaussian Quadrature

- As before, the general treatment changes the integration problem to finding the integral

$$\int_a^b f(x)w(x)dx$$

- The function *w(x)* is introduced to handle infinite intervals (*a*,*b*) and functions *f(x)* that have singularities in the interval of integration.

Intro to Gaussian Quad. Cont.

– Again suppose we know *f(x)* at *N* distinct points $\{x_1, x_2, \ldots, x_N\}$ and consider the interpolating polynomial that approximates *f(x)* at these points

  – the interpolating polynomial is never formed explicitly but integrals of the Lagrange polynomials are determined and yield various rules (Simpson's, trapezoidal, midpoint, …)

$$\int_a^b f(x)w(x)dx \approx \int_a^b \sum_{i=1}^{N} f(x_i)L_i(x)w(x)dx =$$

$$\sum_{i=1}^{N} f(x_i)\int_a^b L_i(x)w(x)dx = \sum_{i=1}^{N} A_i f(x_i)$$

Intro to Gaussian Quad Cont.

– The A_i are called the weights

– The formula $\sum_{i=1}^{N} A_i f(x_i)$ is called the quadrature rule

– All the rules so far have this structure

– The rule integrate exactly all polynomials *f(x)* up to a certain degree, say of degree *d*

  • *d* is called the degree of precision of the rule

Intro to Gaussian Quadrature

- Hence our integration rule can be written as

$$\int_a^b f(x)w(x)dx = \sum_{i=1}^{N} A_i f(x_i) + E(f)$$

- Last time, we saw that the for x_i fixed, the *method of undetermined coefficients* allowed us to pick the weights so that the rule correctly integrated polynomials up to degree *d=N-1*.

Intro to Gaussian Quadrature

- In the method of undetermined coefficients, only the N weights are treated as unknowns.
- Demanding that polynomials of degree N-1 or less be integrated exactly gives use N equations we can use to solve for the weights.
- Can we do better, if we choose the x_i as well?

Intro to Gaussian Quadrature

- The answer is yes!
- Carefully choosing the x_i as well allows the rule to integrate polynomials of up to degree 2N-1. These are the *Gaussian* quadrature rules.
- Gauss (1814), Jacobi(1826), and Christoffel (1877) derived general formulas for the x_i and A_i for many choices of *w(x)*.

Example - Gaussian Quadrature

- As before, it is easier to work on standard interval (-1,1) and then transform to (a,b).

$$\int_{-1}^{1} f(x)dx = \sum_{i=1}^{N} A_i f(x_i) + cf^{(d+1)}(\xi) \qquad w(x) = 1$$

define

$$t = \frac{b-a}{2}(x+1) + a$$

changing variables gives

$$\int_a^b f(x)dx = \sum_{i=1}^{N}\left(\frac{b-a}{2}A_i\right) f\left(\frac{b-a}{2}x_i + \frac{a+b}{2}\right) + c\left(\frac{b-a}{2}\right)^{d+2} f^{(d+1)}(\xi)$$

2 point Gaussian rule

- We look for a rule of the form

$$\int_{-1}^{1} dx f(x) = A_1 f(x_1) + A_2 f(x_2) + E(f)$$

- We require $E(f)=0$ for $f(x)=\{1,x,x^2,x^3\}$

$$f = 1: \quad 2 = A_1 + A_2$$
$$f = x: \quad 0 = A_1 x_1 + A_2 x_2$$
$$f = x^2: \quad \frac{2}{3} = A_1 x_1^2 + A_2 x_2^2$$
$$f = x^3: \quad 0 = A_1 x_1^3 + A_2 x_2^3$$

2 point Gaussian rule cont.

- The last equation follows from the 2nd if $x_2 = -x_1$, so

$$f = 1: \quad 2 = A_1 + A_2$$
$$f = x: \quad 0 = A_1 x_1 - A_2 x_1$$
$$f = x^2: \quad \frac{2}{3} = A_1 x_1^2 + A_2 x_1^2$$

- These equations give

$$A_1 = A_2 = 1$$
$$x_1 = -x_2 = 1/\sqrt{3} = 0.577350269189626$$

General formulas for Gaussian Quadrature

- The modern theory of Gaussian integration for arbitrary *w(x)*, is largely due to Jacobi and Christoffel, who used the theory of orthogonal polynomials.
- They showed that the x_i for an *N*-point Gaussian rule with weight function *w(x)* are the zeros of the *N*th orthogonal polynomial determined by *w(x)* and the interval.

Mathematical interlude...

- Orthogonal polynomials are defined by

$$\left\langle p_N \middle| p_M \right\rangle = \int_a^b dx w(x) p_N(x) p_M(x) = c_N \delta_{N,M}$$

- if c_N is 1 then the polynomials are called *orthonormal.*
- the $p_N(x)$ can be constructed for *w(x)* using Gram-Schmidt orthogonalization.

Gaussian Quadrature cont.

- The weights, A_i, corresponding to the x_i have a nice representation in terms of the orthogonal polynomials, too. They are:

$$A_i = \frac{\left\langle p_{N-1} \middle| p_{N-1} \right\rangle}{p_{N-1}(x_i)\, p'_N(x_i)}$$

Gaussian Quadrature cont.

- For several common *w(x)*, the polynomials, and hence the quadrature rules are well known. E.g.,
  - Gauss-Legendre, $w(x) = 1, -1 < x < 1, p_N(x) = P_N(x)$
  - Gauss-Chebyshev, $w(x) = \frac{1}{\sqrt{1-x^2}}, -1 < x < 1, p_N(x) = T_N(x)$
  - Gauss-Laguerre, $w(x) = x^\alpha e^{-x}, 0 < x < \infty, p_N(x) = L_N^\alpha(x)$
  - Gauss-Hermite, Gauss-Jacobi

Gaussian Quadrature cont.

- One problem with using sequences of Gaussian rules is that the x_i change from rule to rule. Hence, one cannot reuse function values.
  - Kronrod developed a method of adding N+1 points to the points of an Nth order Gaussian rule to obtain a rule of degree 3N+1. (The weights change, so must save f values.)

Constructing a general integrator

- So far we have talked about *fixed* integration rules. (I.e., N is fixed).
- Now we want to write a general purpose integrator using these fixed-rules.
- General purpose integrators can be classed as either
  - iterative or non-iterative
  - adaptive or non-adaptive

General purpose integration

- Iterative schemes compute successive approximations to the integral using higher order rules until some sort of agreement is reached. Non-iterative schemes only do 1 or 2 approximations and stop.
- Adaptive schemes choose the points at which *f(x)* is evaluated depending on the behavior of *f(x)*. Non-adaptive schemes choose points independent of *f(x)*.

Adaptive Quadrature

- In an adaptive scheme, the user inputs a desired accuracy (relative & absolute error).
- A fixed integration rule, or set of fixed rules is then applied to the integration range.
  - If more than one fixed rule is used, the accuracy is tested by comparing the results of the rules. If the test fails, the integration range is split (typically in half) and the rules applied to each piece. This process is repeated until convergence.

Adaptive Quadrature

– If only a single fixed rule is used, then the integration range is split and the fixed rule is applied to each piece. The fixed rule is compared against the sum of the pieces. If no agreement is found, then the pieces are split, etc.

– Typically, the user also enters tolerances for how small the integrator is allowed to divide the ranges.

Adaptive Quadrature Code

- An adaptive integrator based on a Gauss-Kronrod scheme is presented in section 5.3, pages 188-189.
- C++ and F90 versions of the code are in
  - ~bsmith/public_html/CS375/Codes/C++/adapt.C
  - ~bsmith/public_html/CS375/Codes/f90/adapt.f90
- Advantages of G-K based scheme:
  - non-equally spaced points, no end points.
  - reuse previous function calls as much as possible.

G-K based adaptive algorithm

- Over *i*th interval, compute 2 approximations to the integral $Q_i, \hat{Q}_i$

  using 3-point G and 7 point G-K rules.
- Estimate error in interval as

$$E_i = | Q_i - \hat{Q}_i |$$

- Now divide *i*th interval if

$$E_i > \frac{(b_i - a_i)}{(b-a)} \max(AER, RER * \left| \sum_j Q_j \right|)$$

Adaptive Algorithm

- To start the process, form Q for [a,b] and its estimated error ERREST=E. If |ERREST|<=TOL, ANSWER=Q and we stop.
- Otherwise, put {α=a,β=b,Q,E} into a queue (linked list) and start the following loop:

Adaptive Algorithm Loop

- Remove α,β,Q,E from the top of the queue
- compute $\chi=(\alpha+\beta)/2$
- compute QL for $[\alpha,\chi]$ and its error EL
- compute QR for $[\chi,\beta]$ and its error ER
- ANSWER+=((QL+QR)-Q)
- ERREST+=((EL+ER)-E)
- if EL is too big, add α,χ,QL,EL to end of queue
- if ER is too big, add χ,β,QR,ER to end of queue

Adaptive Algorithm

- The loop stops when
- The queue is empty
- |ERREST|<=TOL
- The queue gets too large
- Too many function evaluations are made