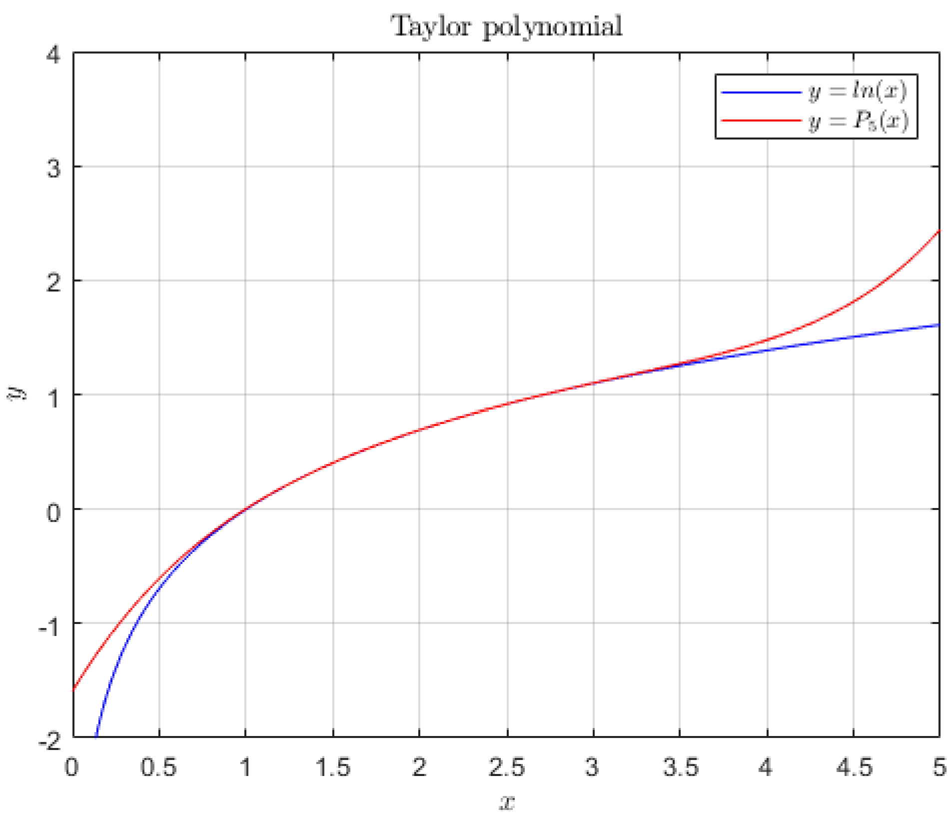
# Generate Taylor Series with User-defined Function: TaylorSeriesFUN

```
%% Generate Taylor Series With User-defined Function: TaylorSeriesFUN

clear, clc, close all
format rational % For dispalying fraction
format compact

syms x
f = log(x); % Define a function symbolically
aValue = 2;
n = 5;
xValue = 1;

disp('Note: TaylorEXP(aValue,xValue) The function inputs must be aValue first then xValue')
disp(['List of the coefficients from C_0 to C_',num2str(n)])

[C,TaylorEXP,Err] = TaylorSeriesFUN(f,aValue,n,xValue) % Call out the function

fprintf('The abosulte error is %f\n',double(Err))

% Plot the function and the taylor polynomial
figure(1)
fplot(f,'b')
hold on
s=linspace(0,5,1e5); % range might vary from different function
t=TaylorEXP(aValue,s);
ylim([-2,4])
grid on
plot(s,t,'r')
title('Taylor polynomial','Interpreter','latex')
xlabel('$x$','Interpreter','latex')
ylabel('$y$','Interpreter','latex')
legend('$y=ln(x)$','$y=P_5(x)$','interpreter','latex')


```

# The Output

```
Note: TaylorEXP(aValue,xValue) The function inputs must be aValue first
      then xValue

List of the coefficients from C_0 to C_5

C =
   Columns 1 through 6
      1588/2291     1/2      -1/8        1/24       -1/64       1/160

TaylorEXP =

  function_handle with value:

   @(a,x)a.*(-1.0./2.0)+x./2.0-(a-x).^2./8.0-(a-x).^3./2.4e+1-(a-x).^4./6.4e
+1-(a-x).^5./1.6e+2+6.931471805599453e-1

Err =
  61/13245

The absolute error is 0.004606
```




*Published with MATLAB® R2023a*

# The Function: TaylorSeriesFUN

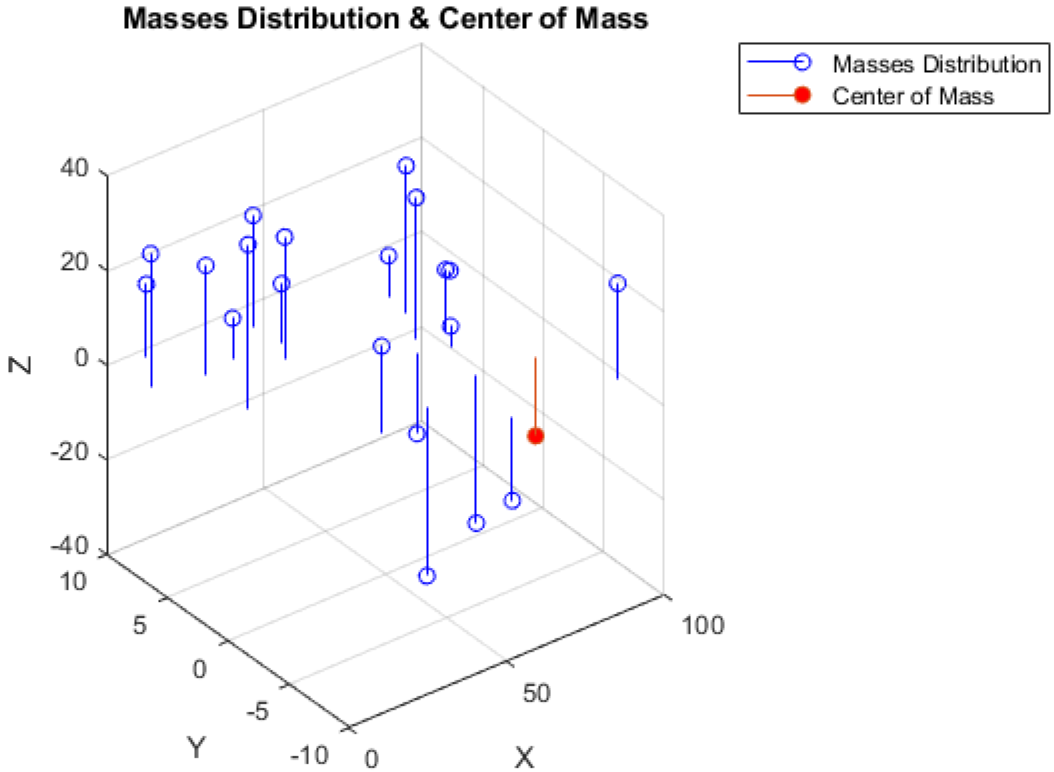
```
function [C,TaylorEXP,Err] = TaylorSeriesFUN(f,aValue,n,xValue)
% This function grnerates taylor series of any given function

%% Inputs & Outputs:

% f: function (symbolic math)
% aValue: point at which the function based on
% n: number of terms needed
% xValue: a point for test out y value

% C: list of coefficient from C_0 to C_n
% TaylorEXP: TaylorEXP(aValue,xValue) function handle
% Err: the aboslute error between f(xValue) and TaylorEXP(aValue,xValue)

%% Function

syms x a

C = zeros(1,n);      % preallocate coefficient
N = sym('x',[1 n]); % preallocate symbolic matrices

for i = 0:1:n
    dnfsym = diff(f,x,i); % find the (i)th derivative: f'n(x)
    dnfasym = subs(dnfsym,x,aValue); % substitute x with aValue to evaluate f'n(a)
    C(1,i+1) = double(dnfasym)/factorial(i); % assign coefficients: [C_0,C_1,C_2,...,C_n]
    N(1,i+1) = (x-a)^i; % Create terms:[(x-a)^0,(x-a)^1,(x-a)^2,...(x-a)^n]

end

TaylorEXP = matlabFunction(sum(C.*N)); % Combine terms and coefficient and put it into function
F = matlabFunction(f); % Turn the original function from 'sym' into function handle
Err = abs(F(xValue)-TaylorEXP(aValue,xValue)); % determine the absolute error

end
```

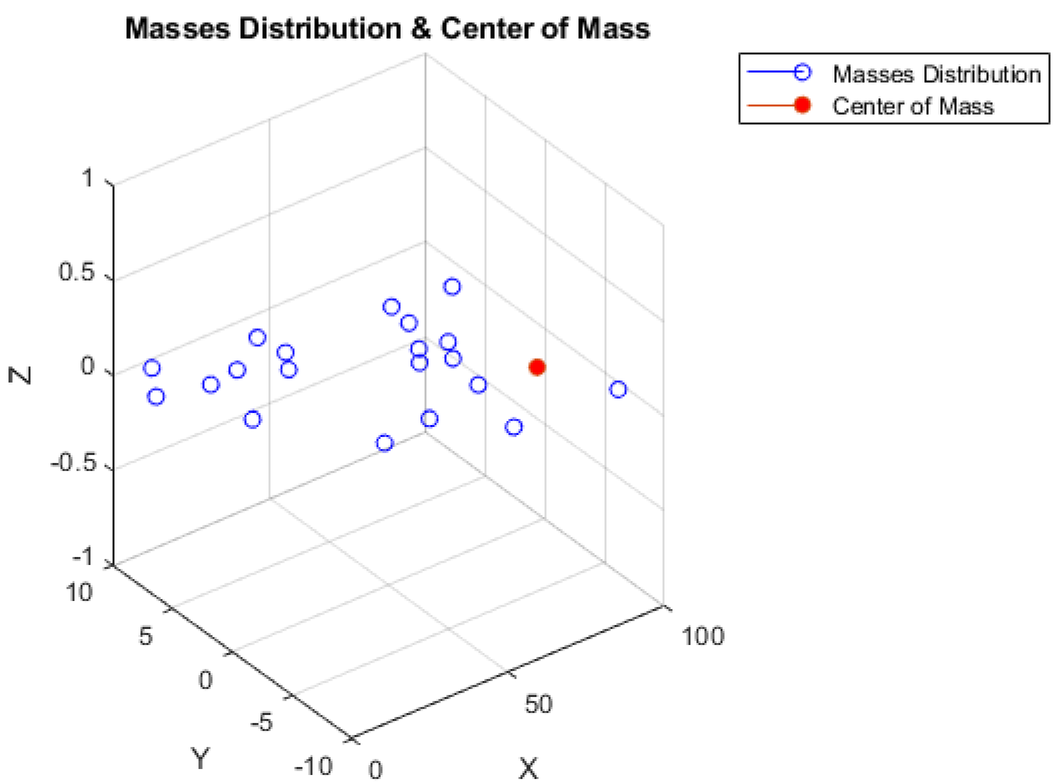
# Calculates Center of Mass for Particles with User-defined Function: MassCenterSUM

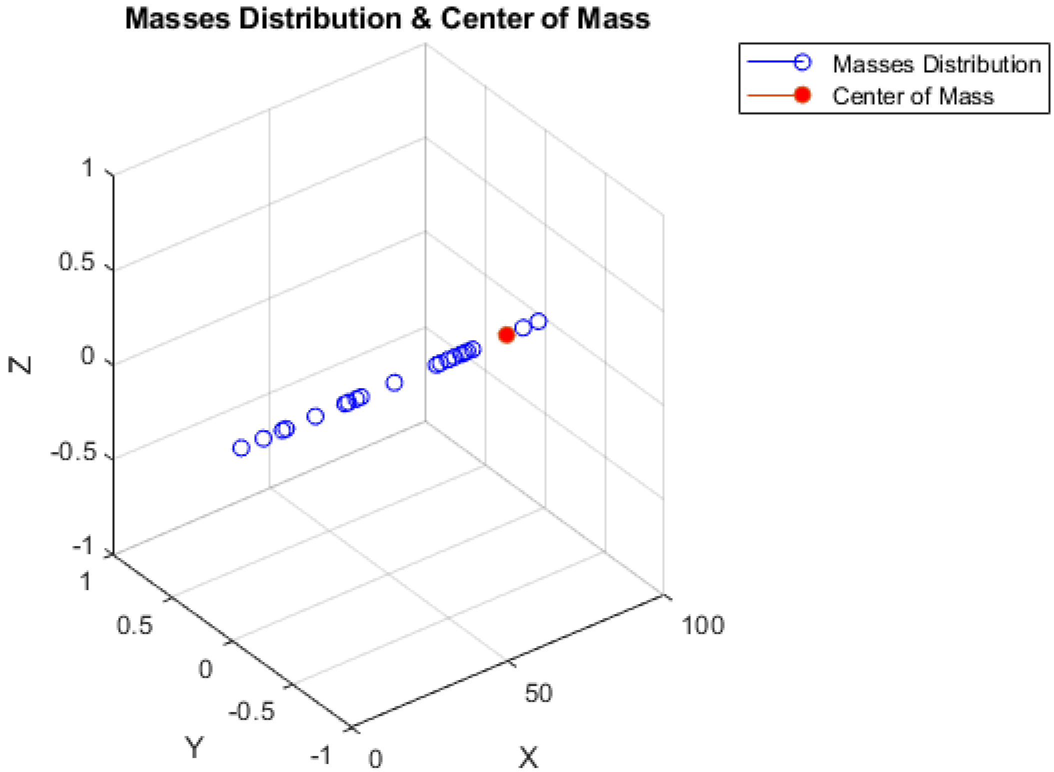
```
%% Calculates Center of Mass for Particles with User-defined Function: MassCenterSUM

clear, clc, close all
M = readmatrix('MassData1.xlsx'); % Import data with Excel file and turn it into a matrix
M = rmmissing(M,1); % remove non number parts of data
xi = M(:,1); % Seperate the matrix into column sets
yi = M(:,2);
zi = M(:,3);
mi = M(:,4);

[COM] = MassCenterSUM(xi,yi,zi,mi); % Run the function and test with different number input variables
sprintf('The center of mass is at x = %+.3f, y = %+.3f, z = %+.3f',COM(1,1),COM(1,2),COM(1,3))
[COM] = MassCenterSUM(xi,yi,mi);
sprintf('The center of mass is at x = %+.3f, y = %+.3f',COM(1,1),COM(1,2))
[COM] = MassCenterSUM(xi,mi);
sprintf('The center of mass is at x = %+.3f',COM(1,1))
[COM] = MassCenterSUM(mi);
sprintf(COM)
```

## The Output

```
ans =
    'The center of mass is at x = +87.833, y = -2.548, z = -16.672'
ans =
    'The center of mass is at x = +87.833, y = -2.548'
ans =
    'The center of mass is at x = +87.833'
ans =

    'Error'
```


Masses Distribution & Center of Mass
Masses Distribution
Center of Mass
40
20
0
-20
-40
Z
10
5
0
-5
-10
Y
0
50
100
X


Masses Distribution & Center of Mass
Masses Distribution
Center of Mass
1
0.5
0
-0.5
-1
Z
10
5
0
-5
-10
Y
0
50
100
X




*Published with MATLAB® R2023a*

## The Function: MassCenterSUM

```
function [COM] = MassCenterSUM(varargin)
% This function calculates center of mass for particles
% Input: ex. 4 column sets containing xi,yi,zi coordinates, and masses mi
% Output: A coordinate of the center of mass in a row vector.

% This function accepts different number of input variables for solving a
% task in one, two, and three dimensions.
% Whatever number of variables are taken, the order has to be xi,yi,zi,mi

if nargin == 4   % Determine number of input variables
    xi = varargin{1}; % Assign variables
    yi = varargin{2};
    zi = varargin{3};
    mi = varargin{4};
    Cx = sum(xi.*mi)/sum(mi); % Apply the formula
    Cy = sum(yi.*mi)/sum(mi);
    Cz = sum(zi.*mi)/sum(mi);
    COM = [Cx,Cy,Cz];

    fig = figure(1);
    movegui(fig,'northeast'); % Top right corner
    stem3(xi,yi,zi,'b')    % Display particles
    hold on
    stem3(Cx,Cy,Cz,'MarkerFaceColor','r')    % Display center of mass; mark red
    xlabel('X'); % Add labels, title, and legend
    ylabel('Y');
    zlabel('Z');
    title('Masses Distribution & Center of Mass');
    legend('Masses Distribution','Center of Mass');

elseif nargin == 3
    xi = varargin{1};
    yi = varargin{2};
    mi = varargin{3};
    Cx = sum(xi.*mi)/sum(mi);
    Cy = sum(yi.*mi)/sum(mi);
    COM = [Cx,Cy];

    fig = figure(2);
    movegui(fig,'east'); % Right center
    stem3(xi,yi,zeros(1,size(mi,1)),'b') % Assign zeros into zi
    hold on
    stem3(Cx,Cy,0,'MarkerFaceColor','r')
    xlabel('X');
    ylabel('Y');
    zlabel('Z');
    title('Masses Distribution & Center of Mass');
    legend('Masses Distribution','Center of Mass');

elseif nargin == 2
    xi = varargin{1};
    mi = varargin{2};
    Cx = sum(xi.*mi)/sum(mi);
    COM = Cx;
```

```
    fig = figure(3);
    movegui(fig,'southeast'); % Bottom right corner
    stem3(xi,zeros(1,size(mi,1)),zeros(1,size(mi,1)),'b') % Assign zeros into yi and zi
    hold on
    stem3(Cx,0,0,'MarkerFaceColor','r')
    xlabel('X');
    ylabel('Y');
    zlabel('Z');
    title('Masses Distribution & Center of Mass');
    legend('Masses Distribution','Center of Mass');

else
    COM = 'Error'; % Display 'Error' when number of varables out of range.
    % disp('The number of input varables is out of range.')
    % error('The number of input varables is out of range.')

end
```

# MassData1.xlsx

| Mi | Xi | Yi | Zi |
|---|---|---|---|
| 21.97321 | 5.922959 | 22.17562 | 1.1863 |
| 20.0634 | 2.462746 | 31.1204 | -7.45444 |
| 55.10689 | -7.93491 | 4.075427 | 6.917686 |
| 23.06616 | 6.130381 | 39.15273 | 11.22623 |
| 38.80016 | 8.79719 | -19.9899 | 8.350633 |
| 52.37725 | 1.977374 | 27.21079 | -3.47194 |
| 78.63589 | 0.941887 | 15.26774 | 15.99723 |
| 54.14278 | 6.958654 | 14.43024 | 9.478923 |
| 32.42348 | -2.98457 | -7.04078 | 7.585167 |
| 48.93565 | 5.033116 | 4.790579 | 11.93258 |
| 56.47665 | 7.213921 | 17.0424 | -1.4844 |
| 21.47604 | -4.9119 | -8.00789 | 1.29033 |
| 69.83339 | 4.607875 | 37.57876 | 14.36836 |
| 61.88257 | 0.114094 | 35.12619 | 10.85067 |
| 75.6802 | -2.71532 | -11.3361 | 15.43013 |
| 53.11947 | 0.274265 | 19.00312 | 23.10682 |
| 86.78038 | 8.591074 | 23.01698 | -17.6904 |
| 88.03405 | -6.01695 | 12.64197 | 5.931594 |
| 80.40724 | 8.203654 | 9.429478 | 12.28218 |
| 6.30702 | 1.4606 | 9.256099 | -5.58828 |