Extrapolation

- Recall, for the Trapezoidal rule we have established,

$$
\begin{aligned}
T_{2M} \quad + \quad EST_{2M} = \sum_{i=1}^{M} T_2^{(i)} + \frac{1}{3}(T_2^{(i)} - T_1^{(i)}), \\
&= \sum_{i=1}^{M} \frac{4}{3} T_2^{(i)} - \frac{1}{3} T_1^{(i)}, \\
&= \sum_{i=1}^{M} \left\{ \frac{h}{3}(f_{i-1} + 2f_{i-1/2} + f_i) - \frac{h}{6}(f_{i-1} + f_i) \right\}, \\
&= \sum_{i=1}^{M} \frac{h}{6} \left\{ f_{i-1} + 4f_{i-1/2} + f_i \right\}, \\
&= S_M \quad = I(f) + O(h^4),
\end{aligned}
$$

(A fourth order approximation to $I(f)$).

Extrapolation (cont)

- This process of taking a basic quadrature rule, applying it with a sequence of 'stepsizes' $h, h/2, h/4, \cdots h/2^k$ and then using a linear combination of the resulting approximations, $A_0, A_1, \cdots A_k$ to obtain a higher order approximation is called extrapolation.
  When the Trapezoidal rule is used as the basic rule this is called Romberg quadrature (or Romberg Integration).

- To justify extrapolation for the Trapezoidal rule we must show that whenever $f(x)$ has $(2k+2)$ continuous derivatives, then the true error satisfies,

$$E_M^T = c_1 h^2 + c_2 h^4 + \cdots c_k h^{2k} + O(h^{2k+2}),$$

  where the c_i's are independent of h.

Extrapolation (cont)

One can then 'eliminate' the h^2 term in the error by taking a linear combination of T_M and T_{2M}. Let h be the interval width associated with T_{2M}. We then have,

$$\begin{aligned} T_{2M} &= I(f) + c_1 h^2 + \cdots c_k h^{2k} + O(h^{2k+2}), \\ T_M &= I(f) + c_1 (2h)^2 + \cdots c_k (2h)^{2k} + O(2h)^{2k+2}. \end{aligned}$$

Defining T_{2M}^1 by,

$$\begin{aligned} T_{2M}^1 \equiv \frac{4T_{2M} - T_M}{3} &= T_{2M} + \frac{1}{3}(T_{2M} - T_M) \\ &= I(f) + c_2^1 h^4 + c_3^1 h^6 + \cdots c_k^1 h^{2k} + O(h^{2k+2}), \end{aligned}$$

we have derived an error expansion for this fourth order approximation.

Extrapolation (cont.)

Similarly, by considering the resulting error expressions for T^1_{2M} and T^1_{4M} we can 'eliminate' the $O(h^4)$ term to obtain,

$$
\begin{aligned}
T^2_{4M} &\equiv \frac{16T^1_{4M} - T^1_{2M}}{15}, \\
&= T^1_{4M} + (\frac{T^1_{4M} - T^1_{2M}}{15}), \\
&= S_{2M} + EST^S_{2M}, \\
&= I(f) + c^2_3 h^6 + c^2_4 h^8 + \cdots c^2_k h^{2k} + O(h^{2k+2}).
\end{aligned}
$$

Extrapolation Contined

This process can continue and we have, in general,

$$T^{m}_{2^m M} \equiv T^{m-1}_{2^m M} + \frac{T^{m-1}_{2^m M} - T^{m-1}_{2^{m-1} M}}{4^m - 1},$$

where we have the following expansion of the error,

$$T^{m}_{2^m M} = I(f) + c^{m}_{m+1} h^{2(m+1)} + \cdots c^{m}_{k} h^{2k} + O(h^{2k+2}).$$

Observations re Extrapolation

- This technique gives high order approximations with error estimation but round-off limits the accuracy that can be achieved. In practice we usually have $m \leq 6$ or 7.

- Note that we can obtain more accuracy by increasing m or M since the error term associated with $T^m_{2^m M}$ is $O(h^{2(m+1)})$ which is $O((\frac{b-a}{M})^{2(m+1)})$. The 'cost' of computing this approximation is $2^m M$ evaluations of the integrand.

Error Estimates for GQ

Let $G_n(f) = \sum_{i=0}^{n} \omega_i f(x_i)$ denote the $(n+1)$ – point Gaussian quadrature rule.

- We have shown,

$$I(f) - G_n(f) = O(b-a)^{2n+3}, \text{ as } (b-a) \to 0.$$

- The rules $G_{n+1}, G_{n+2}, \cdots$, are more accurate (as $(b-a) \to 0$) so we could use,

$$\widehat{EST}_{G_n} \equiv G_{n+k}(f) - G_n(f) = E_{G_n} + O(b-a)^{2(n+k)+3}.$$

- The rules G_{n+k} and G_n have at most one common interpolation point so the computation of this error estimate more than doubles the cost ($2n+k+2$ integrand evaluations).

Error Est for GQ (cont)

- An alternative (to forming an error estimate based on G_{n+k}) is to use the integrand evaluations already available (for the computation of $G_n(f)$) and introduce only the minimum number of extra evaluations required to obtain an effective error estimate.
- This approach leads to a class of quadrature rules called Kronrod quadrature rules, $K_{n+k}(f)$. The error estimate for $G_n(f)$, is then $K_{n+k}(f) - G_n(f)$, where $K_{n+k}(f)$ is more accurate and less expensive to compute than is $G_{n+k}(f)$. Kronrod proposed a particularly effective class of such rules where $k = n + 1$,

$$K_{2n+1}(f) \equiv \sum_{i=0}^{n} a_i f(x_i) + \sum_{j=0}^{n+1} b_j f(y_j),$$

$$K_{2n+1}(f) \equiv \sum_{i=0}^{n} a_i f(x_i) + \sum_{j=0}^{n+1} b_j f(y_j)$$

- The $x_i's$ are the interpolation points associated with $G_n(f)$, and the y_i's are the extra interpolation points necessary to define an accurate approximation to $I(f)$. Kronrod derived these weights (the a_i's and the b_i's) and the extra interpolation points $(y_0, y_1, \cdots y_n)$ so that the resulting rule is order $3n+3$.
- The resulting error estimate is then,

$$EST_{G_n} \equiv K_{2n+1}(f) - G_n(f),$$

with an associated cost of $2n+3$ integrand evaluations and an order of accuracy of $O((b-a)^{3n+4})$.
- These Gauss-Kronrod pairs of rules can be the basis for composite quadrature rules and adaptive methods. These methods are widely used and implemented in numerical libraries.

Adaptive Quadrature

A straightforward implementation of a method based on a basic quadrature rule with error estimate would attempt to provide an approximation, A, that satisfies $|I(f) - A| < TOL$. It would have input parameters,

- The integrand function, $f(x)$.
- The upper and lower limits of integration, a and b.
- The desired accuracy, TOL.

The obvious implementation would be similar to that presented for the composite rules. That is, after applying the basic rule, if the magnitude of the associated error estimate exceeds TOL, the interval $[a, b]$ is subdivided and the basic rule applied to each sub-interval.

Adaptive Quadrature (cont)

- This process of interval halving and updating the approximation to $I(f)$ and the associated error estimate $EST(f)$ continues until $|EST(f)| < TOL$ with some failure condition possible if no convergence is achieved after a reasonable amount of effort (for example after $8 - 10$ subdivisions).

- Such an implementation will work well for functions that are smooth and relatively well behaved over the interval of integration. But such a method can be inefficient if the integrand is badly behaved on only a small part of the interval of integration. In such cases it would be more effective to concentrate the effort (the integrand evaluations) in the neighborhood where the integrand is changing rapidly. This is the key idea behind 'adaptive' quadrature methods.

Adaptive Quadrature (cont)

- In adaptive quadrature methods, a basic rule with an associated error estimate is implemented in a similar fashion to the straightforward implementation discussed above except that uniform interval halving is not used when more accuracy is needed.
- We selectively refine or subdivide only those subintervals which have an associated error estimate that is too large. That is, we use interval halving to update the approximation and error estimate for only a subset of the subintervals.
- At each step we maintain a partitioning of the interval $a = X_0 < X_1 < \cdots X_N = b$ and we have the associated approximation, $A(f)$ and error estimate, $EST(f)$.

Adaptive Quadrature (cont)

- We then have,

$$
\begin{aligned}
A(f) &\equiv \sum_{r=1}^{N} A_r, \\
EST(f) &\equiv \sum_{r=1}^{N} |EST_r|,
\end{aligned}
$$

  where A_r and EST_r are the approximation and error estimate associated with the r^{th} subinterval.

- The effectiveness of this approach depends largely on how one decides which interval to subdivide next (when $|EST(f)|$ exceeds TOL).

Refinement Strategies

Several possible refinement strategies (strategies for choosing which subinterval to halve) are possible. We will consider two alternatives.

- Refine where error contribution exceeds its 'share':
  Each interval is allowed to contribute an amount to the total error that is proportional to its width. The maximum allowable error on the i^{th} subinterval is then $\frac{x_i - x_{i-1}}{b-a} TOL$. This strategy can be effectively implemented recursively or using stacks.
- Refine where error contribution is largest:
  On each step subdivide only the subinterval with the estimate of largest magnitude. Such a strategy can be effectively implemented using an ordered linked data structure, where the ordering is determined by the magnitude of the corresponding estimate, EST_r.

Observation re First Strategy

On termination we will have,

$$
\begin{aligned}
|EST(f)| &\equiv \sum_{i=1}^{N} |EST_i|, \\
&\leq \sum_{i=1}^{N} \left(\frac{x_i - x_{i-1}}{b-a}\right) TOL, \\
&= \left(\frac{TOL}{b-a}\right) \sum_{i=1}^{N} (x_i - x_{i-1}), \\
&= TOL.
\end{aligned}
$$