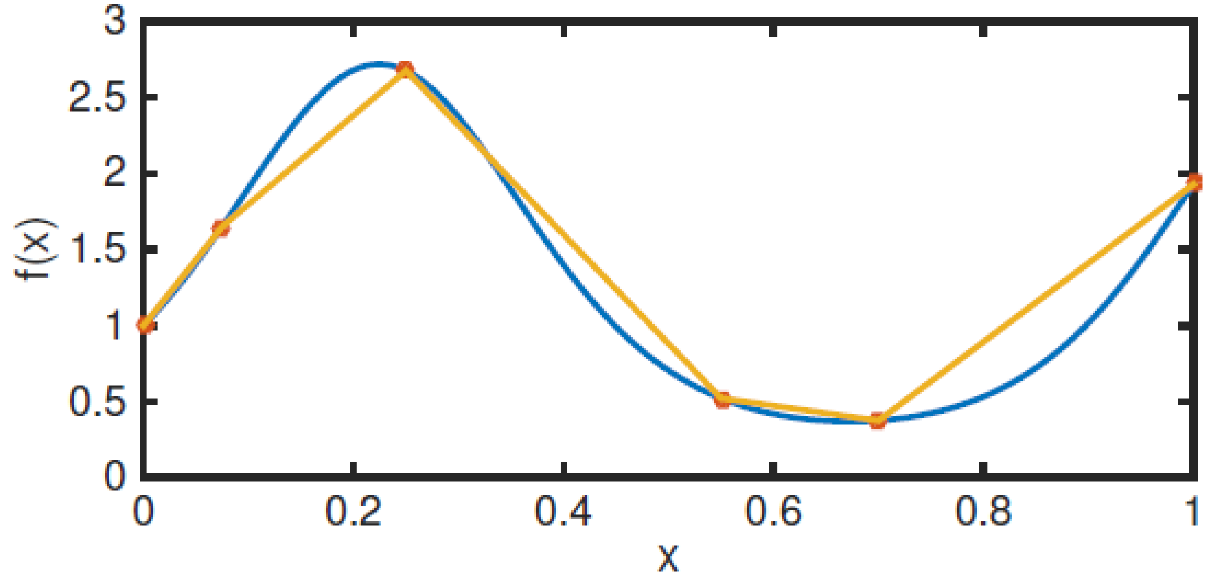
Quadrature (Integration)



Quadrature

- To carry out integration, replace integrand with interpolant to develop approximate formula
- Using interpolation theory we can get both the approximation and the error

Quadrature

- Consider sampling the integrand $f(x)$ at $n+1$ distinct points (nodes) with $(t_0, y_0), (t_1, y_1), \ldots, (t_n, y_n)$ with $t_0 < t_1 < \cdots < t_n$
- Note that the nodes $t_i, i = 0,1, \ldots, n$ are distinct
- Assume even spacing, with $a = t_0, b = t_n, h = (b-a)/n$, and $t_i = a + ih, \quad i = 0,1, \ldots, n$
- We require $p(t_i) = f(t_i), i = 0,1, \ldots, n$
- The resulting approximation to the integral $I = \int_a^b f(x)dx$ is from

$$\int_a^b f(x)\, dx \approx \int_a^b p(x)\, dx.$$

Quadrature

- Using this choice, we get results of the form

$$I = \int_a^b f(x)\,dx \approx Q = \sum_{i=0}^{n} w_i f(t_i) = w_0 f(t_0) + w_1 f(t_1) + \cdots w_n f(t_n)$$

- We should expect different weights w_i from different interpolants
- We should expect more accurate methods from more accurate interpolants

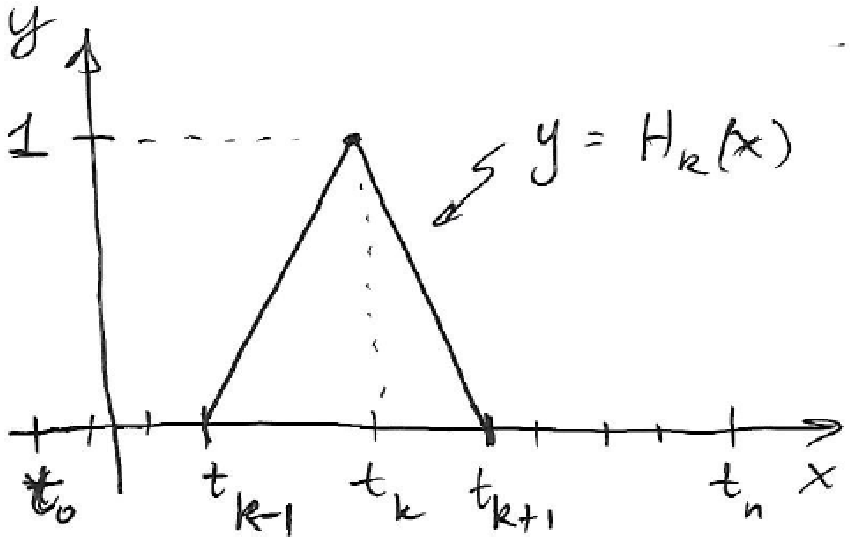
Quadrature

- Let's start with using the PL interpolant :

$$p(x) = \sum_{k=0}^{n} c_k H_k(x),$$

- The hat functions satisy the cardinality conditions: 1 at the node of interest, 0 at every other node:

$$H_k(t_j) = \begin{cases} 1, & \text{if } j = k, \\ 0, & \text{otherwise,} \end{cases}$$



- We choose $c_k = f(t_k)$

Quadrature

- To approximate the integral I, integrate the interpolant:

$$I \approx Q = \int_a^b \sum_{i=0}^{n} f(t_i) H_i(x) dx$$

- We get

$$Q = \sum_{i=0}^{n} f(t_i) \int_a^b H_i(x) dx$$

- The individual integrals are areas under the hat functions, which are the weights:

$$w_i = \int_a^b H_i(x) dx$$

Quadrature

- Here's one weight, for $i = 1$

$$w_1 = \int_a^b H_1(x)dx = \int_{t_0}^{t_1} \frac{x - t_0}{t_1 - t_0} dx + \int_{t_1}^{t_2} \frac{t_2 - x}{t_2 - t_1} dx = \frac{h}{2} + \frac{h}{2} = 1$$

- We get the same thing at all interior points
- At the ends, we get $w_0 = w_n = \frac{1}{2}$
- So, the weights are

$$w_i = \begin{cases} h, & i = 1, \ldots, n-1, \\ \frac{1}{2}h, & i = 0, n. \end{cases}$$

and then

$$I = \int_a^b f(x)\,dx \approx T_f(h) = h\left[\frac{1}{2}f(t_0) + f(t_1) + f(t_2) + \cdots + f(t_{n-1}) + \frac{1}{2}f(t_n)\right]$$

- "Trapezoid formula" or "Trapezoidal Rule" (composite)

Quadrature: Trapezoidal rule

- How about the error?
- From interpolation with $p(x)$ for nodes h apart, we know that

$$\|f(x) - p(x)\|_\infty = \max_{x \in [a,b]} |f(x) - p(x)| \leq Mh^2$$

- If we integrate, then the error over the interval is proportional to $(b - a)h^2$, so the error is still $O(h^2)$
- There is a famous result that gives us even more info…

Quadrature: trapezoidal rule

- The Euler-Maclaurin formula gives use

$$I = \int_a^b f(x)\,dx = T_f(h) - \frac{h^2}{12}\left[f'(b) - f'(a)\right] + \frac{h^4}{740}\left[f'''(b) - f'''(a)\right] + O(h^6)$$

$$= T_f(h) - \sum_{k=1}^{\infty} \frac{B_{2k}}{(2k)!}\left[f^{(2k-1)}(b) - f^{(2k-1)}(a)\right],$$

- Here B_{2k} are called Bernoulli numbers; one way to get them:

$$\frac{te^{xt}}{e^t - 1} = \sum_{m=0}^{\infty} B_m(x)\frac{t^m}{m!}, \qquad |t| < 2\pi$$

- In other words, expand the fraction at right in powers of t, and take the appropriate coefficients for use in the E-M formula
- The error is $O(h^2)$ unless derivatives same at both ends

Quadrature: Trapezoidal rule

- Function `trapezoid.m`
- Assumes even grid point spacing
- Input function f, endpoints a and b, and number of subintervals n

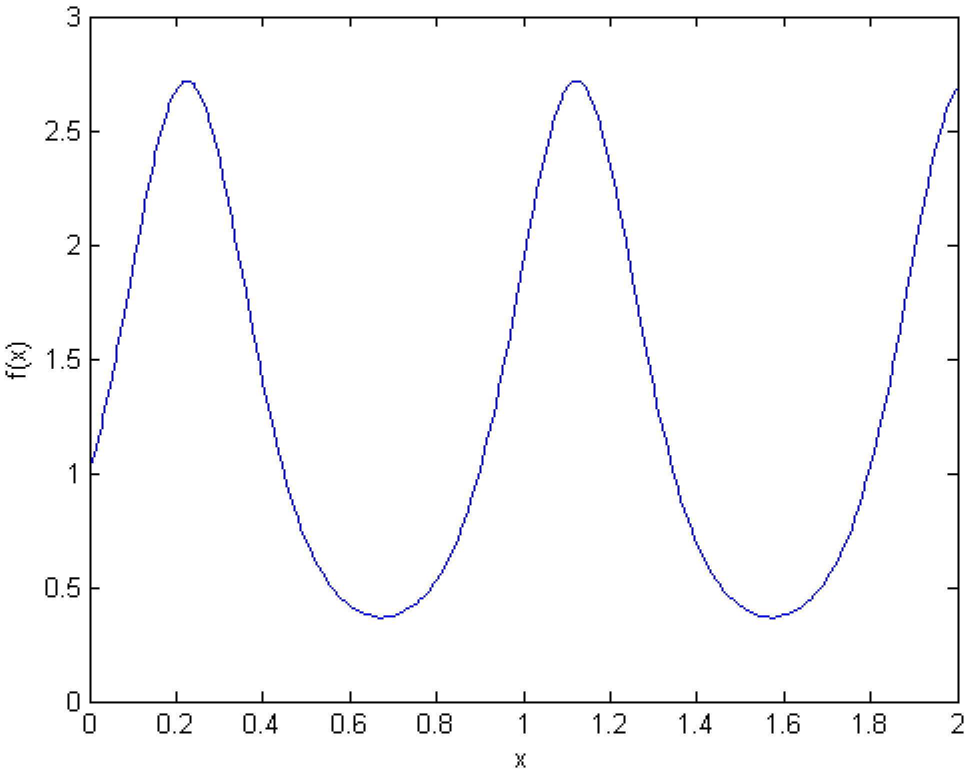
```
function [T,t,y] = trapezoid(f,a,b,n)
%TRAP Trapezoid formula quadrature.
% Input:
%   f   integrand (function)
%   a,b interval of integration (scalars)
%   n   number of interval divisions
% Output:
%   T   approximation to integral(f,a,b)
%   t   vector of nodes used
%   y   vector of function values at nodes

h = (b-a)/n;
t = a + h*(0:n)';
y = f(t);
T = h * ( sum(y) - 0.5*(y(1) + y(n+1)) );
```

Trapezoidal rule: example

- Consider $f(x) = e^{\sin(7x)}$ on $[0,2]$
- First use Matlab builtin `integral` to get "exact" answer

```
close all
f = @(x) exp(sin(7* x));
a = 0; b = 2;
fplot(f,[a,b])
% use built-in function integral to get "exact" answer
I = integral (f,a,b,'abstol',1e-14 , 'reltol',1e-14)
```



Trapezoidal rule: example

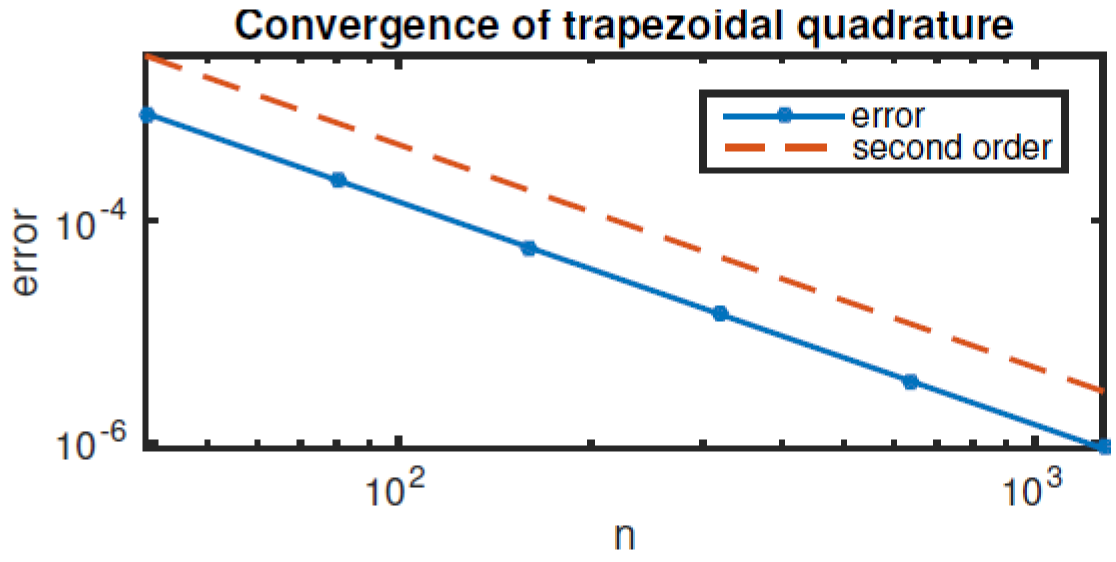
- Consider $f(x) = e^{\sin(7x)}$ on $[0,2]$
- First use Matlab builtin `integral` to get "exact" answer; $I = 2.6632\ldots$
- Then, compute some integrals with text function for different n; first error is 9.17e-4

```
% one answer for fixed n
T = trapezoid (f,a,b ,40) ;
err = I - T
% compute for sequence of n to check convergence order
n_ = 40*2.^(0:5)';
err_ = [];
for n = n_'
  T = trapezoid (f,a,b,n);
  err_ = [ err_; I-T];
end
```

Trapezoidal rule: example

- Consider $f(x) = e^{\sin(7x)}$ on [0,2]
- First use Matlab builtin `integral` to get "exact" answer; $I = 2.6632 \ldots$
- Then, compute some integrals with text function for different n; first error is 9.17e-4
- Compute results for sequence of n to check convergence
- It is $O(h^2)$

```
figure
loglog (n_ ,err_ ,'.-')
hold on , loglog (n_ ,3e-3*( n_/n_ (1)) .^( -2) ,'--')
xlabel ('n'), ylabel ('error '), axis tight
title (' Convergence of trapezoidal quadrature ')
legend ('error ','second order ')
```



Quadrature: Trapezoidal rule

- Try some things yourself:

1. Try $f(x) = |\sin(2\pi x)|$ for [0,2]
2. $f(x) \;=\; e^{\sin(7x)}$ on $[0,2\pi/7]$
3. $f(x) \;=$ sawtooth(x) on [0,2]

- How does the error behave in each case?

Quadrature: More Newton-Cotes rules

- If we increase the degree of interpolant, accuracy improves
- Let $p_2(x)$ be quadratic interpolating $(-h, f(-h)), (0, f(0)), (h, f(h))$

$$P(x) = \frac{x(x-h)}{2h^2} f(-h) - \frac{x^2 - h^2}{h^2} f(0) + \frac{x(x+h)}{2h^2} f(h)$$

- Then

$$I \approx \int_{-h}^{h} p_2(x)dx = \frac{h}{3}[f(-h) + 4f(0) + f(h)]$$

- "Simpon's Rule" or "Simpson 1/3 Rule"
- We need composite form

Quadrature: Simpson 1/3 rule

- For composite rule, say we have grid points at $t_i = a + ih$, $i = 0,1, \dots, n$ with n even
- We can create an integer number of the small subintervals of length 2h, so we can put together a bunch of these small integrals
- Then

$$\int_a^b f(x)\,dx \approx \frac{h}{3}\left[f(t_0) + 4f(t_1) + 2f(t_2) + 4f(t_3) + 2f(t_4) + \cdots \right.$$

$$\left. \cdots + 2f(t_{n-2}) + 4f(t_{n-1}) + f(t_n)\right].$$

- There is $O(h^4)$ for this method

Quadrature: Simpson rule

- Function `Simp2.m`
- Assumes even grid point spacing
- Input function f, endpoints a and b, and number of grid spacings n (must be even)

```
function y = Simp2(f,a,b,n)
% Function Simp2 uses the composite Simpson 1/3 rule
% to approximate the integral of function f -- vectorized now
%
% Input: f - function to integrate
%        a - lower limit of integration
%        b - upper limit of integration
%        n - number of grid spacings (must be even)
 h = (b-a)/n;
 % n intervals means n+1 points; use that in linspace
 x = linspace(a,b,n+1);
 F=feval(f,x);
 % now do integration
 S = F(1) + 4*sum(F(2:2:n)) + 2*sum(F(3:2:n-1)) + F(n+1);
 y = S*h/3;
```

Quadrature: Simpson 1/3 rule

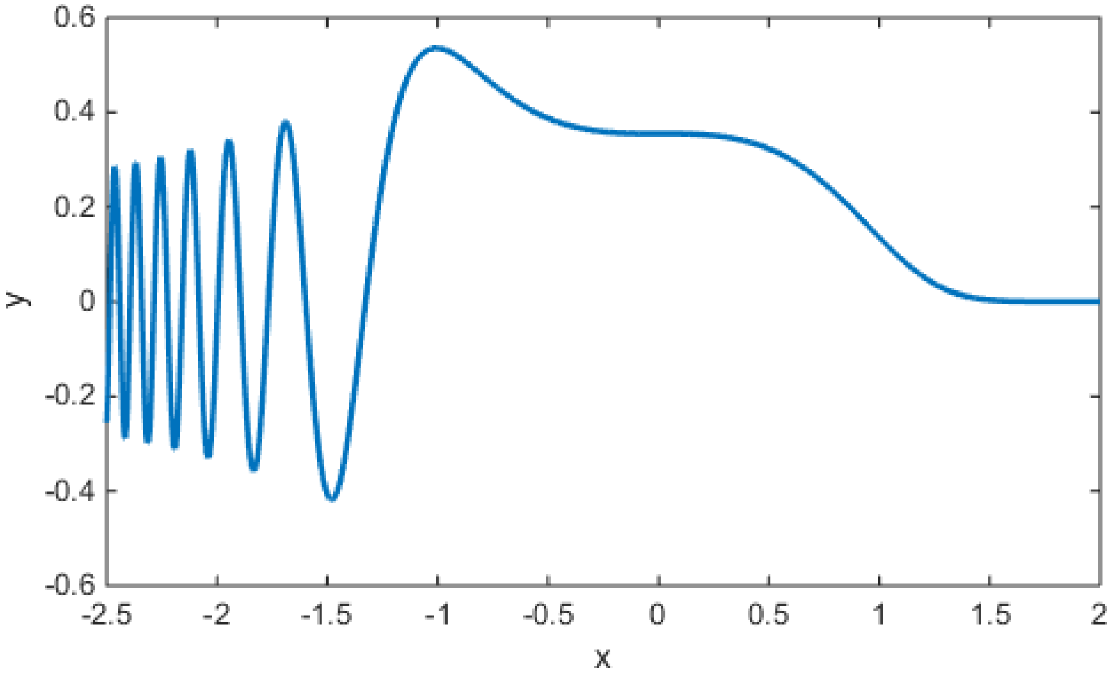
- Try some things yourself using Simp2.m:

1. $f(x) = e^{\sin(7x)}$ on $[0,2]$
2. Try $f(x) = |\sin(2\pi x)|$ for [0,2]
3. $f(x) = e^{\sin(7x)}$ on $[0,2\pi/7]$
4. $f(x)$ = sawtooth(x) on [0,2]

- How does the error behave in each case?

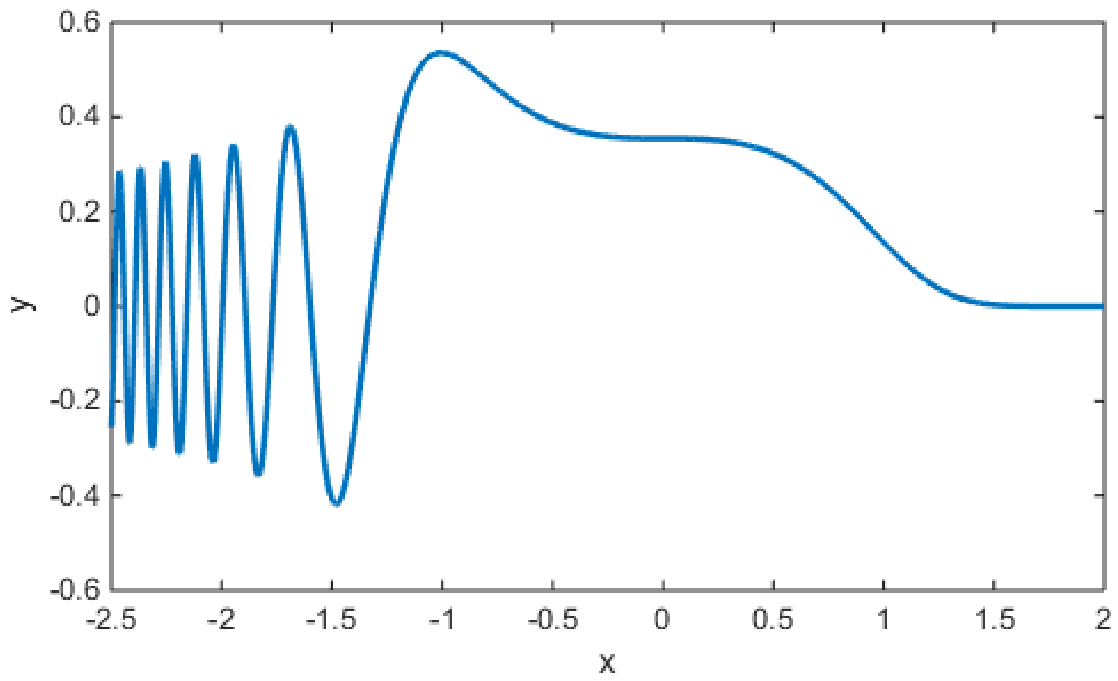
Adaptive Quadrature

- Some functions vary faster in one part of the domain compared to another
- An extreme example is Ai(x^3), shown at right
- We would want to put more nodes to interpolate accurately where there is rapid oscillation (imagine using PL interp)
- It is similar for integration: more points needed where there is fast variation



Adaptive Quadrature

- Strategy: estimate error using knowledge of Simpson rule
- Start by one and two intervals over whole domain
- Apply Simpson rule on all intervals
- Estimate error; if larger than tolerance, then subdivide again in half that did not satisfy tolerance (could be one or both)
- Recursively do this in each subdomain



Adaptive Quadrature

- Simpson rule error is $O(h^4)$
- For one interval, $I = S_1 + Ch^4$
- For two intervals over same limits, $I = S_2 + Ch^4/16$
- We assume C is same for both, but we don't know it
- Subtract the two, and solve for Ch^4
- This gives estimate for error: $\mathrm{E} \approx Ch^4 = \frac{S_1 - S_2}{15} = \delta$
- We compute S_1 and S_2, from the method, then use them to estimate the error
- If $\delta > tol$, then subdivide the interval by calling `adaptsimp.m` again (apply the test again with subdivision)

Adaptive Quadrature

```
function [Q,t] = adaptsimp(f,a,b,tol,fl,fm,fr)
%ADAPTSIMP Adaptive implementation of Simpson's rule with error control.
% Input:
%    f      integrand (function)
%    a      left endpoint (scalar)
%    b      right endpoint (scalar)
%    tol    desired error bound (scalar)
% Output:
%    Q      estimate of int(f,a..b)
%    t      evaluation nodes of f (vector, for information only)

m = (a+b)/2;  h = (b-a)/2;
```

Adaptive Quadrature

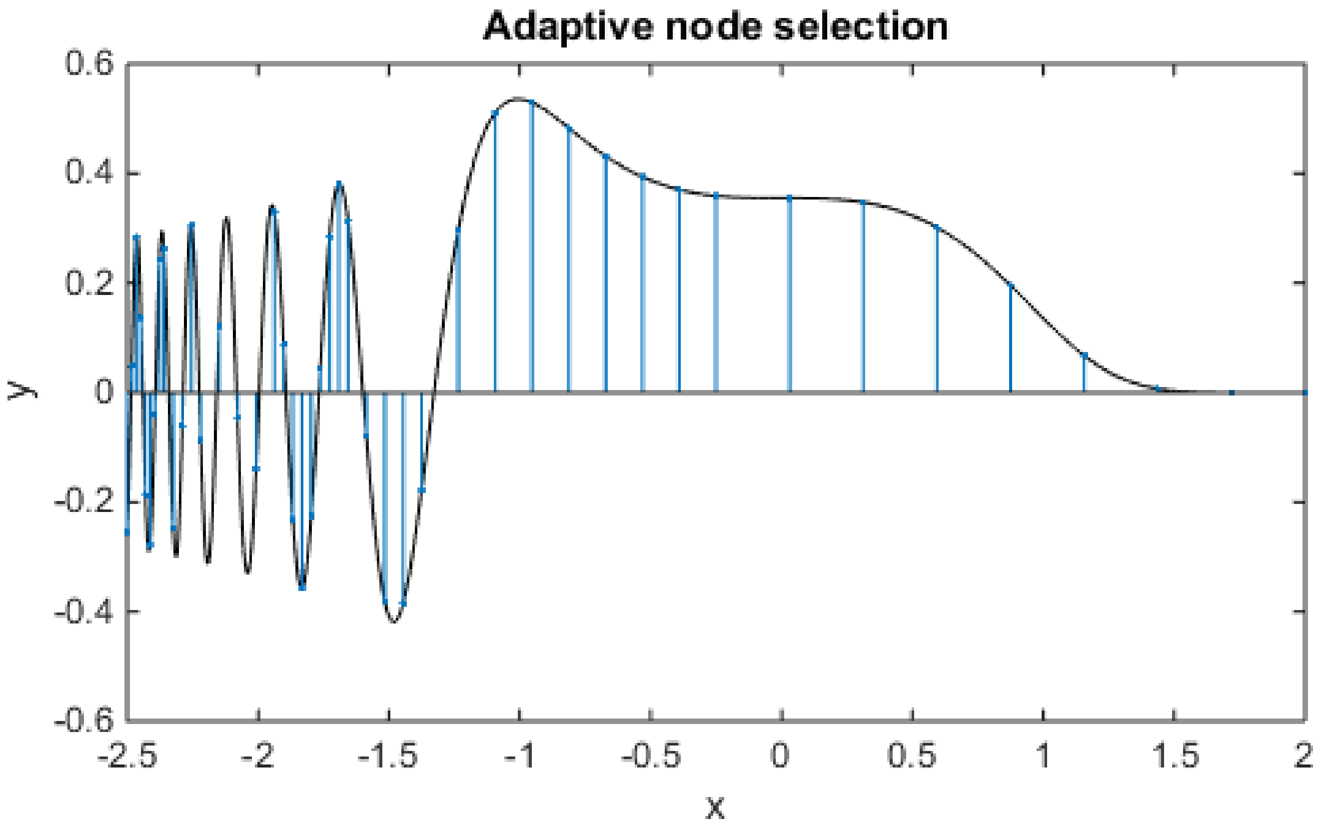
```
% Find the f-values if not supplied recursively.
if nargin==4
  fl = f(a);
  fm = f(m);
  fr = f(b);
end

% Simpson rule estimates at two stepsizes.
S1 = (h/3) * (fl + 4*fm + fr);
lm = (a+m)/2;   flm = f(lm);
rm = (m+b)/2;   frm = f(rm);
S2 = (h/6) * (fl + 4*flm + 2*fm + 4*frm + fr);

delta = (S1-S2)/15;   % error estimate for S2
if abs(delta) < tol   % accept
  Q = S2;
  x = [a lm m rm b]';
else                  % divide and conquer
  [I1,x1] = adaptsimp(f,a,m,tol,fl,flm,fm);
  [I2,x2] = adaptsimp(f,m,b,tol,fm,frm,fr);
  Q = I1+I2;
  x = [ x1; x2(2:end) ];   % midpoint is duplicated
end
```

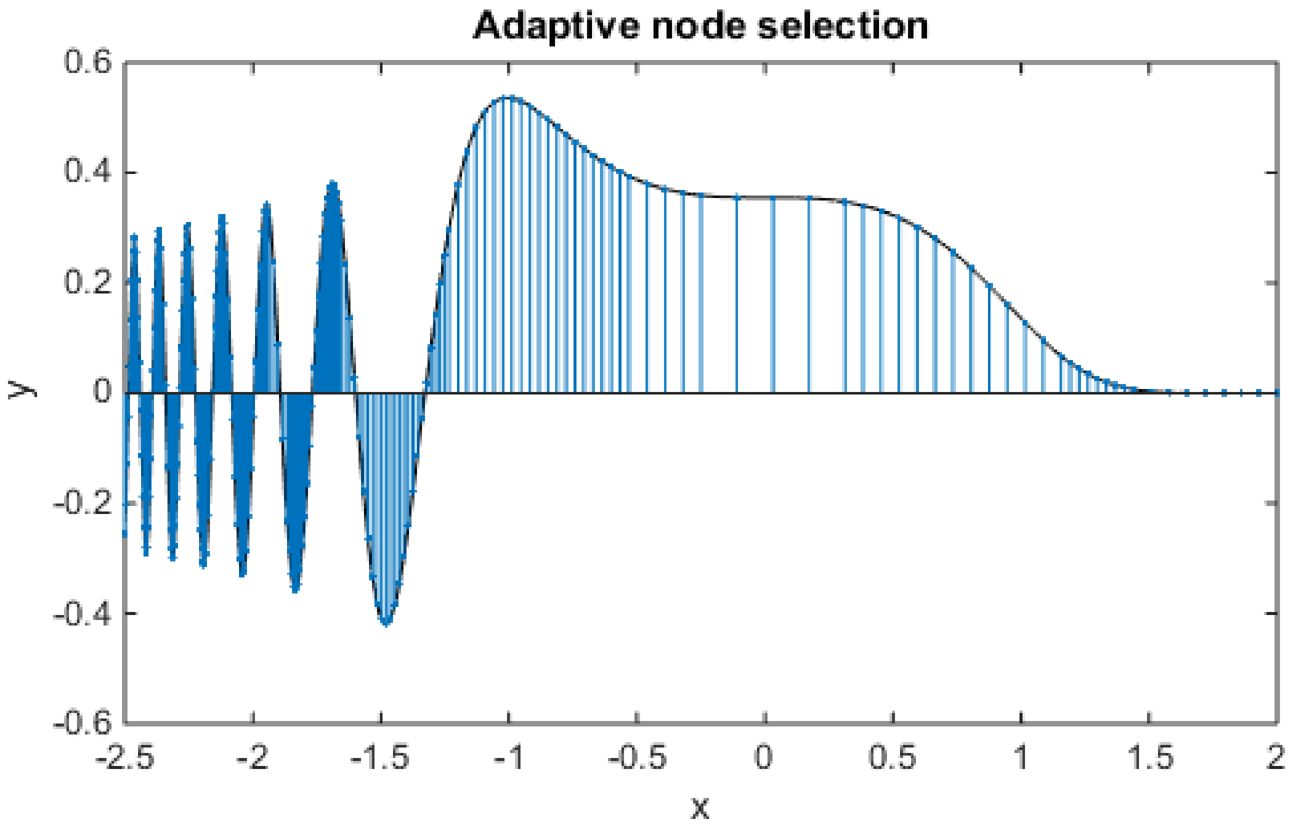
Adaptive Quadrature example

- Consider f(x)=Ai(x^3) over [-2.5,2]
- Uses example file on Sakai
- Calls adaptsimp.m
- Plot at right shows function, and stem plot at points where function values were for tolerance of 1e-3
- More, but not enough, *f* evals in left end of domain



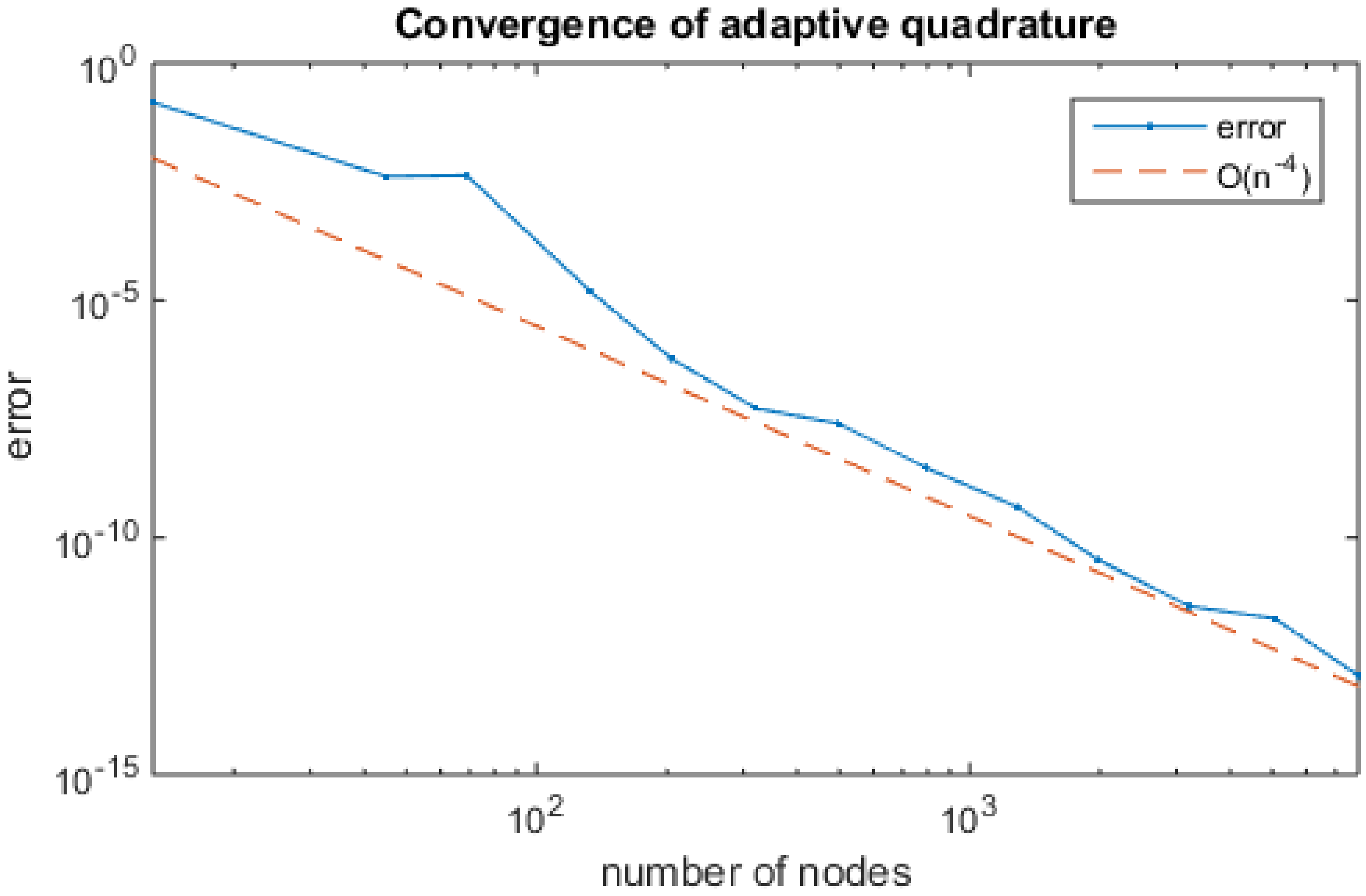
Adaptive Quadrature example

- Consider f(x)=Ai(x^3) over [-2.5,2]
- Calls adaptsimp.m
- Plot at right shows function, and stem plot at points where function values were for tolerance of 1e-6
- Many more *f* evals in left end of domain to drive down error



Adaptive Quadrature example

- Consider f(x)=Ai(x^3) over [-2.5,2]
- Calls adaptsimp.m
- How does error behave?
- No more constant h, but we can compare to number of function evaluations n
- Does behave like $O(n^{-4})$



Adaptive Quadrature

- Try: a different function that oscillates faster or changes rapidly in a different part of the domain.

Adaptive Quadrature

- Text approach is based on extrapolation
- Skipping that approach this semester