%% MATLAB Code for Cam-Clay Model

clear all;

%% License

out1=fprintf('\n \t\t MATLAB CODE FOR SIMULATION OF MODIFIED CAMCLAY\n');

out2=fprintf('\t Copyright (C) 2011 Krishna Kumar, University of Cambridge\n');

out11=fprintf('\n\t\t\t The program is distributed under GNU GPL v 2.0 ');

out12=fprintf('\n\t\t view license agreement at http://www.gnu.org/licenses/\n');

%{

This program is free software: you can redistribute it and/or modify

it under the terms of the GNU General Public License as published by

the Free Software Foundation, either version 3 of the License, or

(at your option) any later version.

This program is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see <http://www.gnu.org/licenses/>');

%}

%% Input Parameters

out3=fprintf('\nINPUT PARAMETERS FOR MODIFIED CAMCLAY\n\n');

cp=input('Enter the inital Consolidation pressure (kPa) (eg., 150 kPa) = ');%cp=150;

p0=input('Enter the initial Confining pressure (kPa) (eg., 150 kPa) = ');%p0=150;

M=input('Enter the value of Critical Friction Anlge M (eg., 0.95) = ');%M=0.95;

l=input('Enter the value of Lamda (eg., 0.2) = ');%l=0.2;

k=input('Enter the value of Kappa (eg., 0.04) = ');%k=0.04;

N=input('Enter the value of N (eg., 2.5) = ');%N=2.5;

v=input('Enter the value of poissons ratio (eg., 0.15) = ');%v=0.15;

%% Computation of Other Parameters (V,e0 and OCR)

pc=cp;V=N-(l\*log(pc));e0=V-1;OCR=cp/p0;%Initalizing confining pressure

%% Strain Increament and Strain Matrix Definition

out4=fprintf('\nSTRAIN INCREAMENT AND ITERATION\n\n');

iteration=input('Enter number of iterations to perform (eg., 5000) = ');

if (iteration <=2000 || iteration == ' ')

iter=2000;

out5=fprintf('\nThe iterations entered is too low using defaults %f\n',iter);

else iter=iteration;

end

strsteps=input('Enter the strain increament (in decimal) (eg., 0.01) = ');

if (strsteps > 0.01|| strsteps == ' ' || strsteps <= 0.)

ide=0.01;

out6=fprintf('\nThe strain step entered is too low using defaults %f\n',ide);

else ide=strsteps;

end

de=ide/100;

es=0:ide:(iter-1)\*ide; %strain

dstrain=[de;-de/2;-de/2;0;0;0];%strain increament

%% Block Memory allocation

De=zeros(1,6);

dfds=zeros(6,1);

dfdep=zeros(6,1);

u=zeros(1,iter);

p=zeros(1,iter);

q=zeros(1,iter);

%% Yield Surface and Conditions

p1=(0:pc);% CSL in p-q space

q1 = M\*p1;

qy=(M^2\*(pc\*p1-p1.^2)).^0.5;%Plot the initial yield locus

%% Initialize

a=1;

S=[p0;p0;p0;0;0;0];

strain=[0;0;0;0;0;0];

p(a)=(S(1)+2\*S(3))/3;

q(a)=(S(1)-S(3));

yield=(q(a)^2/M^2+p(a)^2)-p(a)\*pc; %Defining the yield surface

%% CamClay Iteration Uni-Loop Iteration for OC/NC & Inside/Outside Yield

while a<iter

K=V\*p(a)/k;G=(3\*K\*(1-2\*v))/(2\*(1+v));

if yield==0, pc=(q(a)^2/M^2+p(a)^2)/p(a);

else pc=cp;

end

%Elastic Stiffness and other Matrix

for m=1:6

for n=1:6

if m<=3

if yield <0, dfds(m,1)=0;dfdep(m,1)=0;

else

dfds(m,1)=(2\*p(a)-pc)/3 + 3\*(S(m)-p(a))/M^2;

dfdep(m,1)=(-p(a))\*pc\*(1+e0)/(l-k)\*1;

end

if m==n

De(m,n)= K+4/3\*G; %Elastic Stiffness

else if n<=3, De(m,n)=K-2/3\*G;

end

end

end

if m>3, dfds(m,1)=0; dfdep(m,1)=0;%df/ds' and %df/dep

if m==n, De(m,n)= G; %Elastic Stiffness

else De(m,n)=0;

end

end

end

end

%Stiffness Matrix

if yield<0, D=De;

else D=De-((De\*dfds\*(dfds')\*De)/((-(dfdep')\*dfds+(dfds')\*De\*dfds)));

end

%Stress and Strain Updates

dS=D\*dstrain;

S=S+dS;

strain=strain+dstrain;

%Subsequent cycle update

a=a+1;

p(a)=(S(1)+S(2)+S(3))/3;

q(a)=S(1)-S(3);

u(a)=p0+q(a)/3-p(a);

if yield<0, yield=q(a)^2+M^2\*p(a)^2-M^2\*p(a)\*pc;

else yield=0;

end

end

%% Results and Plots

if OCR<=1

out7=fprintf('\n Soil is Normally Consolidated with a OCR of = %d \n',OCR);

else

out8=fprintf('\n Soil is Over Consolidated with a OCR of = %d \n',OCR);

end

disp('Choose your Plot Options, Enter number in bracket');

f=input('Plot Request: (1) Single Page; (2)Multiple Page =');

if f==1, r=2;c=2;

else if f==2, r=1;c=1;

else out9=fprintf('\nEnter either 1 or 2 \n');

f=input('Plot Request: (1) Single Page; (2)Multiple Page =');

end

end

disp('The Stress Path and other plots are being generated...');

if OCR<=1

figure1 = figure('Name','Soil is Normally Consolidated','Color',[1 1 1]);

else

figure1 = figure('Name','Soil is Over Consolidated','Color',[1 1 1]);

end

if f==2

figure1 = figure(1);

else

subplot(r,c,1,'parent',figure1)%Deviatoric Stress Vs. Axial Strain

end

plot(es,q)

xlabel('Axial strain, \epsilon\_a (%)')

ylabel('Deviatoric Stress, q (kPa)')

title('Deviatoric Stress Vs. Axial Strain')

if f==2%Deviatoric Stress Vs. Axial Strain

if OCR<=1

figure2=figure(2);

plot(p,q,p1,q1);

else

figure2=figure(2);

plot(p,q,p1,q1,p1,qy);

end

axis equal

xlabel('Mean Stress, p (kPa)')

ylabel('Deviatoric Stress, q (kPa)')

title('Stress Path')

else subplot(r,c,2,'parent',figure1),

if OCR<=1

plot(p,q,p1,q1);

else

plot(p,q,p1,q1,p1,qy);

end

axis equal

xlabel('Mean Stress, p (kPa)')

ylabel('Deviatoric Stress, q (kPa)')

title('Stress Path')

end

if f==2%Excess Pore water Pressure;

figure3=figure(3);

plot(es,u);

else

subplot(r,c,3,'parent',figure1), plot(es,u)

end

xlabel('Axial strain, \epsilon\_a (%)')

ylabel('Excess Pore Water Pressure, u (kPa)')

title('Excess Pore Water Pressure Vs. Axial Strain')

% Create an output folder

status = mkdir('Output');

cd('Output');

if f==2

saveas(figure1,'DeviatoricStress\_vs\_AxialStrain','fig')

print(figure1,'-depsc2','DeviatoricStress\_vs\_AxialStrain.eps')

print(figure1,'-dtiff','-r600','DeviatoricStress\_vs\_AxialStrain.tiff')

saveas(figure2,'StressPath','fig')

print(figure2,'-depsc2','StressPath.eps')

print(figure2,'-dtiff','-r600','StressPath.tiff')

saveas(figure3,'ExcessPWP\_vs\_AxialStrain','fig')

print(figure3,'-depsc2','ExcessPWP\_vs\_AxialStrain.eps')

print(figure3,'-dtiff','-r600','ExcessPWP\_vs\_AxialStrain.tiff')

else

saveas(figure1,'MCC','fig')

print(figure1,'-depsc2','MCC.eps')

print(figure1,'-dtiff','-r600','MCC.tiff')

end

cd ../