《计算流体力学基础》第二次作业

选择不同的计算一阶导数和二阶导数的数值格式，选择不同的函数 f(x)，在不同的网格尺度 h 上计算一阶导数和二阶导数的误差，并分析格式的舍入误差和截断误差的规律。

姓名：杨阳
学号：150*******
专业：流体力学

建议：

1. 初始网格划分取为 8 分，每次将网格加密一倍，总共加密 14 次，即网格数为 $n=8;16;32;...;131072$。如果计算资源允许，将所有可能的 n 都计算一遍。

2. 误差定义为节点上的平均绝对误差，记为 E，绘制 lnE-lnh 的曲线图。

3. 根据所用格式的精度，可以首先选择一个适当阶数的多项式，此时无截断误差，只有舍入误差；然后，再选择一个任意函数（建议 $f(x)=\sin(k_1 x)+\cos(k_2 x)$，选择合适的 k_1 和 k_2），其导数可以精确计算，此时截断误差和舍入误差都起作用，分析误差对网格尺度的依赖关系。

4. 采用单精度和双精度分别计算。

问题的求解：

对于一阶导数的计算选择后向一阶差分和二阶中心差分的计算格式，对于二阶导数的计算采用后向一阶差分和二阶中心差分格式，并比较不同差分格式对不同函数的导数的数值计算效果。

差分格式列表

$\frac{\partial f}{\partial x}$	一阶后向差分	$\frac{f_i - f_{i-1}}{\Delta x}$
	二阶中心差分	$\frac{f_{i+1} - f_{i-1}}{2\Delta x}$
$\frac{\partial^2 f}{\partial x^2}$	一阶后向差分	$\frac{f_i - 2f_{i-1} + f_{i-2}}{\Delta x^2}$
	二阶中心差分	$\frac{f_{i+1} - 2f_i + f_{i-1}}{\Delta x^2}$

计算函数列表：

一次函数	$f(x)=x+1 \qquad x\in[0,\ 10]$
二次函数	$f(x)=x^2+x+1 \qquad x\in[0,\ 10]$
指数函数	$f(x)=e^x \qquad x\in[0,\ 10]$
三角函数	$f(x)=sin(5x)+cos(10x) \qquad x\in[0,\ 10]$

为了方便绘图展示结果，并且计算量并不大，所以采用 MATLAB® 软件完成程序设计的计算。

根据数值格式精度的事后估计方法有：

$$\|u_e-u_h\|=Ch^p \qquad \|u_e-u_{2h}\|=C\left(2h\right)^p=2^pCh^p$$

$$\log_2\left(\frac{\|u_e-u_{2h}\|}{\|u_e-u_h\|}\right)=\log_2\left(2^p\right)=p$$

用这样的方法就能估计出数值格式精度 P，但是由于舍入误差的影响实际上 P 并不是精确的，但可以用回归分析去估计。

如果我们令

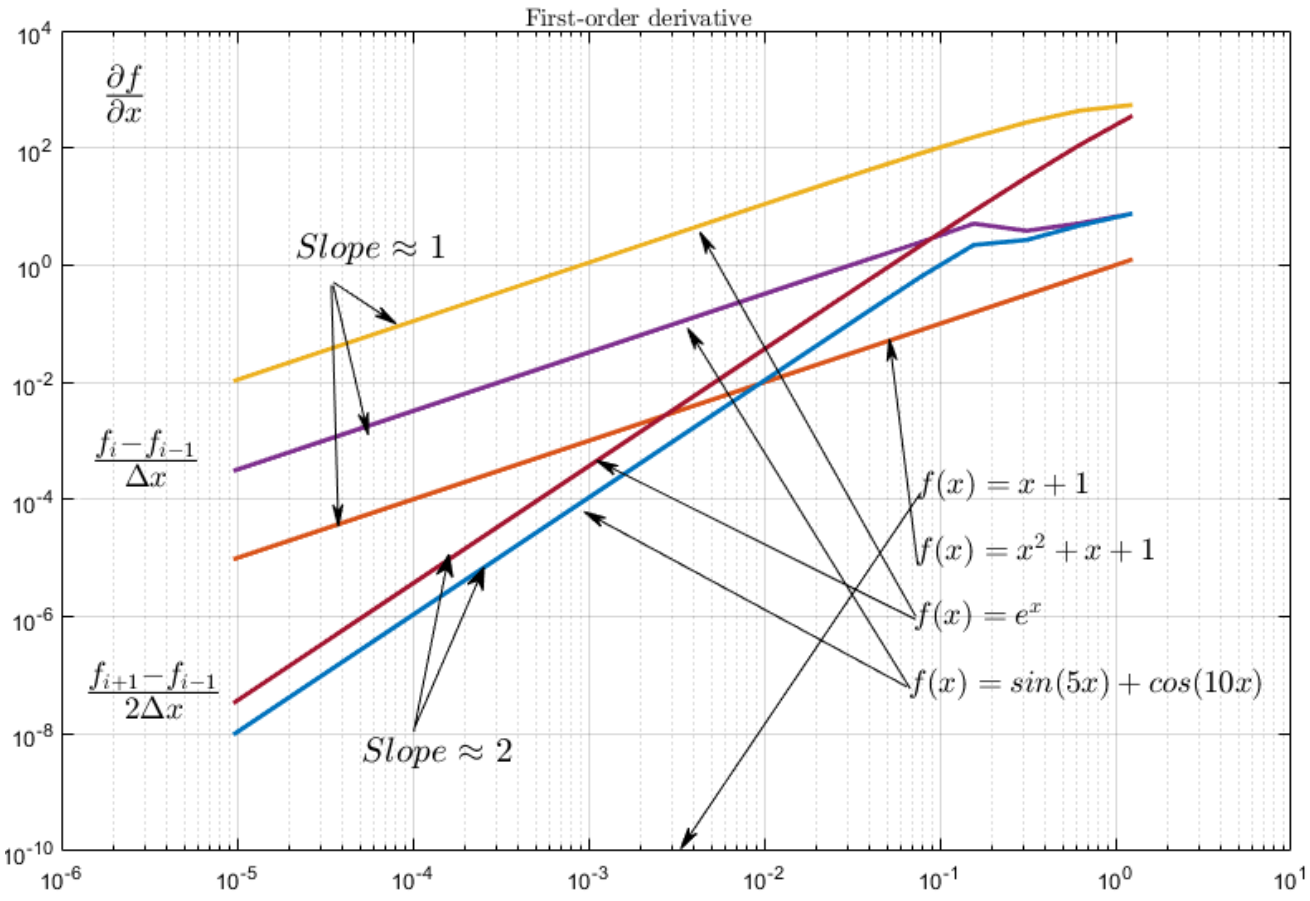
$$E_i=\sum_i\left\|u_e-u_{h_i}\right\|_1/N$$

那么通过做出 $\lg(h_i)-\lg(E_i)$ 图并分析曲线大概的斜率就可以知道格式精度的阶数 P。

在具体程序设计过程里面考虑了不同网格密度下数据的相关性，具体的实现过程中可以分别去计算不同网格密度下面的结果，而不同网格密度之间的数据没有依赖关系，这样计算一个点存一个点的方式可以算很大的规模对储存空间要求低，但是会耗费很大计算量。这里采用计算最密集情况下网格点上的函数值，其他网格密度下需要用函数值直接根据网格密度之间步长之间的关系取用已经计算出的函数值，可以节省计算量和时间，但对储存要求较高。因为网格两等分加密的过程实际上是一个二叉树的结构。按照这样的方式设计出程序。

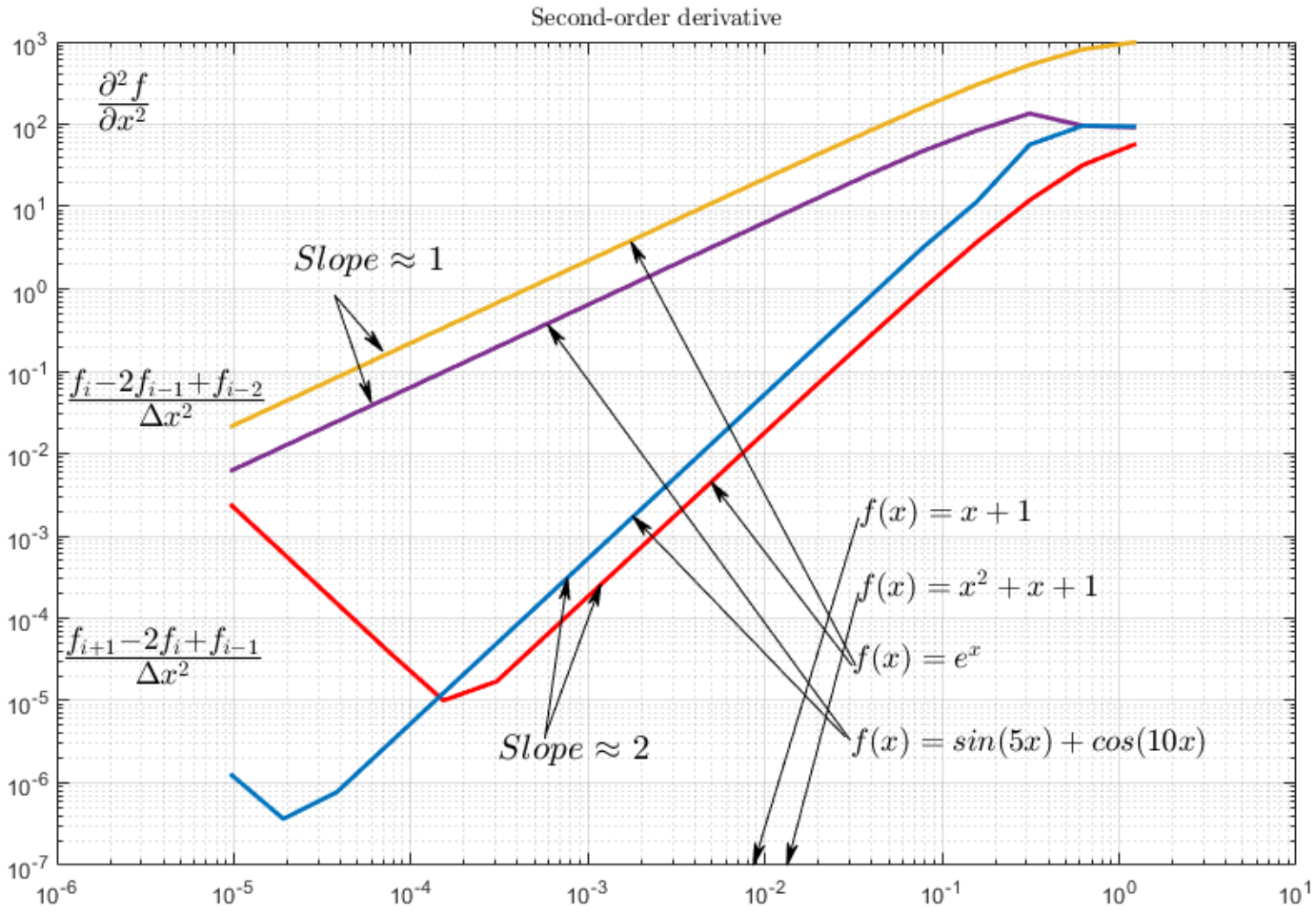
计算结果如下面两幅图所示，其中的第一幅图为一阶导数的计算结果图，第二幅

图为二阶导数计算的结果。



注意到因为有一些函数在相应的数值格式下面的截断误差是零，所以在对数坐标纵轴上对应的事负无穷，所以没有出现在图中。

一阶导数的数值格式中，一阶精度格式对于一次函数没有截断误差，对于二次函数和其他阶数更高的解析函数的截断误差和步长间的关系的曲线斜率大致为 1，也验证了一阶精度格式的精度是 P 阶的，特别注意到在网格较为稀疏的时候曲线有些偏离直线，因为这个时候函数在网格间隔内变化剧烈，三角函数震荡后剧烈这个现象明显，数值格式在步长大的时候没法捕捉到函数局部的特征。而二阶精度的格式对于小于或等于二次的多项式截断误差是零，所以图中表示不出来，而对于更高阶的解析函数截断误差和步长的关系曲线大概满足斜率是 2 的关系，也就验证了数值差分格的精度是 2 阶。



同样的对于二阶导数也和一阶导数类似，可以验证数值格式的精度的阶数，其结果如图所示，所不同的事我们看到当网格步长过小的时候反而会使得总的误差增加，这部分误差包括了截断误差和舍入误差两个部分，截断误差应按照理论预期变化，而没有按照预期变化的这个部分其实是由于舍入误差造成的。所以看出并不是步长越短，网格越细密越好，我们要综合考虑舍入误差和截断误差，也就是舍入误差和截断误差之间的博弈：步长越小截断误差越小，但是需要的步数越多需要的计算量越大，由于计算机字长的限制，每一次计算都会存在舍入误差，计算量越大舍入误差累计越严重。

因为 MATLAB 默认是双精度的，而强制单精度会比较麻烦，所以不做这种情况，但可以预料到用单精度舍入误差会更严重。实际上舍入误差和计算机位数有关系：

64 位计算机：

$$Machine\ epsilon = 2^{-52}$$

32 位计算机：

$$Machine\ epsilon = 2^{-20}$$

程序附录

RoundVSTruncation.m

```
%% Title
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Program name:RoundVSTruncation.m
%%%% Purpose: To compare Rounding Erro and Truncation Erro
%%%% Aurthor: Yang Yang
%%%% Date: 2015.10.20
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Control parameters
xspan = [0 10];          % Span of x
K = 18;                % Times of Change mesh number
N_0 = 8;               % Initial number of span split

%% Compute and statictics erro for first-order derivative
```

```
% using first-order back difference
format:
% \frac{f_{i} - f_{i-1}}{\Delta x}
E_11 = zeros(3,K);        % temporary array
to store mean absolute erro
h = zeros(1,K);
[E_11,h] = FOBDF1st(K,N_0,xspan);

% using second order centered difference
format:
% \frac{f_{i+1} - f_{i-1}}{2\Delta x}
E_12 = zeros(3,K);        % temporary array
to store mean absolute erro
[E_12,h] = SOCDF1st(K,N_0,xspan);


%% Compute and statistics erro for second-
order derivative

% using three point first-order back
difference format:
% \frac{f_{i} - 2f_{i-1} + f_{i-2}}{\Delta
x^{2}}
```

```
E_21 = zeros(3,K);        % temporary array
to store mean absolute erro
[E_21,h] = FOBDF2nd(K,N_0,xspan);

% using three point second order centered
difference format:
% \frac{f_{i+1} - 2f_{i} + f_{i-1}}{\Delta
x^{2}}
E_22 = zeros(3,K);        % temporary array
to store mean absolute erro
[E_22,h] = SOCDF2nd(K,N_0,xspan);


%% plot and compare details
% First order derivative
figureplot1(h, [E_11;E_12])

% Second order derivative
figureplot2(h, [E_21;E_22])

%% Output data
save Result K N_0 xspan h E_11 E_12 E_21
E_22
```

TestFunction.m

```
function y = TestFunction( x )
% This function is used to compute test
function's value
% here input x and y are column
y = zeros(4,length(x));
y(1,:) = x + 1;
y(2,:) = x.^2 + x + 1;
y(3,:) = exp(x);
y(4,:) = sin(5*x) + cos(10*x);

end
```

DTestFunction.m

```
function y = DTestFunction( x )
% This function is used to compute test
function's first order Derivative value
% here input x and y are column
y = zeros(4,length(x));
```

```
y(1,:) = 1;

y(2,:) = 2*x + 1;

y(3,:) = exp(x);

y(4,:) = 5*cos(5*x) - 10*sin(10*x);


end
```

DDTestFunction.m

```
function y = DDTestFunction( x )

% This function is used to compute test

function's second order Derivative value

% here input x and y are column

y = zeros(4,length(x));

y(1,:) = 0;

y(2,:) = 2;

y(3,:) = exp(x);

y(4,:) = -25*sin(5*x) - 100*cos(10*x);


end
```

FOBDF1st.m

```
function [E,h] = FOBDF1st(K,N_0,xspan)
% This function is to compute Erro of each
mesh density
%% Initial Output
E = zeros(3,K);
h = zeros(1,K);
%% Span split
N = N_0*2^(K-1);                          %
number of span split
delta_x = (xspan(2) - xspan(1)) /
N;       % step size of x

%% nodes' position
x = zeros(1,N+1);                         %
Store noeds' position
x(1) = xspan(1);
for i = 2:N+1
   x(i) = x(i-1) + delta_x;                %
Nodes' position
end
```

```
clear i                                          %
free memory

%% Compute function value on this points
f = TestFunction( x );
df = DTestFunction( x );
%ddf = DDTestFunction( x );

%% Main loop to compute mean absolute erro
for k = 1:4
    step = 0;
  for i = 1:K                    % index i
represent mesh density level
    temp = 0;
    % normal case
    index = 2^(step);           % define
index, using index to replace 1 at each
step
    delta = delta_x*index;      % mesh
step size in current step
    counter = 0;                % to record
how many points we compute
    for j = 1:index:N           % index j
```

```
represent x mesh nodes
        if (j - index) >= 1
            temp = temp + abs( (f(k,j) -
f(k,j-index)) / delta - df(k,j) );  %FOBDF
            counter = counter + 1;
        end
    end
    % store
    E(k,i) = temp/counter;
    h(i) = delta;
    step = step + 1;
  end

end


end
```

SOCDF1st.m

```
function [E,h] = SOCDF1st(K,N_0,xspan)
```

```
% This function is to compute Erro of each
mesh density
%% Initial Output
E = zeros(3,K);
h = zeros(1,K);
%% Span split
N = N_0*2^(K-1);                          %
number of span split
delta_x = (xspan(2) - xspan(1)) /
N;       % step size of x

%% nodes' position
x = zeros(1,N+1);                         %
Store noeds' position
x(1) = xspan(1);
for i = 2:N+1
   x(i) = x(i-1) + delta_x;                %
Nodes' position
end
clear i                                   %
free memory

%% Compute function value on this points
```

```
f = TestFunction( x );
df = DTestFunction( x );
%ddf = DDTestFunction( x );

%% Main loop to compute mean absolute erro
for k = 1:4
    step = 0;
  for i = 1:K                    % index i
represent mesh density level
    temp = 0;
    % normal case
    index = 2^(step);            % define
index, using index to replace 1 at each
step
    delta = delta_x*index;       % mesh
step size in current step
    counter = 0;                 % to record
how many points we compute
    for j = 1:index:N            % index j
represent x mesh nodes
       if j >1 && j<N
            temp = temp +
```

```
abs( (f(k,j+index) - f(k,j-index)) /
(2*delta) - df(k,j) );   % SOCDF
            counter = counter + 1;
        end
    end
    % store
    E(k,i) = temp/counter;
    h(i) = delta;
    step = step + 1;
  end

end


end
```

FOBDF2nd.m

```
function [E,h] = FOBDF2nd(K,N_0,xspan)
% This function is to compute Erro of each
mesh density
```

```
%% Initial Output
E = zeros(3,K);
h = zeros(1,K);
%% Span split
N = N_0*2^(K-1);                                %
number of span split
delta_x = (xspan(2) - xspan(1)) /
N;       % step size of x

%% nodes' position
x = zeros(1,N+1);                               %
Store noeds' position
x(1) = xspan(1);
for i = 2:N+1
   x(i) = x(i-1) + delta_x;                      %
Nodes' position
end
clear i                                         %
free memory

%% Compute function value on this points
f = TestFunction( x );
%df = DTestFunction( x );
```

```
ddf = DDTestFunction( x );

%% Main loop to compute mean absolute erro
for k = 1:4
    step = 0;
  for i = 1:K                    % index i
represent mesh density level
    temp = 0;
    % normal case
    index = 2^(step);           % define
index, using index to replace 1 at each
step
    delta = delta_x*index;      % mesh
step size in current step
    counter = 0;                % to record
how many points we compute
    delta_square = delta^2 ;
    for j = 1:index:N           % index j
represent x mesh nodes
       if (j - index - index) >= 1
            temp = temp + ...
                abs( (f(k,j) - 2*f(k,j-
```

```
index) + f(k,j-index-index)) /
delta_square - ddf(k,j) );  %FOBDF
            counter = counter + 1;
        end
    end
    % store
    E(k,i) = temp/counter;
    h(i) = delta;
    step = step + 1;
  end

end


end
```

SOCDF2nd.m

```
function [E,h] = SOCDF2nd(K,N_0,xspan)
% This function is to compute Erro of each
mesh density
%% Initial Output
```

```
E = zeros(3,K);
h = zeros(1,K);
%% Span split
N = N_0*2^(K-1);                                %
number of span split
delta_x = (xspan(2) - xspan(1)) /
N;       % step size of x

%% nodes' position
x = zeros(1,N+1);                               %
Store noeds' position
x(1) = xspan(1);
for i = 2:N+1
   x(i) = x(i-1) + delta_x;                      %
Nodes' position
end
clear i                                         %
free memory

%% Compute function value on this points
f = TestFunction( x );
%df = DTestFunction( x );
ddf = DDTestFunction( x );
```

```
%% Main loop to compute mean absolute erro
for k = 1:4
    step = 0;
  for i = 1:K                    % index i
represent mesh density level
    temp = 0;
    % normal case
    index = 2^(step);           % define
index, using index to replace 1 at each
step
    delta = delta_x*index;      % mesh
step size in current step
    counter = 0;                % to record
how many points we compute
    delta_square = delta^2;
    for j = 1:index:N           % index j
represent x mesh nodes
       if (j - index) >= 1 && (j +
index )<=N
            temp = temp + ...
                abs( (f(k,j+index) -
```

```
2*f(k,j) + f(k,j-index)) / delta_square -
ddf(k,j) );  %SOCDF
            counter = counter + 1;
        end
    end
    % store
    E(k,i) = temp/counter;
    h(i) = delta;
    step = step + 1;
  end

end


end
```

Figureplot1.m

```
function figureplot1(X1, YMatrix1)
%CREATEFIGURE(X1, YMATRIX1)
%  X1:  vector of x data
%  YMATRIX1:  matrix of y data
```

```
%  Auto-generated by MATLAB on 24-Oct-2015
19:05:19

% Create figure
figure1 = figure('Color',[1 1 1]);
% Create axes
axes1 = axes('Parent',figure1);
hold(axes1,'on');

% Create multiple lines using matrix input
to loglog
loglog1 =
loglog(X1,YMatrix1,'Parent',axes1,'LineWid
th',2);
set(loglog1(1),'LineWidth',0.5);
set(loglog1(5),'LineWidth',0.5);
set(loglog1(6),'LineWidth',0.5);

% Create title
title({'First-order
derivative'},'FontSize',20,'Interpreter','
latex');
```

```
box(axes1,'on');
% Set the remaining axes properties
set(axes1,'XColor',[0 0
0],'XGrid','on','XMinorTick','on','XScale'
,'log','YColor',[0 0 0],'YGrid',...

'on','YMinorTick','on','YScale','log','ZCo
lor',[0 0 0]);
% Create textbox
annotation(figure1,'textbox',...
    [0.137222910216724 0.182608695652174
0.075625386996904 0.121739130434783],...
    'String',{'$\frac{f_{i+1} - f_{i-
1}}{2\Delta x}$'},...
    'Interpreter','latex',...
    'FontSize',20,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create textbox
annotation(figure1,'textbox',[0.1488328173
37462 0.8 0.030733746130031
```

```
0.095652173913043],...
    'String','$\frac{\partial f}{\partial
x}$',...
    'Interpreter','latex',...
    'FontSize',20,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create textbox
annotation(figure1,'textbox',...
    [0.141759967842915 0.452008106116434
0.0531795665634667 0.0826086956521745],...
    'String','$\frac{f_{i} - f_{i-
1}}{\Delta x}$',...
    'Interpreter','latex',...
    'FontWeight','bold',...
    'FontSize',20,...
    'FitBoxToText','off',...
    'EdgeColor',[1 1 1]);

% Create arrow
annotation(figure1,'arrow',[0.352486187845
304 0.374585635359116],...
```

```
    [0.228813559322034
0.404661016949153]);

% Create arrow
annotation(figure1,'arrow',[0.352486187845
304 0.396685082872928],...
    [0.230932203389831
0.391949152542373]);

% Create textbox
annotation(figure1,'textbox',...
    [0.656990113405068 0.235795873249816
0.094975232198141 0.065217391304348],...
    'String','$f(x) = sin(5x) +
cos(10x)$',...
    'Interpreter','latex',...
    'FontSize',14,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create textbox
annotation(figure1,'textbox',...
    [0.659974051964504 0.303574060427414
```

```
0.09497523219814
1
0.065217391304348],'String','$f(x) =
e^{x}$',...
    'Interpreter','latex',...
    'FontSize',14,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create textbox
annotation(figure1,'textbox',...
    [0.660748045772551 0.367041267501843
0.094975232198141 0.065217391304348],...
    'String','$f(x) = x^{2} + x + 1$',...
    'Interpreter','latex',...
    'FontSize',14,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create textbox
annotation(figure1,'textbox',...
    [0.662183996715883 0.432737656595431
0.09497523219814
1
0.065217391304348],'String','$f(x) = x +
```

```
1$',...
    'Interpreter','latex',...
    'FontSize',14,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create arrow
annotation(figure1,'arrow',[0.301657458563
536 0.323756906077348],...
    [0.671610169491525
0.523305084745763],'HeadWidth',6);

% Create arrow
annotation(figure1,'arrow',[0.301657458563
536 0.341436464088398],...
    [0.674847457627119
0.633474576271186],'HeadWidth',6);

% Create arrow
annotation(figure1,'arrow',[0.665193370165
746 0.524861878453039],...
    [0.271186440677966
0.629237288135593],'HeadWidth',6);
```

```
% Create arrow
annotation(figure1,'arrow',[0.669613259668
508 0.651933701657459],...
    [0.396186440677966
0.618644067796611],'HeadWidth',6);

% Create arrow
annotation(figure1,'arrow',[0.668508287292
818 0.532596685082873],...
    [0.343220338983051
0.724576271186441],'HeadWidth',6);

% Create arrow
annotation(figure1,'arrow',[0.668508287292
818 0.467403314917127],...
    [0.340101694915254
0.497881355932203],'HeadWidth',6);

% Create arrow
annotation(figure1,'arrow',[0.662983425414
365 0.459668508287294],...
    [0.273305084745763
0.447033898305085],'HeadWidth',6);
```

```
% Create textbox
annotation(figure1,'textbox',...
    [0.270613259668512 0.671610169491525
0.120546961325968 0.063559322033898],...
    'String','$Slope \approx 1$',...
    'Interpreter','latex',...
    'FontWeight','bold',...
    'FontSize',16,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create textbox
annotation(figure1,'textbox',...
    [0.312602209944755 0.171610169491525
0.120546961325968 0.063559322033898],...
    'String','$Slope \approx 2$',...
    'Interpreter','latex',...
    'FontWeight','bold',...
    'FontSize',16,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create arrow
```

```
annotation(figure1,'arrow',[0.300552486187
845 0.304972375690608],...
    [0.671610169491525
0.432203389830508],'HeadWidth',6);

% Create arrow
annotation(figure1,'arrow',[0.671823204419
89 0.520441988950276],[0.466101694915254
0.112288135593221],...
    'HeadWidth',6);
```

Figureplot2.m

```
function figureplot2(X1, YMatrix1)
%CREATEFIGURE(X1, YMATRIX1)
%  X1:  vector of x data
%  YMATRIX1:  matrix of y data

%  Auto-generated by MATLAB on 24-Oct-2015
19:06:19
```

```
% Create figure
figure1 = figure('Color',[1 1 1]);

% Create axes
axes1 = axes('Parent',figure1);
hold(axes1,'on');

% Create multiple lines using matrix input
to loglog
loglog1 =
loglog(X1,YMatrix1,'Parent',axes1);
set(loglog1(3),'LineWidth',2);
set(loglog1(4),'LineWidth',2);
set(loglog1(7),'LineWidth',2,'Color',[1 0
0]);
set(loglog1(8),'LineWidth',2);

% Create title
title({'Second-order
derivative'},'EdgeColor',[1 1
1],'FontSize',20,'Interpreter','latex');

box(axes1,'on');
% Set the remaining axes properties
```

```
set(axes1,'XGrid','on','XMinorTick','on','
XScale','log','YGrid','on','YMinorTick','o
n','YScale',...
    'log');
% Create textbox
annotation(figure1,'textbox',...
    [0.620195046439639 0.273913043478262
0.094975232198141
0.0652173913043483],'String','$f(x) =
e^{x}$',...
    'Interpreter','latex',...
    'FontSize',14,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create textbox
annotation(figure1,'textbox',...
    [0.619421052631585 0.191304347826087
0.094975232198140 7 0.0652173913043483],...
    'String','$f(x) = sin(5x) +
cos(10x)$',...
    'Interpreter','latex',...
```

```
    'FontSize',14,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create textbox
annotation(figure1,'textbox',...
    [0.145856353591161 0.804375352051225
0.0519337016574582 0.0992856648979276],...
    'String',{'$\frac{\partial
^{2}f}{\partial x^{2}}$'},...
    'Interpreter','latex',...
    'FontSize',20,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create textbox
annotation(figure1,'textbox',...
    [0.270613259668512 0.671610169491525
0.120546961325968 0.063559322033898 1],...
    'String','$Slope \approx 1$',...
    'Interpreter','latex',...
    'FontWeight','bold',...
    'FontSize',16,...
```

```
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create textbox
annotation(figure1,'textbox',...
    [0.398790055248626 0.188559322033898
0.120546961325968 0.063559322033898],...
    'String','$Slope \approx 2$',...
    'Interpreter','latex',...
    'FontWeight','bold',...
    'FontSize',16,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create textbox
annotation(figure1,'textbox',...
    [0.62317898499907 0.34373618275608
0.094975232198141 6 0.065217391304348],...
    'String','$f(x) = x^{2} + x + 1$',...
    'Interpreter','latex',...
    'FontSize',14,...
    'FitBoxToText','off',...
    'EdgeColor','none');
```

```
% Create arrow
annotation(figure1,'arrow',[0.627624309392
265 0.539226519337023],...
    [0.307203389830508
0.489406779661017],'HeadWidth',6);

% Create arrow
annotation(figure1,'arrow',[0.435359116022
1 0.470343392299688],[0.23728813559322
0.388157894736842],...
    'HeadWidth',6);

% Create arrow
annotation(figure1,'arrow',[0.435359116022
1 0.449531737773153],[0.235169491525424
0.394736842105263],...
    'HeadWidth',6);

% Create arrow
annotation(figure1,'arrow',[0.626430801248
699 0.437044745057232],...
    [0.233552631578947
0.644736842105263],'HeadWidth',6);
```

```
% Create arrow
annotation(figure1,'arrow',[0.629834254143
652 0.490738813735 69],...
    [0.307203389830509
0.726973684210526],'HeadWidth',6);

% Create textbox
annotation(figure1,'textbox',...
    [0.127685485308239 0.515304415700268
0.158116022099448 0.091101694915254 2],...
    'String','$\frac{f_{i} - 2f_{i-1} +
f_{i-2}}{\Delta x^{2}}$',...
    'Interpreter','latex',...
    'FontSize',20,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create textbox
annotation(figure1,'textbox',...
    [0.125604319855585 0.25635593220339
0.150381215469614 0.095338983050847 5],...
    'String',{'$\frac{f_{i+1} - 2f_{i} +
```

```
f_{i-1}}{\Delta x^{2}}$'},...
    'Interpreter','latex',...
    'FontSize',20,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create arrow
annotation(figure1,'arrow',[0.303867403314
918 0.334027055150885],...
    [0.673728813559322
0.618421052631579],'HeadWidth',6);

% Create arrow
annotation(figure1,'arrow',[0.303867403314
917 0.326742976066597],...
    [0.673728813559322
0.567434210526316],'HeadWidth',6);

% Create arrow
annotation(figure1,'arrow',[0.623309053069
719 0.490114464099896],...
    [0.236842105263158
0.455592105263158],'HeadWidth',6);
```

```
% Create textbox
annotation(figure1,'textbox',...
    [0.624614935942397 0.417907148120855
0.094975232198141
0.0652173913043482],'String','$f(x) = x +
1$',...
    'Interpreter','latex',...
    'FontSize',14,...
    'FitBoxToText','off',...
    'EdgeColor','none');

% Create arrow
annotation(figure1,'arrow',[0.631633714880
333 0.565745856353591],...
    [0.453947368421053
0.110169491525424],'HeadWidth',6);

% Create arrow
annotation(figure1,'arrow',[0.631633714880
333 0.586740331491713],...
    [0.379934210526316
0.110169491525424],'HeadWidth',6);
```