

## Lecture 11: Even more quadrature

### Outline

- 1) Quick Review of single-interval quadrature
- 2) Extended Newton-Cotes formulas
  - A) Extended Trapezoidal Rule
  - B) Extended Simpson's Rule
  - C) Higher order rules
- 3) Toward's high accuracy: Two approaches
  - A) Adaptive Quadrature: matlab's quad, quadl (and quadtx)
  - B) Romberg Integration and Richardson's extrapolation
- 4) Numerical differentiation (?)

### Review: Basic Newton Cotes Formulas

In general, all quadrature schemes can be written as

Basic schemes:

Mid-Point:

Trapezoidal:

Simpsons Rule:

## Review: Gauss-Legendre Quadrature

(choose both  $x_i$  and  $w_i$  to maximize polynomial degree)

2-Point rule:

3-Point rule:  $x_i$  are roots of  $P_3(x) = x/2 (5x^2 - 3)$ ,  $w_i = [5 \ 8 \ 5]/9$

## Affine Transformation of the Interval [a,b]

Note: for Gauss-Legendre quadrature...a bit of matlab shorthand

given, a,b,f(x) and assume e.g. 3 point GL

$H=b-a;$

$x = @(t) a+H/2*(t+1)$

$t_i = [-1 \ 0 \ 1]*\text{sqrt}(3/5);$

$w_i = [5 \ 8 \ 5]/9;$

$Q = H/2*w_i*f(x(t_i));$

## Review Example

Let  $I = \int \sin(x)dx$  for  $x$  in  $[0, \pi]$  (exact  $I=2$ )

Quadrature Rule	$Q$	relative error $r$
Mid Point	$\pi$	0.57
Trapezoidal	0	1.00
Simpsons	$2\pi/3=2.0944$	0.047
2-Point GL	1.9358	0.032
3-Point GL	2.0014	0.0007

## h-Refinement and Extended Newton-Cotes formulas

Big Idea: rather than add points to increase polynomial degree, add points to reduce "grid-spacing"  $h$ .

Example: Extended Trapezoidal rule

## **h-Refinement and Extended Newton-Cotes formulas**

Extended Simpson's Rule:

## **Combining extended estimates for higher order accuracy**

Example:  $T_1, T_2$  and  $S_1$

## Combining extended estimates for higher order accuracy

Example:  $T_N, T_{2N}$  and  $S_N$

## Combining extended estimates for higher order accuracy

Example: Can do the same trick with Extended Simpsons Rule

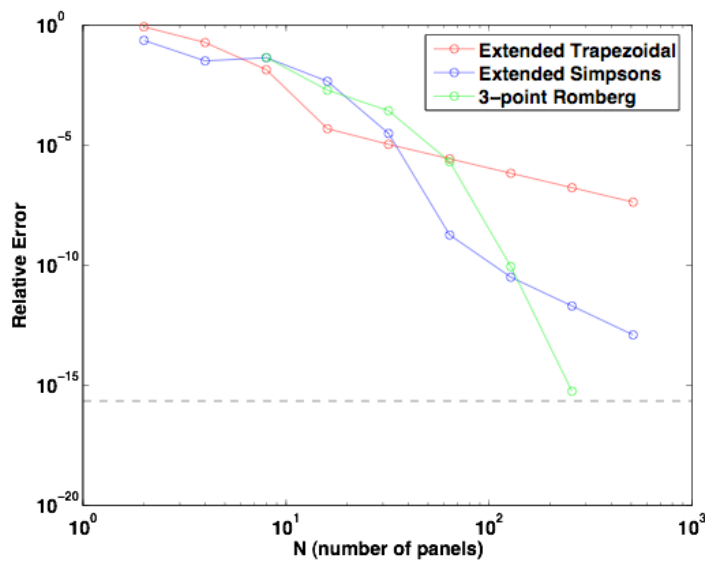
$S_1, S_2, Q_1$

## Higher accuracy: Approach #1 -- Romberg Integration and Richardson's Extrapolation to the limit

Form a tableau from the sequence  $T_1, T_2, \dots, T_{2N}$

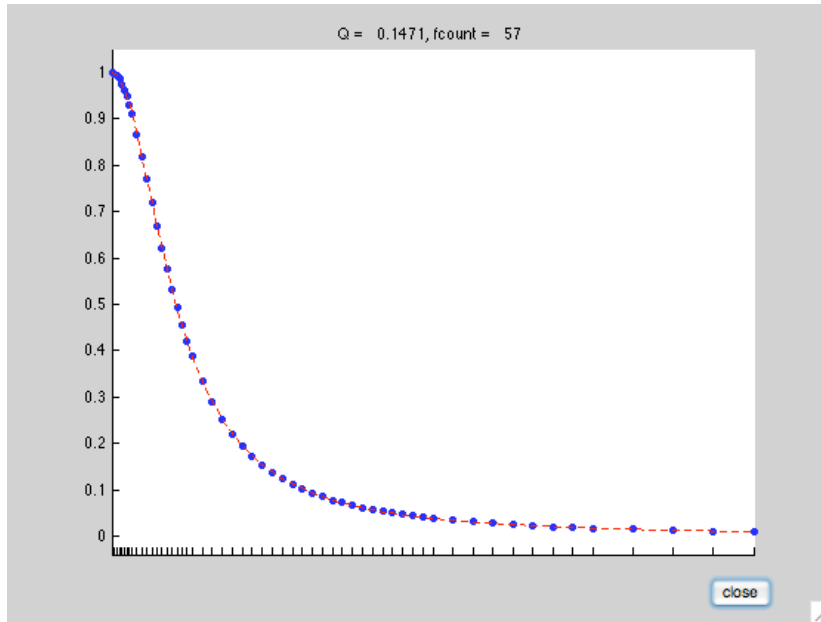
## Higher accuracy: Approach #1 -- Romberg Integration and Richardson's Extrapolation to the limit

Example (from homework) estimate integral  $f(x) = 1/(1+100x^2)$  on  $[0,1]$



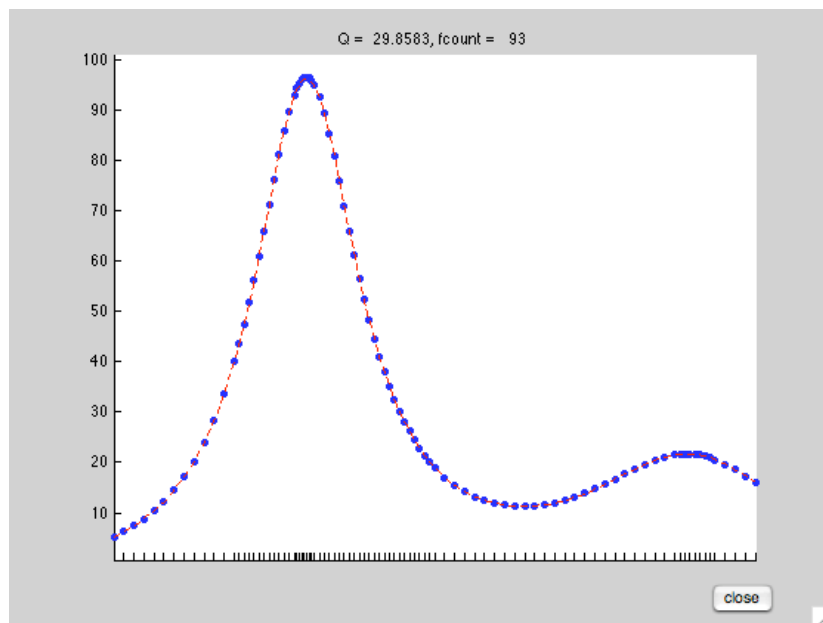
## Higher accuracy: Approach #2 -- Adaptive quadrature

$$f(x) = 1/(1+100x^2) \quad I = \tan^{-1}(10x)/10 \text{ on } x = [0,1] \quad (I(1)=0.1471128)$$



## Higher accuracy: Approach #2 -- Adaptive quadrature

$$f(x) = \text{humps}(x) \quad \text{on } x = [0,1]$$



## Higher accuracy: Approach #2 -- Adaptive quadrature

The algorithm: divide and compare

## Higher accuracy: Approach #2 -- Adaptive quadrature

quadtx: a recursive algorithm for adaptive high-order quadrature

```
function [Q] = quadtx(F,a,b,tol)
% Initialization
c = (a + b)/2;
fa = F(a); fc = F(c); fb = F(b);

% Recursive call
[Q,k] = quadtxstep(F, a, b, tol, fa, fc, fb);
fcount = k + 3;

% -----
function [Q,fcount] = quadtxstep(F,a,b,tol,fa,fc,fb)

% Recursive subfunction used by quadtx.

h = b - a;
c = (a + b)/2; d = (a + c)/2; e = (c + b)/2;
fd = F(d); fe = F(e);
Q1 = h/6 * (fa + 4*fc + fb);
Q2 = h/12 * (fa + 4*fd + 2*fc + 4*fe + fb);
if abs(Q2 - Q1) <= tol
    Q = Q2 + (Q2 - Q1)/15;
    fcount = 2;
else
    [Qa,ka] = quadtxstep(F, a, c, tol, fa, fd, fc);
    [Qb,kb] = quadtxstep(F, c, b, tol, fc, fe, fb);
    Q = Qa + Qb;
    fcount = ka + kb + 2;
end
```

**Higher accuracy: Approach #1 -- Romberg  
Integration and Richardson's Extrapolation to the  
limit**

An alternative Interpretation of Romberg Integration