APPENDIX

MATLAB Primer and Code Listings

C

At several locations in the text, numerical methods were used to calculate and plot solutions to a variety of continuum mechanics problems. Although other options are available, the author has found MATLAB software well suited to conduct such numerical work. This particular software has the necessary computational and plotting tools to enable very efficient and simple applications. MATLAB is a professional engineering and scientific software package developed and marketed by MathWorks, Inc. In recent years, it has achieved widespread and enthusiastic acceptance throughout the engineering community. Many engineering schools now require and/or use MATLAB as one of their primary computing tools. Its popularity is due to a long history of well-developed and tested products, to its ease of use by students, and to its compatibility across many different computer platforms. The purpose of this appendix is to present a few MATLAB basics to aid the reader in applying particular software applications and to list several of the codes used in the text. The software package itself contains an excellent *Help* package that provides extensive information on various commands and procedures, and many books are available on the software package. Also much information can simply be found on the web itself through standard keyword search procedures. It is assumed that the reader has some prior computational background and experience and thus has a basic understanding of programming techniques.

C.1 GETTING STARTED

MATLAB is both a computer programming language and a software environment for using the language. Under the MS Windows Operating System, the MATLAB window will appear as shown in Fig. C.1. It is from this window header bar that the *Help* menu can be accessed and this provides extensive information on most topics. In this command window, the user can type instructions after the prompt " > >". However, it is of course much more efficient to create and save application programs within the *Editor* window. This window is activated by going to the File menu in the Command window and selecting either *New Script* to start a new creation or *Open* to open an existing file. MATLAB files are called m-files and have the extension *.m. Within the Editor window, a new application code can be created or an existing one

Continuum Mechanics Modeling of Material Behavior. http://dx.doi.org/10.1016/B978-0-12-811474-2.00012-5

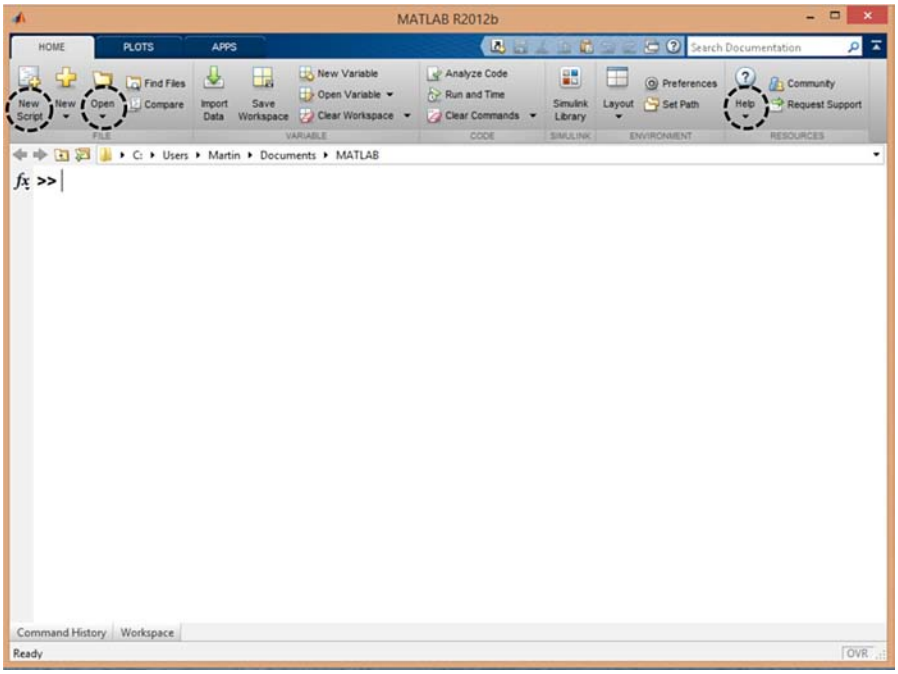

FIGURE C.1

MATLAB command window.

can be modified. In either case, the resulting file can then be saved for later use, and the current file can be run from this window. An example program appearing in the Editor window is shown in Fig. C.2.

C.2 EXAMPLES

We will not attempt a step-by-step explanation of various MATLAB commands, but rather will instead pursue a *learn-by-example* approach. In this fashion, most of the needed procedures will be demonstrated through the presentation of several example codes that have been previously used in the text. Listed codes typically illustrate: use of array I/O; standard calculation steps; various plotting and display schemes (Cartesian, polar, vector, etc.) with labeling methods; "for-end" looping; numerical integration; and a few other more specialized commands. Note that, in general, code lines preceded with a "%" symbol will not be executed and are used for comments to explain the coding. A semicolon ending a code line will suppress screen-printing of that particular calculation. The reader with previous programming experience should

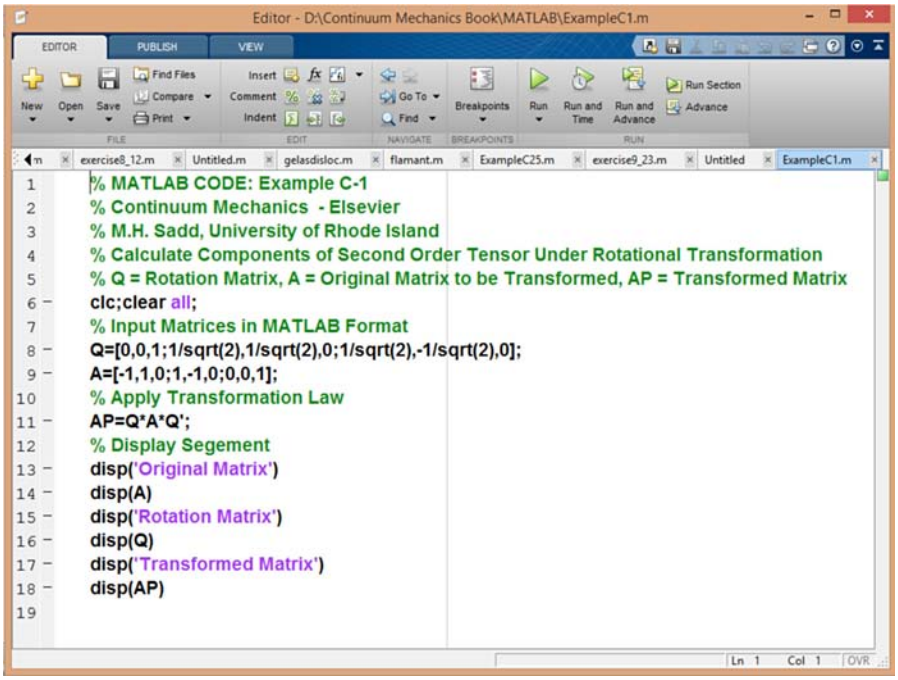

FIGURE C.2

MATLAB editor window.

be able to quickly review these examples and use them as is or make modifications to develop their own codes.

```
% MATLAB CODE: Example C-1
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Calculate Components of Second Order Tensor Under Rotational Transformation
% Q = Rotation Matrix, A = Original Matrix to be Transformed, AP = Transformed Matrix
clc;clear all;
% Input Matrices in MATLAB Format
Q=[0,0,1;1/sqrt(2),1/sqrt(2),0;1/sqrt(2),-1/sqrt(2),0];
A=[-1,1,0;1,-1,0;0,0,1];
% Apply Transformation Law
AP=Q*A*Q';
% Display Segement
disp('Original Matrix')
disp(A)
disp('Rotation Matrix')
disp(Q)
disp('Transformed Matrix')
disp(AP)
```

```
% MATLAB CODE: Example C-2
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Numerically Calculate Invariants, Principal Values
% and Directions of a Matrix
% Program Uses Matrix from Example 2.11.1
clc;clear all;
% Input Matrix
A=[-1,1,0;1,-1,0;0,0,1]
% Calculate Invariants
invariants=[trace(A),(trace(A)^2-trace(A*A))/2,det(A)]
[V,L]=eig(A);
% Principal Values are the Diagonal Elements of the L Matix
principal_values=[L(1,1),L(2,2),L(3,3)]
% Principal Directions are the Columns of the V Matrix
principal_directions=[V(:,1),V(:,2),V(:,3)]
```

```
% MATLAB CODE: Example C-3
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% 2-D Deformation Examples: Unit Square and Circle
clc;clear all;clf
% Insert Unit Reference Square
figure(1)
X=[0,1,1,0,0];Y=[0,0,1,1,0];
fill(X,Y,[.95 .95 .95],'linestyle','--','linewidth',2)
axis equal;
axis([-0.5 3 -0.5 3])
hold on, grid on
% Add Deformed Element
xx=1.2*X+0.2*Y+1.2;yy=1.2*Y+0.2*X+1.2;
fill(xx,yy,[.8 .8 .8],'linewidth',2)
% Deformation of Unit Circular Area
ang=0:0.01:2*pi; r=0.5;
X=r.*cos(ang)+0.5;
Y=r.*sin(ang)+0.5;
figure(2)
fill(X,Y,[.95 .95 .95],'linestyle','--','linewidth',2)
axis equal;
axis([-0.5 3 -0.5 3])
hold on, grid on
% Add Deformed Circular Element
xx=1.2*X+0.2*Y+1.2;yy=1.2*Y+0.2*X+1.2;
fill(xx,yy,[.8 .8 .8],'linewidth',2)
```

```
% MATLAB CODE: C-4
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Polar & Cartesian Plot Example 6.2.4 - Figure 6.13
clc;clear all;
% Input (r/a)- Variable and Generate Angular Coordinate Space
r=1;
t=[0:0.01:2*pi];
% Calculation Loop
st=0.5*(1+(1/r)^2)-0.5*(1+3*(1/r)^4)*cos(2*t);
% Plotting Call
figure(1)
polar(t,st,'k')
title('Non-Dimensional Hoop-stress Around Hole')
% Distance Decay Plot
r=[1:0.1:8];
st=0.5*(2+r.^(-2)+3*r.^(-4));
figure(2)
plot(r,st,'k','linewidth',2)
xlabel('Dimensionless Distance, \itr/\ita ')
ylabel('Dimensionless Stress, \itT_\theta_\theta/\itT')
grid on
```

```
% MATLAB CODE: C-5
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Flamant Problem Example 6.2.5 - Figure 6.15 Plots
clc;clear all;
%Plot Cartesian Stresses
x=[-5:0.1:5];
sy=-2./(pi*(x.^2+1).^2);
txy=-(2.*x)./(pi*(x.^2+1).^2);
figure(1);grid on;hold on
plot(x,sy,'k','linewidth',2)
plot(x,txy,'k','linewidth',2)
xlabel('Dimensionless Distance, \itr/\ita ')
ylabel('Dimensionless Stress')
%Plot Radial Stress Contours
[x,y]=meshgrid(-2:0.1:2,0:0.1:4);
r=sqrt(x.^2+y.^2);
t=asin(y./r);
sr=-(2/pi).*sin(t)./r;
figure(2)
y=-y;
contour(x,y,sr,35,'k','linewidth',1.5)
axis equal
title('\itT_r_r Contours - Flamant Problem')
```

```
% MATLAB CODE: C-6
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Uniform Potential Flow Problem Example 6.3.1 - Figure 6.16 Plot
clc;clf;clear all
[x,y]=meshgrid(0:0.1:1,0:0.1:1);
% Plot velocity vector field (V=1)
vx=1.*x./x;vy=0.*x./x;
figure(1)
nscale=0.08;
quiver(x,y,vx*nscale,vy,'k','linewidth',1.5,'AutoScale','off')
xlabel('x');ylabel('y')
axis([0,1.2,-0.2,1.2])
title('Velocity Field for Uniform Flow (0<x,y<1)')
% Plot Stream & Potential function contours
s=y;p=x;
figure(2)
contour(x,y,s,10,'k','linewidth',1.5)
xlabel('x');ylabel('y')
hold on; axis equal
contour(x,y,p,10,'k--','linewidth',1.5)
title('Stream & Potential Function Contours for Uniform Flow (0<x,y<1)')
```

```
% MATLAB CODE: C-7
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Potential Flow in a Corner Example 6.3.2 - Figure 6.18 Plots
clc;clf;clear all
[x,y]=meshgrid(0:0.1:1,0:0.1:1);
% Plot velocity vector field
vx=x;vy=-y;
figure(1)
quiver(x,y,vx,vy,'k','linewidth',1.5)
xlabel('x');ylabel('y')
title('Velocity Field for Flow in a Corner (0<x,y<1)')
% Plot Stream & Potential function contours
[x,y]=meshgrid(0:0.05:1,0:0.05:1);
s=x.*y;p=0.5*(x.^2-y.^2);
figure(2)
contour(x,y,s,20,'k','linewidth',1.5)
xlabel('x');ylabel('y')
hold on; axis equal
contour(x,y,p,20,'k--','linewidth',1.5)
title('Stream & Potential Function Contours for Corner Flow (0<x,y<1)')
```

```
% MATLAB CODE: C-8
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Potential Flow Around Cylinder Example 6.3.3 - Figure 6.20 Plots
% Must Add Cylinder by Hand
clc;clf;clear all
[x,y]=meshgrid(-4:1:4,-4:1:4);
[t,r]=cart2pol(x,y);
% Plot velocity vector field
```

```
vr=(1-(1./r).^2).*cos(t);
vt=-(1+(1./r).^2).*sin(t);
vx=vr.*cos(t)-vt.*sin(t);
vy=vr.*sin(t)+vt.*cos(t);
figure(1)
nscale=0.4
quiver(x,y,vx*nscale,vy*nscale,'k','linewidth',2,'AutoScale','off')
xlabel('x');ylabel('y');axis equal
title('Velocity Field for Flow Around a Cylinder')
% Plot Stream function contours
[x,y]=meshgrid(-4:0.2:4,-4:0.2:4);
[t,r]=cart2pol(x,y);
s=r.*(1-(1./r).^2).*sin(t);
figure(2)
contour(x,y,s,20,'k','linewidth',1.5)
xlabel('x');ylabel('y');axis equal
title('Stream Function Contours for Flow Around a Cylinder')


% MATLAB CODE: C-9
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Viscous Flow Examples 6.4.1, 6.4.2, 6.4.3 Figure Plots
clc;clf;clear all
% Example 6.4.1 Velocity Distribution Plot
[x,y]=meshgrid(0,-1:0.1:1);
vx=1-y.^2;vy=zeros(size(y),1);
figure(1)
quiver(x,y,vx,vy,'k','linewidth',2,'AutoScale','off')
title('Velocity Field for Plane Poiseuille Flow')
hold on
plot(vx,y,'k--','linewidth',2)
axis([-0.5,1.5,-1,1])
% Example 6.4.2 Velocity Distribution Plots
y=[0:0.05:1];
for P=[-8,-4,0,4,8]
vx=y+P*(1-y).*y;
figure(2)
plot(vx,y,'k','linewidth',2)
hold on
end
title('Velocity Field for Plane Couette Flow')
% Example 6.4.3 Velocity Distribution Plots
[t,r,z]=meshgrid(0:pi/8:pi,0:0.1:1,0);
[x,y,z]=pol2cart(t,r,z);
u=0*(x.^2+y.^2+z.^2);
v=0*(x.^2+y.^2+z.^2);
w=1-r.^2;
figure(3)
scale=1;
quiver3(x,y,z,u,v,w*scale,'k','linewidth',2,'AutoScale','off')
hold on
T=[0:pi/50:2*pi];R=ones(size(T));Z=zeros(size(T))
X=cos(T);Y=sin(T)
plot3(X,Y,Z,'k','linewidth',2)
axis off
title('Velocity Field for Hagen-Poiseuille Flow')
```

```
% MATLAB CODE: C-10
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Example 6.5.1 Figure Plots
clc;clf;clear all
t=[0:0.1:10];
for tau=[1,4,8]
  G=exp(-t/tau);
  figure(1)
  plot(t,G,'k','linewidth',2)
  hold on;grid on;
  xlabel('Time , t ')
  ylabel('Normalized Relaxation Function G(t) / E')
end
for tau=[1,4,8]
  J=(tau+t)/tau;
  figure(2)
  plot(t,J,'k','linewidth',2)
  hold on;grid on;
  xlabel('Time , t ')
  ylabel('Normalized Creep Function J(t) E')
end
```

```
% MATLAB CODE: C-11
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Example 6.5.2 Figure Plots
clc;clf;clear all
t=[0:0.1:10];
G=ones(length(t));
figure(1)
plot(t,G,'k','linewidth',2)
hold on;grid on;
xlabel('Time , t ')
ylabel('Normalized Relaxation Function G(t) / E')
for tau=[1,4,8]
  J=1-exp(-t/tau);
  figure(2)
  plot(t,J,'k','linewidth',2)
  hold on;grid on;
  xlabel('Time , t ')
  ylabel('Normalized Creep Function J(t) E')
end
```

```
% MATLAB CODE: C-12
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Example 6.5.4 Kelvin Model - Constant Loading Rate
% Solutions & Plots Using Numerical ODE Integrator
clc;clear all;clf
% Input Model Parameters
```

```
eta=100;E=50;
for R=[10,20,40]
% Use Anonymous Function
ode1=@(t,e)(R*t-E*e)/eta
% Call Solver With Anonymous Function Name,
% Independent Variable Range and Initial Value of Dependent Variable
[t,e]=ode45(ode1,[0:0.01:4],0);
% Plotting Segment
T=R*[0:0.01:4];
figure(1)
plot(e,T,'k','Linewidth',2)
xlabel('Strain, \epsilon'),ylabel('Stess, T')
title('Stress - Strain Response')
hold on;grid on
axis([0,0.45,0,60])
figure(2)
plot(t,e,'k','Linewidth',2)
xlabel('Time, t'),ylabel('Strain, \epsilon'),title('Strain-Time Response')
grid on; hold on
figure(3)
plot(t,T,'k','Linewidth',2)
grid on,hold on
xlabel('Time, t'),ylabel('Stress, T'),title('Stress-Time Response')
end
```

```
% MATLAB CODE: C-13
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Example 6.5.5 Kelvin Model Under Loading and Unloading
% Solutions & Plots Using Numerical ODE Integrator
clc;clear all;clf
% Input Kelvin Model & Loading Parameters
eta=100;E=200;
ode1=@(t,e)(60*t-E*e)/eta;
% Determine Loading Strain-Time Values
[t,e]=ode45(ode1,[0:0.001:1],0);
ee=e(end)
% Determine Loading Stress-Time Values
T=60*[0:0.001:1];
% Plot Loading Segment
plot(e,T,'k','Linewidth',2)
xlabel('Strain, \epsilon'),ylabel('Stess, T')
title('Stress - Strain Response')
hold on;grid on
axis([0,0.3,0,70])
% Determine Unloading Strain-Time Values
ode1=@(t,e)(120-60*t-E*e)/eta;
[t,e]=ode45(ode1,[1:0.001:2],ee);
% Determine Unloading Stress-Time Values
T=120-60*[1:0.001:2];
% Plot Unloading Segment
plot(e,T,'k','Linewidth',2)
```

```
% MATLAB CODE: C-14
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Plot Relaxation and Creep Functions
% For Three Parameter Viscoelastic Solid & Fluid Models
clc;clear all;clf
% Input Three Parameter Solid Model Parameters
p1=5;q0=0.5;q1=10;
t=[0:0.01:20];
% Three Parameter Solid Plot
G=(q1/p1)*exp(-t/p1)+q0*(1-exp(-t/p1));
J=(p1/q1)*exp(-q0*t/q1)+(1/q0)*(1-exp(-q0*t/q1));
figure(1)
plot(t,G,'k','linewidth',2)
grid on, hold on
plot(t,J,'--k','linewidth',2)
title('Three Parameter Solid Model')
xlabel('Time,\it t');ylabel('Relaxation and Creep Functions')
axis([0,20,0,2])
legend('Relaxation Function','Creep Function', 0)
% Three Parameter Fluid Plot
% Input Three Parameter Fluid Model Parameters
p1=5;q1=10;q2=10
G=(1/p1)*(q1-(q2/p1))*exp(-t/p1);
J=(t/q1)+(((p1*q1)-q2)/q1^2)*(1-exp(-q1*t/q2));
figure(2)
plot(t,G,'k','linewidth',2)
grid on, hold on
plot(t,J,'--k','linewidth',2)
title('Three Parameter Fluid Model')
xlabel('Time,\it t');ylabel('Relaxation and Creep Functions')
axis([0,20,0,2])
legend('Relaxation Function','Creep Function', 0)
```

```
% MATLAB CODE: C-15
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Example 6.6.1 Thick-Walled Cylinder Problem
% Plot ElastoPlastic In-Plane Stresses
clc;clear all;clf
for rp=[0.5,0.65,0.75,1];
   for k=50:100
r=k/100;
if r<rp
   r=[0.5:0.01:rp];
   Trr=-1+rp.^2-log((rp.^2)./r.^2);
   Ttt=1+rp.^2-log((rp.^2)./r.^2);
else
   r=[rp:0.01:1];
   Trr=-(rp.^2)./(r.^2)+rp.^2;
   Ttt=(rp.^2)./(r.^2)+rp.^2;
```

```
end
plot(r,Trr,'k', 'linewidth', 2)
hold on
plot(r,Ttt,'k--', 'linewidth', 2)
xlabel('Dimensionless Distance, r/r_2')
ylabel('Dimensionless Stress')
grid on;
   end
end
```

```
% MATLAB CODE: C-16
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Example 6.6.2 Plastic Torsion Example Plot
clc;clear all;clf
a=[1.0:0.1:3];
T=1-0.25*a.^(-3);
plot(a,T,'k', 'linewidth', 2)
ylabel('Dimensionless Torque , T/T_U')
xlabel('Dimensionless Angle of Twist , \alpha/\alpha_p')
grid on
```

```
% MATLAB CODE: C-17
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Example 7.2.3, Thermal Stresses in Annular Plate
clc;clear all;clf
R=3;
r=[1:0.01:R];
Tr=(1/(2*log(R)))*(-log(R./r)-(R^2-1)^(-1)*(1-(R^2./r.^2))*log(R));
Tt=(1/(2*log(R)))*(1-log(R./r)-(R^2-1)^(-1)*(1+(R^2./r.^2))*log(R));
T=log(R./r)./log(R);
figure(1)
plot(r,T,'k','linewidth',2)
grid on
xlabel('Dimensionless Radial Distance, r/r_i')
ylabel('Dimensionless Temperature, \theta/\theta_i')
figure(2)
plot(r,Tr,'k--','linewidth',2)
hold on;grid on
plot(r,Tt,'k','linewidth',2)
xlabel('Dimensionless Radial Distance, r/r_i')
ylabel('Dimensionless Stress')
```

```
% MATLAB CODE: C-18
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Poroelasticity Example 7.3.1
% Pore Pressure and Surface Displacement
clc;clear all;clf
% Pore Pressure
t=[0:0.001:0.5];
for m=1:3;
   x=m/4;
p=0;
for n=1:2:25
p=p+(4/(n*pi))*sin(n*pi*x/2).*exp(-n^2*pi^2*t);
end
figure(1)
plot(t,p,'k','linewidth',2.2)
xlabel('Dimensionless Time, \tau');
ylabel('Dimensionless Pore Pressure')
%title(['Temperature Distribution Solution , Time = ',num2str(t)])
axis([0,0.5,0,1])
grid on , hold on
end
u=0;
for n=1:2:25
u=u+(8/(n^2*pi^2))*(1-exp(-n^2*pi^2*t));
end
figure(2)
U=1*(1+1*u)
plot(t,U,'k','linewidth',2.2)
xlabel('Dimensionless Time, \tau');
ylabel('Dimensionless Surface Displacement, u(0,t)/K_1')
grid on
```

```
% MATLAB CODE: C-19
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Poroelasticity Example 7.3.2
% Pore Pressure Calculation & Plot
clc, clear all
% Input Material Properties
n=0.2;nu=0.5;B=1;
M=(1-n)/(nu-n);
x=[0:0.02:20];
f1=tan(x);f2=M*x;
figure(1)
plot(x,f1,x,f2)
grid on
axis([16 24 0 80])
% Collect roots from graph
%xr=[1.45,1.57,4.67,4.71,7.828,7.842]
xr=[1.3,4.63,7.8,10.97,14.107,17.2];
% Pore Pressure Calculation & Plot
```

```
A1=1/(B*(1-nu));
for t=[0.01,0.1,0.5,1];
x=[0:0.01:1];
p=0;
for n=1:6
   Bn=sin(xr(n))/(xr(n)-sin(xr(n))*cos(xr(n)));
   p=p+Bn*(cos(xr(n)*x)-cos(xr(n)))*exp(-xr(n)^2*t);
end
P=2*p/A1;
figure(2)
plot(x,P,'k','linewidth',2)
%axis([0 1 0 0.4])
grid on, hold on
xlabel('Dimensionless Distance, ')
ylabel('Dimensionless Pore Pressue, p/p_o')
end


% MATLAB CODE: C-20
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Example 8.3.1 Uniaxial Nonlinear Elastic Response
% Calculation & Plot Cauchy and PK1 Stresses
clc;clear all ;clf
% Piola-Kirchhoff Stress Plots
% Mooney-Rivlin Material
a1=0.5;a2=0.3;
L=1:0.01:4;
To=(L-(1./(L.^2))).*(a1-(a2./L));
figure(1)
plot(L,To,'k','linewidth',2)
xlabel('Stretch Ratio, \lambda')
ylabel('Piola-Kirchhoff Axial Stress')
hold on
% Neo-Hookean Material
To=a1*(L-(1./(L.^2)));
plot(L,To,'k--','linewidth',2)
legend(['Mooney: \alpha_1 = ',num2str(a1),' \alpha_2 = ',num2str(a2)]...
   ,['Neo-Hookean: \alpha_1 = ',num2str(a1),' \alpha_2 = 0'],2)
grid on
% Cauchy Stress Plots
% Mooney-Rivlin Material
T=(L.^2-(1./L)).*(a1-(a2./L));
figure(2)
plot(L,T,'k','linewidth',2)
xlabel('Stretch Ratio, \lambda')
ylabel('Cauchy Axial Stress')
hold on
% Neo-Hookean Material
T=a1*(L.^2-(1./L));
plot(L,T,'k--','linewidth',2)
legend(['Mooney: \alpha_1 = ',num2str(a1),' \alpha_2 = ',num2str(a2)]...
   ,['Neo-Hookean: \alpha_1 = ',num2str(a1),' \alpha_2 = 0'],2)
grid on
```

```
% MATLAB CODE: C-21
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Uniaxial Nonlinear Elastic Response
% Calculation & Plot Mooney, Neo-Hookean, Material X Stresses
clc;clear all ;clf
L=1:0.1:2.5;
% Cauchy Stress Plots
% Mooney-Rivlin Material
% Specify Mooney Parameters
a1=??;a2=??;
T=(L.^2-(1./L)).*(a1-(a2./L));
plot(L,T,'k:','linewidth',2.5)
xlabel('Stretch Ratio, \lambda')
ylabel('Cauchy Axial Stress, T_1_1')
hold on
% Neo-Hookean Material
% Specify Neo-Hookean Parameter
a1=??;
T=a1*(L.^2-(1./L));
plot(L,T,'k--','linewidth',2.5)
% Material X Linearized Data Fit
% Specify Linear Equation Fit
Lfit=??;
T=(L.^2-(1./L))*2.*Lfit;
plot(L,T,'k','linewidth',2.5)
legend(['Mooney: \alpha_1 = ',num2str(a1),' \alpha_2 = ',num2str(a2)]...
   ,['Neo-Hookean: \alpha_1 = ',num2str(a1),' \alpha_2 = 0']...
   ,['Material X (Linearized Data Fit)'],2)
grid on
```

```
% MATLAB CODE: C-22
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Example 8.3.1 Uniaxial Nonlinear Elastic Response
% Calculation & Plot Mooney-Rivlin & Rivlin & Saunders Test Data
clc;clear all; clf
% Load and Plot Data
d=load('elasticdata.txt')
figure(1)
plot(d(:,1),d(:,2),'o','markersize',7,'markerfacecolor','k')
axis([0.2,1,1.7,2.5])
xlabel('1/\lambda'); ylabel('(\alpha_1-\alpha_2/\lambda)/2')
title('Finite Elastic Tensile Data, Rivlin & Saunders (1951)')
hold on
grid on
% Re-Plot and Curve Fit Data
L=d(:,1).^-1;
figure(2)
plot(L,d(:,2),'o','markersize',7,'markerfacecolor','k')
axis([1,4,1.7,2.5])
xlabel('\lambda'); ylabel('(\alpha_1-\alpha_2/\lambda)/2')
title('Polynomial Curve Fit To Tensile Data')
hold on
grid on
```

elasticdata.txt

```
0.27 1.90
0.28 1.88
0.30 1.86
0.31 1.84
0.33 1.83
0.35 1.835
0.375 1.84
0.4 1.85
0.43 1.88
0.47 1.91
0.52 1.95
0.56 2.0
0.625 2.08
0.68 2.15
0.74 2.24
0.79 2.28
0.85 2.31
0.91 2.41
```

```
p=polyfit(L,d(:,2),2)
xp=1:0.1:4;
yp=polyval(p,xp)
plot(xp,yp,'k','linewidth',2.5)
% Plot Stress-Strain Response Using Curve Fit Constitutive Function
T=(xp.^2-(1./xp)).*2.*yp;
figure(3)
plot(xp,T,'k','linewidth',2.5)
xlabel('Stretch Ratio, \lambda')
ylabel('Cauchy Axial Stress, T_1_1')
title('Stress - Deformation Behavior for Finite Elastic Material')
grid on; hold on
% Plot Mooney-Rivlin Prediction
L=1:0.1:4;a1=4;a2=0.3
T=(L.^2-(1./L)).*(a1-(a2./L));
plot(L,T,'k--','linewidth',2.5)
```

```
% MATLAB CODE: C-23
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Calculation & Plot Gradient Elasticity Dislocation Example 9.5.1
clc;clear all;clf
for n=[1,3,10]
c=.01*n;
r=0.001:0.01:3;
ege=abs((1/(4*pi))*(-(1./r)+(1/sqrt(c)).*besselk(1,r/sqrt(c))));
ece=abs((1/(4*pi))*(-(1./r)));
plot(r,ege,'k-','linewidth',2)
hold on
plot(r,ece,'k--','linewidth',2)
axis([0,3,0,0.5]);grid on
xlabel('Distance, r'); ylabel('| \epsilon_x_z / b |')
title('Strain Field Near Screw Dislocation')
% text(0.03,0.15,'Gradient Elasticity, c=0.01')
% text(0.25,0.35,'Classical Elasticity')
legend('Gradient Elasticity','Classical Elasticity')
end
```

```
% MATLAB CODE: C-24
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Example 9.5.2 Gradient Elasticity
% Vertical Displacement - Flamant Problem
clc;clear all;clf
c=.1;nu=0.3;cs=sqrt(c);
x=-5:0.01:5;X=abs(x);
Vy=log(X*cs)-0.6;
plot(x,Vy,'k-','linewidth',2)
hold on
Vyg=(log(X*cs)+besselk(0,X))-(1-2./(X.^2)+besselk(2,X))/(2*(1-nu));
plot(x,Vyg,'k--','linewidth',2)
axis([-5,5,-3,0]);grid on
legend('Classical Elasticity','Gradient Elasticity')
xlabel('Dimensionless Distance, x/sqrt(c)');
ylabel('Dimensionless Vertical Displacement')
```

```
% MATLAB CODE: C-25
% Continuum Mechanics  - Elsevier
% M.H. Sadd, University of Rhode Island
% Example 9.7.1 Uniaxial Damage Mechanics Model Plot
clc;clear all;clf
for m=[1,1.5,2]
for n=1:100
   e(n)=n/50;
   D(n)=1-exp(-m*e(n));
   s(n)=(1-D(n))*e(n);
end
figure(1)
plot(e,s,'k-','linewidth',2)
xlabel('Strain, \epsilon')
ylabel('Nondimensional Stress, \sigma/E')
grid on; hold on
figure(2)
plot(e,D,'k-','linewidth',2)
xlabel('Strain, \epsilon')
ylabel('Damage Parameter, D')
grid on; hold on
end
```