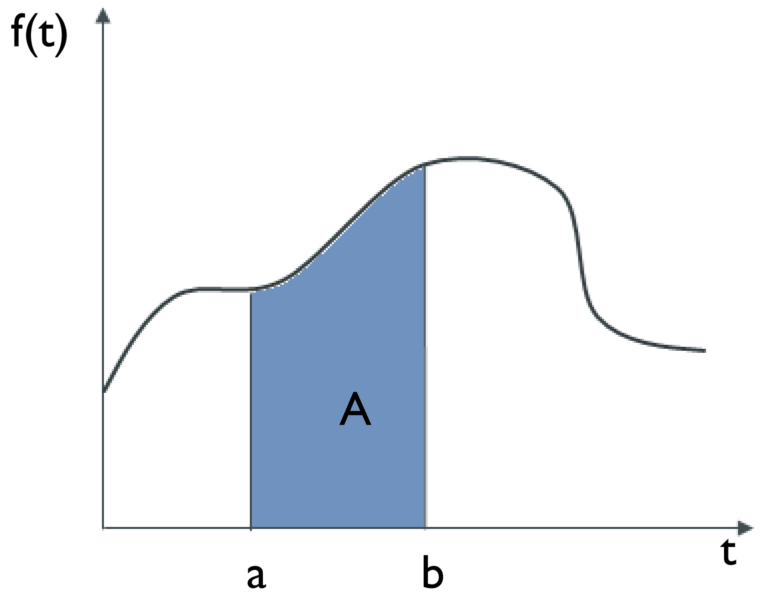
PROGRAMMING WITH MATLAB

WEEK 11

NUMERICAL METHODS FOR DIFFERENTIAL EQUATIONS

INTEGRATION

- The integral of a function is interpreted as the area under the function curve. In this case, the integral of f (t) is the area A under the curve of f(t) from t = 0 to t = b.



INTEGRATION

- To calculate the integral of a function, MATLAB has numerical integration functions.

quad: is a numerical method used to find the area under the graph of a function.

Syntax: q = quad(fun,a,b), uses recursive adaptive Simpson's rule to compute the integral of function fun from a to b. fun is a function handle. Limits a and b must be finite.

To compute the integral:

$$\int_{0}^{3} \frac{x}{x^2+3}\,dx$$

```
>> f =@(x)x./(x.^2+x);
>> q = quad(f,0,3)
q =
   1.3863
```

INTEGRATION

integral:

Syntax: q = integral(fun,xmin,xmax), numerically integrates function fun from xmin to xmax using global adaptive quadrature.

To compute the integral:

$$\int_0^{\infty} e^{-x} cosx dx$$

```
>> fun = @(x)exp(-x).*cos(x);

>> q = integral(fun,0,Inf)

q =

    0.5000
```

INTEGRATION

trapz:

Syntax: q = trapz(y), returns the approximate integral of Y via the trapezoidal method with unit spacing.

q = trapz(x,y) integrates y with spacing increment x

```
>> y = (1:7).^3    % y contains function values f(x)=x^3 in the domain [1,7]
y =
    1    8    27    64   125   216   343
>> q = trapz(y)
q =
   612
```

To compute the integral:

$$\int_{0}^{\pi/2} cosx dx$$

```
>> x = 0:pi/100:pi/2;
>> y = cos(x);
>> q = trapz(x,y)
q =
    0.9999
```

INTEGRATION

Double integral:

Syntax: q = integral2(fun,xmin,xmax,ymin,ymax), numerically integrates function fun(x,y).

To compute the integral of the function f(x,y):

$$f(x,y) = \frac{1}{\sqrt{x^2 + y}}$$

```
>> fun = @(x,y) 1./( sqrt(x.^2 + y));
>> q = integral2(fun,1,2,0,3)
q =
    1.6035
```

INTEGRATION

Triple integral:

Syntax: q = integral3(fun,xmin,xmax,ymin,ymax,zmin,zmax), numerically integrates function fun(x,y,z).

To compute the integral of the function f(x,y,z):

$$f(x,y,z) = \frac{1}{x^2 + y^2 + z^2}$$

```
>> fun = @(x,y,z) 1./(x.^2 + y.^2 + z.^2);

>> q = integral3(fun,0,1,0,2,0,3)

q =

    3.1482
```

DIFFERENTIATION

- The diff function of MATLAB gives approximate derivative calculations.

Syntax: y = diff(x), calculates differences between adjacent elements of x.

If x is a vector of length k, then y = diff(x) returns a vector of length k-1

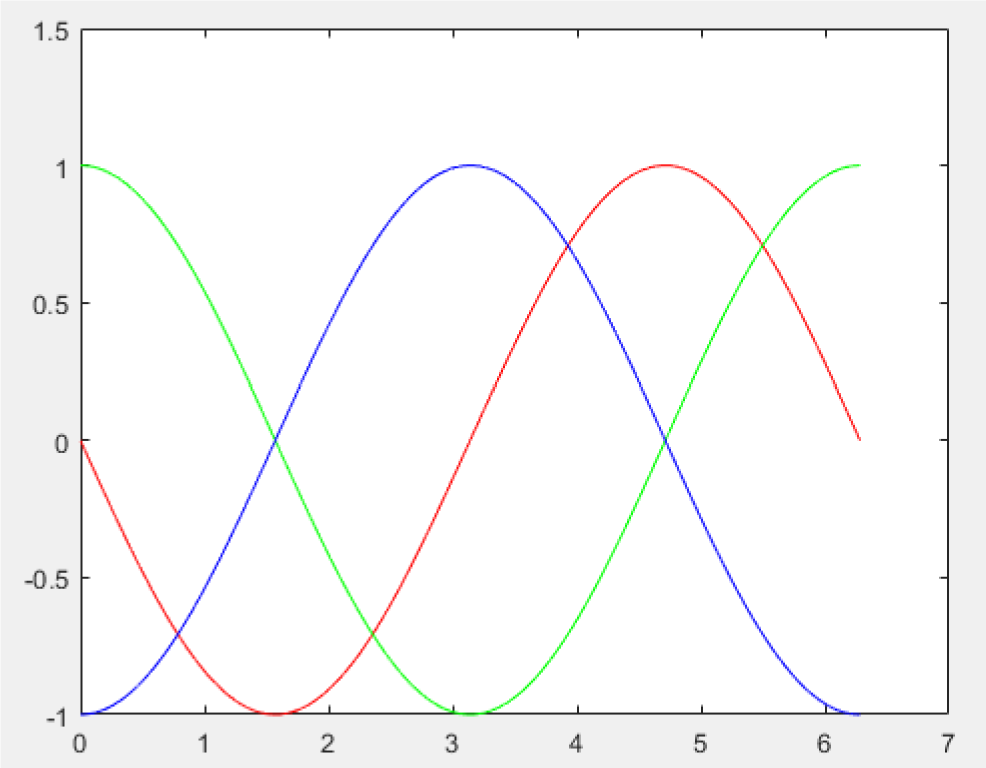
y = [x(2) - x(1), x(3) - x(2), . . . , x(k) - x(k -1)]

```
>> x = [1 3 5 7 11 17 13 19 23 23];

>> y = diff(x)

y =

     2     2     2     4     6    -4     6     4     0
```

DIFFERENTIATION

- Approximate derivatives with diff

```
>> d = 0.0001;   % step size
>> x = 0:d:2*pi;
>> fun = cos(x);
>> y = diff(fun)/d; % first derivative
>> z = diff(y)/d;   % second derivative
>> plot(x(:,1:length(y)),y,'r', x,fun,'g', x(:,1:length(z)),z,'b')
```



DIFFERENTIATION

- Numerical gradient.

Syntax: dfdx = gradient(f), returns the one-dimensional numerical gradient of vector f. The output dfdx corresponds to ∂d/∂x, which are the differences in the x (horizontal) direction.

[dfdx, dfdy] = gradient(f,dx,dy), computes the gradient of the function f(x,y), where dfdx and dfdy represent the partial derivatives, and dx, dy represent the spacing.

DIFFERENTIATION

- First-Order Differential Equations.

The ode45 function of MATLAB can be used to solve the equation dy/dt = f(t, y)

Syntax: [t,y] = ode45(odefun,tspan,y0), where tspan = [t0 tf], integrates the system of differential equations dy/dt=f(t,y) from t0 to tf with initial conditions y0. the inputs of the odefun must be t and y, and its output must be a column vector representing dy/dt. The number of rows in this column vector must equal the order of the equation.

DIFFERENTIATION

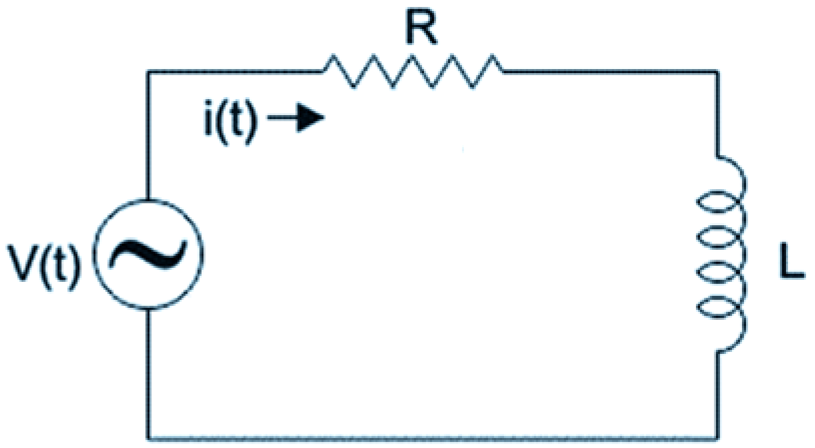
- An application of ode solver. RL circuit

the equation describing the response of this system from

an initial state of zero current is as follows:

$$L\frac{di}{dt} + Ri = V$$

$$\frac{di}{dt} + \frac{R}{L}i = \frac{V}{L}$$



Analytical solution:

$$i = \frac{V}{R}\left(1 - e^{-\frac{R}{L}t}\right)$$

DIFFERENTIATION

For R = 1, L = 1 ve V = 1, we define the following function:

```
function [ dydt ] = RLCircuit( t, y )
dydt = 1-y;
end
```

Calculate the analytical solution:

```
>> yTrue = 1-exp(-t);
```

Calculate with ode45 function:

```
>> [t, y] = ode45(@RLCircuit, [0, 5], 0);
```

plot two together:

```
>> plot(t,y,'*',t,yTrue), xlabel('Time (s)'), ylabel('Inductor Current')
```

DIFFERENTIATION