Numerical Differentiation and Integration

SELİS ÖNEL, PhD

Quotes of the Day

Trust has to be earned, and should come only after the passage of time. – ***Arthur Ashe***

Trust cannot be commanded; and yet it is also correct that the only one who earns trust is the one who is prepared to grant trust.
- ***Gustav Heinemann***

Numerical Integration

- We know
  - Definite integrals arise in many different areas, and
  - Fundamental Theorem of Calculus is a powerful tool for evaluating definite integrals

- However → it cannot always be applied
  - There are some functions which do not have an anti-derivative, which can be expressed in terms of familiar functions such as polynomials, exponentials and trigonometric functions.

- Ex: $\exp(-x^2)$ is an important function since it is the probability density function for the normal distribution

Numerical Integration

- Allows approximate integration of functions that are analytically defined or given in tabulated form
- Idea is to fit a polynomial to functional data points and integrate it
- The most straightforward numerical integration technique uses the **Newton-Cotes** rules (also called quadrature formulas), which approximate a function at evenly spaced data points by various degree polynomials
- If the endpoints are tabulated, then the 2-point formula is called the **Trapezoidal rule** and the 3-point formula is called the **Simpson's rule**
  - Trapezoidal rule (linear)
  - Simpson's rule (parabolic)
- The 5-point formula is called **Boole's rule**
- A generalization of the trapezoidal rule is **Romberg integration**, which can yield accurate results for many fewer function evaluations

Trapezoidal Rule

- Numerical integration method based on integrating the linear interpolation formula

$$I = \int_a^b f(x)dx$$

Approximating $f(x)$ by linear interpolation gives:

$$g(x) = \frac{b-x}{b-a} f(a) + \frac{x-a}{b-a} f(b)$$

$$I = \int_a^b f(x)dx \approx \int_a^b g(x)dx = \frac{b-a}{2}(f(a) + f(b))$$

$$I = \int_a^b f(x)dx = \frac{(b-a)}{2}(f(a) + f(b)) + E$$

E is the truncation error given by: $E \approx -\frac{1}{12}(b-a)^3 f''$

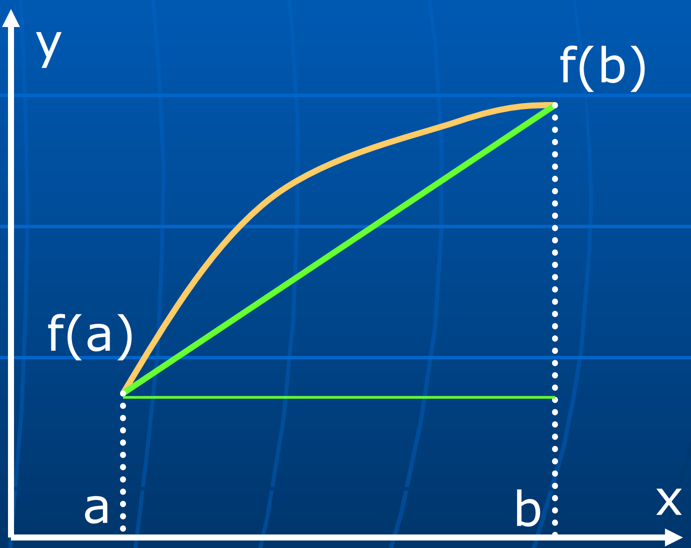
Trapezoidal Rule

- Numerical integration method based on approximating the area under the graph y=f(x) by the trapezoid formed below:

$$\int_a^b f(x)dx = (b-a)f(a) + \frac{1}{2}(b-a)[f(b)-f(a)]$$

$$= (b-a)[f(a) + \frac{1}{2}f(b) - \frac{1}{2}f(a)]$$

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



This alone is not a good approximation, therefore …

Extended Trapezoidal Rule

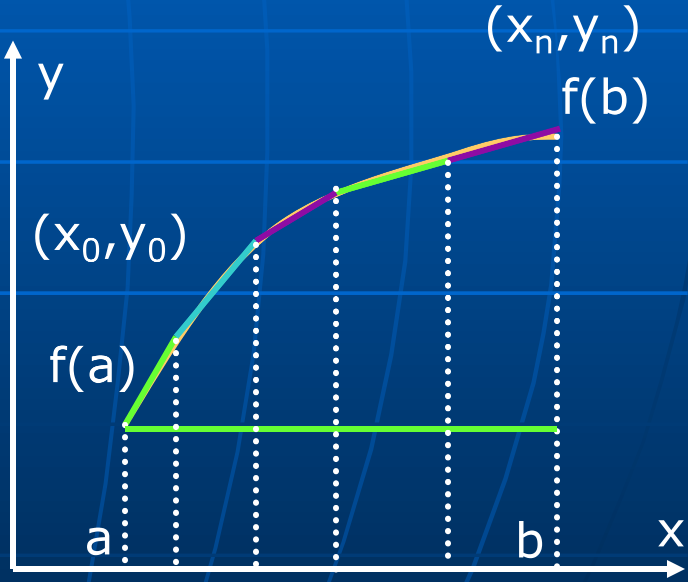
… break the region [a,b] into n equal smaller pieces and apply the approximation on each piece. On the smaller pieces, the graph looks more and more like a straight line so the approximation should improve:

$$\text{let } h = \frac{b-a}{n}, \quad \text{and } \; y_i = f(x_i, y_i)$$

$$\int_a^b f(x)dx \approx \frac{h}{2}(y_0 + y_1) + \frac{h}{2}(y_1 + y_2) + \ldots + \frac{h}{2}(y_{n-1} + y_n)$$

$$= \frac{h}{2}(y_0 + y_1 + y_1 + y_2 + \ldots + y_{n-1} + y_{n-1} + y_n)$$

$$= \frac{h}{2}(y_0 + 2y_1 + 2y_2 + 2y_3 + \ldots + 2y_{n-1} + y_n)$$



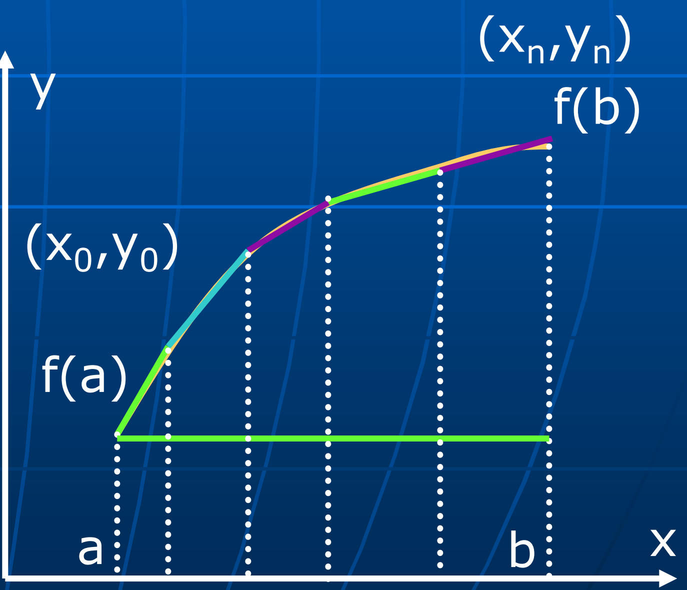
Extended Trapezoidal Rule

… the error E becomes

$$E \approx -\frac{b-a}{12} h^2 \overline{f''} \quad \text{or equivalently} \quad E \approx -\frac{(b-a)^3}{12n^2} \overline{f''}$$

where $\overline{f''}$ is the average of $f''(\mathrm{x})$ in $a < x < b$.

f″ is the second derivative of f(x)



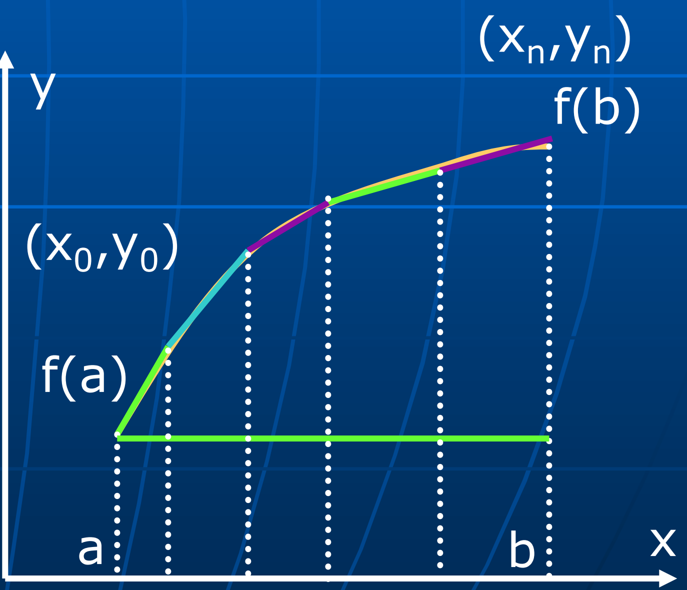
Extended Trapezoidal Rule in MATLAB®

… the extended trapezoidal rule can be written in MATLAB® as:

```
I=h*(sum(f)-0.5*(f(1)+f(length(f))))
```

where

f is an array of f_i for equispaced abscissa points with interval size h



Ex: Extended Trapezoidal Rule in MATLAB®

```
I=h*(sum(f)-0.5*(f(1)+f(length(f))))
```

An automobile of mass M=2000 kg is cruising at a speed of 30 m/s. The engine is suddenly disengaged at t=0 s. How far does the car travel before the speed reduces to 15 m/s?

The force equation for cruising after t=0 is given by:

Acceleration force = Aerodynamic resistance + Rolling resistance

$2000u(du/dx)=-8.1u^2-1200$

where

u: velocity of car,

x: linear distance travelled after t=0

(Ref: Nakamura, 2nd ed., pg.208)

Ex: Extended Trapezoidal Rule in MATLAB®

Acceleration force = Aerodynamic resistance + Rolling resistance

2000u(du/dx)=-8.1u²-1200

Rewriting this equation gives:

$$\frac{2000 \cdot u \cdot du}{-8.1u^2 - 1200} = dx$$

Integrating gives: $\displaystyle\int_{30}^{15} \frac{2000 \cdot u \cdot du}{-8.1u^2 - 1200} = \int_{15}^{30} \frac{2000 \cdot u \cdot du}{8.1u^2 + 1200} = \int_{0}^{x} dx = x$

Using 16 data points or 15 intervals to evaluate the LHS: $i = 1, 2, \ldots, 16$

$$\Delta u = \frac{30 - 15}{i - 1} = \frac{15}{15} = 1, \qquad u_i = 15 + (i-1)\Delta u, \qquad \frac{du}{dx} = f_i = \frac{2000u_i}{8.1u_i^2 + 1200}$$

Using trapezoidal rule: $\displaystyle x \approx \Delta u \left[\sum_{1}^{16} f_i - 0.5(f_1 + f_{16}) \right]$

Ex: Extended Trapezoidal Rule in MATLAB®

Acceleration force = Aerodynamic resistance + Rolling resistance

$2000u(du/dx)=-8.1u^2-1200$

```
%Adopted from Nakamura, 2nd ed., pg.209
clear,
npoints=16;   i=1:npoints;
h=(30-15)/(npoints-1);
u=15+(i-1)*h;
f=2000*u./(8.1*u.^2+1200);
I=h*(sum(f)-0.5*(f(1)+f(length(f))))
```

I = 1.27504041491912e+002

Trapezoidal Rule

Trapezoidal Rule provides a reasonable approximation to a definite integral if large number of steps are taken

The error in the approximation originates in the fact that general graphs are curved and Trapezoidal rule approximates them by straight lines

An approximation, which takes into account the curvature of the graph, can also be formed: the result is a more efficient approximation called *Simpson's Rule.*

Simpson's Rule

- Simpson's Rule is formed by approximating a general curve by a parabola
- In this picture, the red graph is a parabola which approximates the yellow graph
- Remember: A parabola is the graph of a quadratic function

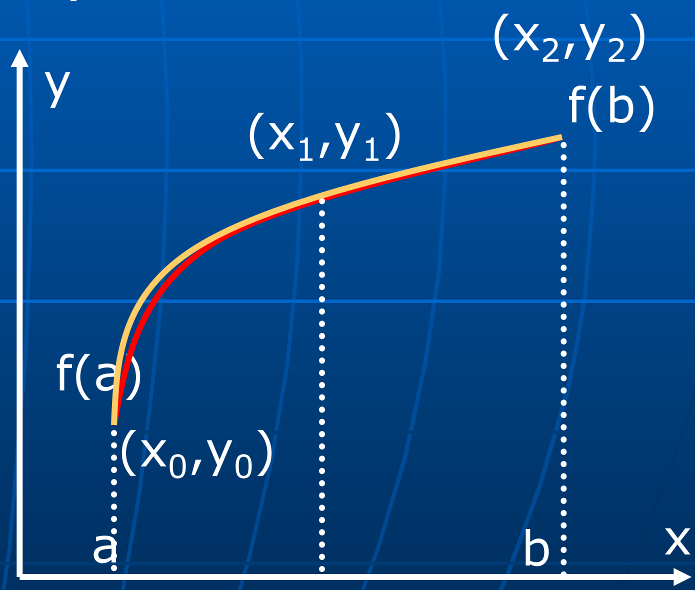
$y=ax^2+bx+c$

To find a, b and c

Three points on the function

(x_0,y_0), (x_1,y_1), (x_2,y_2)

need to be used to fix the parabola



Simpson's Rule

Approximating the function to be integrated by a quadratic polynomial gives the Basic Simpson's rule

For $y=ax^2+bx+c$ and (x_0,y_0), (x_1,y_1), (x_2,y_2)

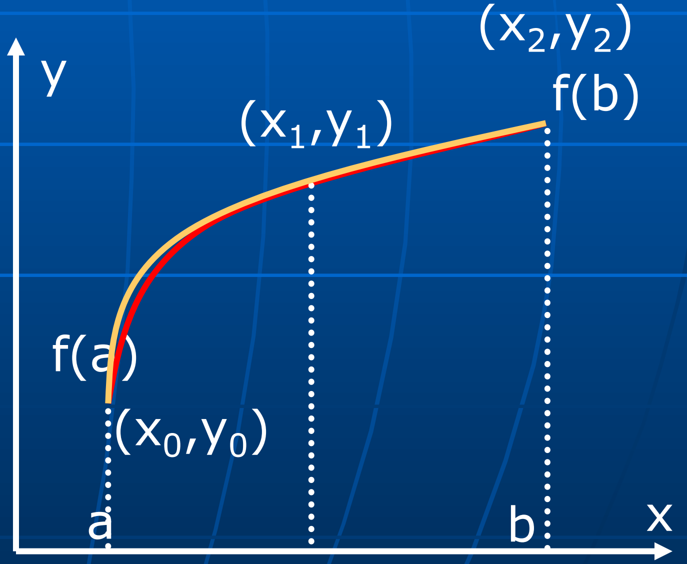
$$\text{let } h=\frac{b-a}{2}, \quad \text{and}$$

$$x_0=a, \quad x_1=x_0+h=\frac{b+a}{2}, \quad x_2=b,$$

$$y_i=f(x_i)$$

$$\int_a^b f(x)dx \approx \frac{h}{3}(y_0+4y_1+y_2)$$

$$=\frac{b-a}{6}\left[f(a)+4f\left(\frac{b+a}{2}\right)+f(b)\right]$$



Composite Simpson's Rule

Apply the idea of subdivision of intervals into n even number of intervals

$$\int_a^b f(x)dx = \int_a^{x_2} f(x)dx + \int_{x_2}^b f(x)dx$$

$$\approx \frac{h}{3}\left[f(a) + 4f\left(x_1\right) + f(x_2)\right] + \frac{h}{3}\left[f(x_2) + 4f\left(x_3\right) + f(b)\right]$$

or

$$\int_a^b f(x)dx \approx \frac{h}{3}\left[f(a) + 4f\left(x_1\right) + 2f(x_2) + 4f\left(x_3\right) + f(b)\right]$$

In general, for n even, h=(b-a)/n and Simpson's rule is given by:

$$\int_a^b f(x)dx \approx \frac{h}{3}\left[f(a) + 4f\left(x_1\right) + 2f(x_2) + 4f\left(x_3\right) + 2f\left(x_4\right) + \ldots + 2f\left(x_{n-2}\right) + 4f\left(x_{n-1}\right) + f(b)\right]$$

Ex: Approximating Pi/4

$$\text{Approximating } \pi/4: \qquad \int_0^1 \frac{1}{1+\mathrm{x}^2}\,dx = \arctan(1) = \frac{\pi}{4}$$

$$f(x) = \frac{1}{1+\mathrm{x}^2}, \qquad a = 0, \qquad \mathrm{b}=1, \quad \mathrm{h}=1/2$$

$$\int_0^1 \frac{1}{1+\mathrm{x}^2}\,dx \approx \frac{1}{6}\left[f(0) + 4f(\tfrac{1}{2}) + f(1)\right] = \frac{1}{6}\left[\frac{1}{1} + (4)\frac{4}{5} + \frac{1}{2}\right] = \frac{47}{60} \approx 0.78333$$

The exact solution for Pi/4 gives 0.78539816339745

Ex: Approximating Pi/4, Trapezoid

```
>> I=funtrapezoid(inline('(1+x^2)^-1'),0,1,4)
I =   0.78279411764706

function I=funtrapezoid(f,a,b,n)
%Finds integral of a function f on the interval [a,b]
%with n subintervals
%Adopted from Fausett 2nd Ed., pg.418
h=(b-a)/n;      S=f(a);
for i=1:n-1,
   x(i)=a+h*i;  S=S+2*f(x(i));
end
S=S+f(b);       I=h*S/2;
```

Ex: Approximating Pi/4, Composite Simpson

```
>> I=funsimpson(inline('(1+x^2)^-1'),0,1,4)
I =   0.78539215686274

function I=funsimpson(f,a,b,n)
%Finds integral of a function f on the interval [a,b]
%with n subintervals (n must be even)
%Adopted from Fausett 2nd Ed., pg.418
h=(b-a)/n;          S=feval(f,a);
for i=1:2:n-1,     x(i)=a+h*i;  S=S+4*feval(f,x(i));
end
for i=2:2:n-2,    x(i)=a+h*i;   S=S+2*feval(f,x(i));
end
S=S+feval(f,b);  I=h*S/3;
```

Newton-Cotes Open Formulas

Simplest examples of Newton-Cotes closed formulas → Trapezoid and Simpson rules: Use function evaluations at the end points of the interval of integration

The Midpoint Rule

→If we use function evaluations at points within the interval, say $x_m=(a+b)/2$, then we get the midpoint rule:

$$\int_a^b f(x)dx \approx (b-a)f\left(\frac{a+b}{2}\right)$$

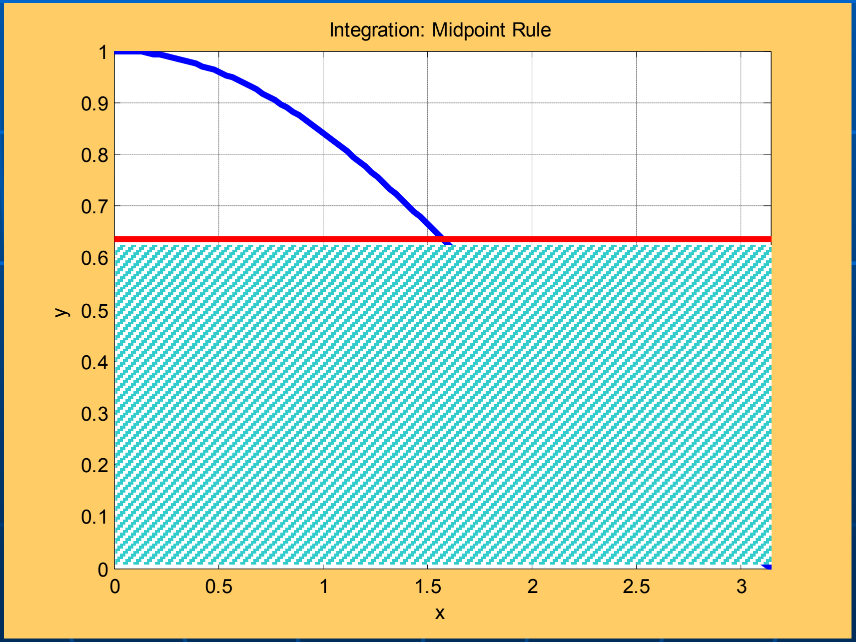
Assuming that f is twice continuously differentiable, the midpoint rule with error is given as:

$$\int_a^b f(x)dx = (b-a)f\left(\frac{a+b}{2}\right) + \frac{(b-a)^3}{24}f''(\eta), \quad \text{for some } \eta \in [a,b]$$

Ex: The Midpoint rule

Using the midpoint rule to to approximate the integral: $S = \int_0^{\pi} \frac{\sin(x)}{x} dx$

gives: $$\int_0^{\pi} \frac{\sin(x)}{x} dx \approx \pi \frac{\sin(\pi/2)}{\pi/2} = \pi \frac{1}{\pi/2} = 2$$



Derivatives

Numerical differentiation: Finding estimates for the derivative (slope) of a function by evaluating the function at only a set of discrete points

Simplest difference formulas to approximate the derivative of a function are based on using a straight line to interpolate the given data (i.e. using two data-points)

First derivative, forward difference formula $f'(x_i) \approx \dfrac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}$

First derivative, backward difference formula $f'(x_i) \approx \dfrac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}}$

First derivative, central difference formula $f'(x_i) = \dfrac{f(x_{i+1}) - f(x_{i-1})}{x_{i+1} - x_{i-1}}$

Second derivative, central difference formula $f''(x_i) = \dfrac{f(x_{i+1}) - 2f(x_i) + f(x_{i-1})}{h^2}$

$h = x_i - x_{i-1}$

Numerical Differentiation

Numerical differentiation is more difficult than numerical integration: Why?
→ Small changes in a function can create large changes in its slope

If data to be differentiated are obtained experimentally, the best approach is to:

- Find a least-squares fit to the data→ Use MATLAB®'s function polyfit(x,y,n) to find the coefficients of the polynomial of degree n that best fits the data in the least-squares sense
- Then differentiate the approximating function

MATLAB® Commands: Differentiation

P=polyfit(x,y,n)

%Finds the coefficients of the polynomial of degree n that best fits the data in the least squares sense

polyval(P,x)

%evaluates the polynomial P at x

polyder(P)

%differentiates polynomial P

diff(x)

%forward or backward difference approximation to dy/dx

MATLAB® Commands: Integration

trapz(x,y)

%uses composite trapezoid rule for the data points given in vectors x and y (with unit spacing)

To compute the integral for spacing other than one, multiply Z by the spacing increment

Input Y can be complex

- If Y is a vector, trapz(Y) is the integral of Y.
- If Y is a matrix,trapz(Y) is a row vector with the integral over each column.
- If Y is a multidimensional array, trapz(Y) works across the first nonsingleton dimension.

Ex: trapz

On a uniformly spaced grid:

```
>> X1 = 0:pi/100:pi;    Y1 = sin(X1);
>> Z1=trapz(X1,Y1),    Z2= pi/100*trapz(Y1)
Z1 =  1.99983550388744
Z2 =  1.99983550388744
```

Creating a nonuniformly spaced grid:

```
>> X = sort(rand(1,101)*pi);   Y = sin(X);
>> Z = trapz(X,Y);
Z =  1.99806848802083
```

The result is not as accurate as the uniformly spaced grid

MATLAB® Commands: Integration

quad	Use adaptive Simpson quadrature
quadl	Use adaptive Lobatto quadrature
quadv	Vectorized quadrature
dblquad	Numerically evaluate double integral
triplequad	Numerically evaluate triple integral

MATLAB® Commands: Integration

Q=quad(‘f’,xmin,xmax)

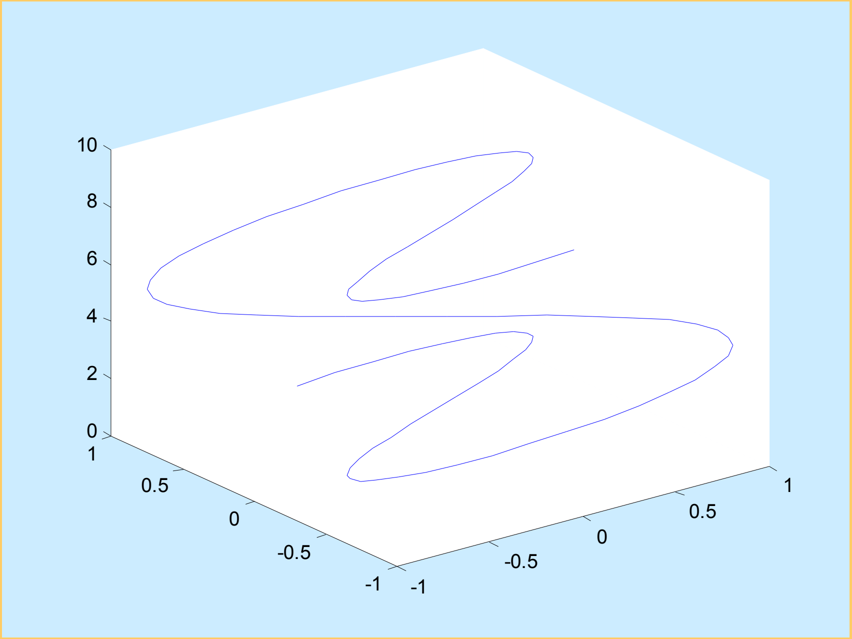
Q=quadl(‘f’,xmin,xmax)

%evaluate function f at whatever points are necessary to achieve accurate results

‘f’ is a string containing the name of the function

Ex: Quad

```
% using quad or quadl to
% compute the length of a curve
t = 0:0.1:3*pi;
% plot of the parameterizing
% equations gives:
plot3(sin(2*t),cos(t),t)
% The arc length formula says the
% length of the curve is the integral
% of the norm of the derivatives of
% the parameterized equations
f = inline('sqrt(4*cos(2*t).^2+sin(t).^2+1)');
% Integrating this function with
% a call to quad
len = quad(f,0,3*pi)
```



len = 17.22203188956838

MATLAB® Commands: Double Integration

```
q = dblquad(fun,xmin,xmax,ymin,ymax)
q = dblquad(fun,xmin,xmax,ymin,ymax,tol)
q = dblquad(fun,xmin,xmax,ymin,ymax,tol,method)
```

- calls the quad function to evaluate the double integral fun(x,y) over the rectangle xmin <= x <= xmax, ymin <= y <= ymax
- fun is a function handle
- method specifies the quadrature function, instead of the default quad. Valid values for method are @quadl or the function handle of a user-defined quadrature method that has the same calling sequence as quad and quadl

MATLAB® Commands: Double Integration

```
>>dblquad(@(x,y)sqrt(1-(x.^2+y.^2)).*(x.^2+y.^2<=1),-1,1,-1,1)
ans =    2.0944

>> F = @(x,y)y*sin(x)+x*cos(y);
>>Q = dblquad(F,pi,2*pi,0,pi)
Q =   -9.8696
```