Project #7, Due 12/04/2019

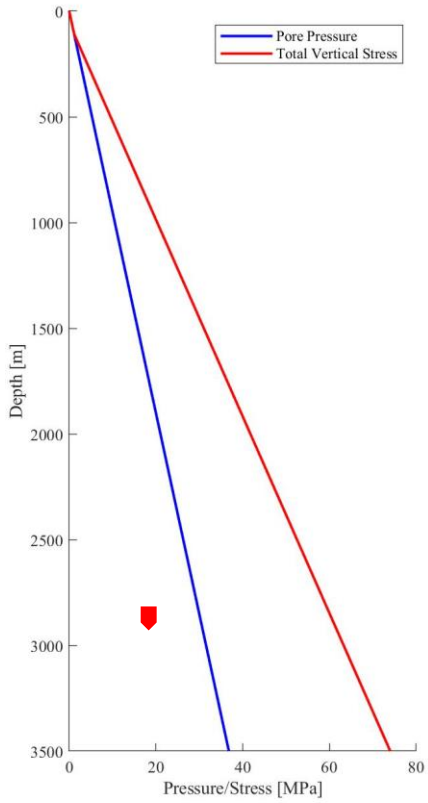
Name: Chuxi Liu UTEID: cl44262

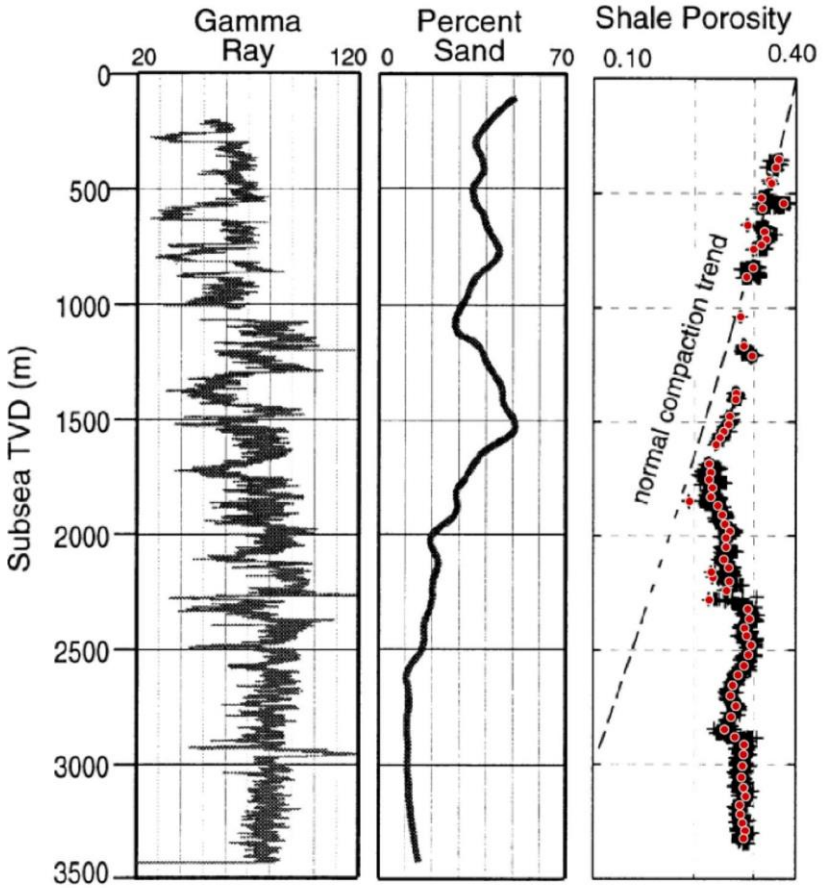
Question 1

a) This part can be easily done using Matlab. Since in the part c, SI units are used and also the log units are in SI, we convert the pore pressure gradient to SI units:

$$\frac{dP_p}{dz} = 0.465\frac{psi}{ft} \times 6894.76\frac{Pa}{psi} \times \frac{1}{0.3048}\frac{ft}{m} = 10518.6\frac{Pa}{m}$$

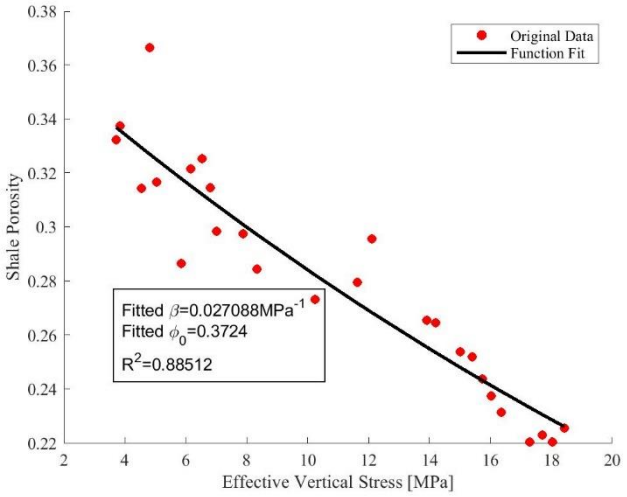
Since the pore pressure gradient is assumed to be hydrostatic, the calculation for pore pressure can start at depth 0m. The generated plot is shown below on the left, the completed digitization is shown below on the right:

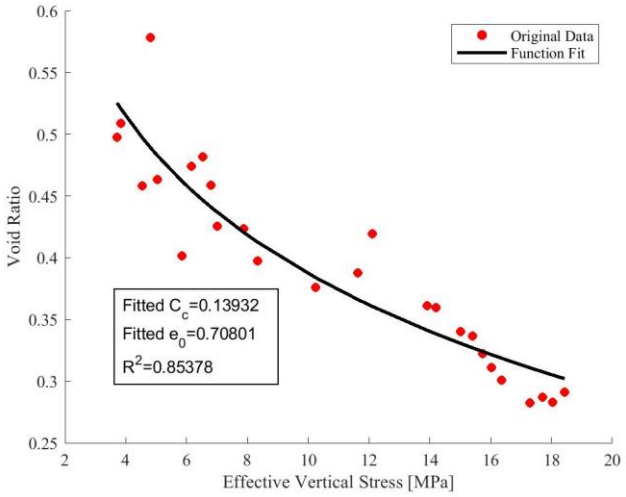




b) This part can also be achieved using Matlab. The same conversion is done for the total vertical stress gradient. I picked the sea floor depth as 110.6 meters, which is shown by the first data point on the "percent sand" log track. For depth above this value, pore pressure gradient is used since it reflects hydrostatic gradient. Below the seafloor, total vertical stress gradient is used to reflect rock weights. The resulting plot is shown above

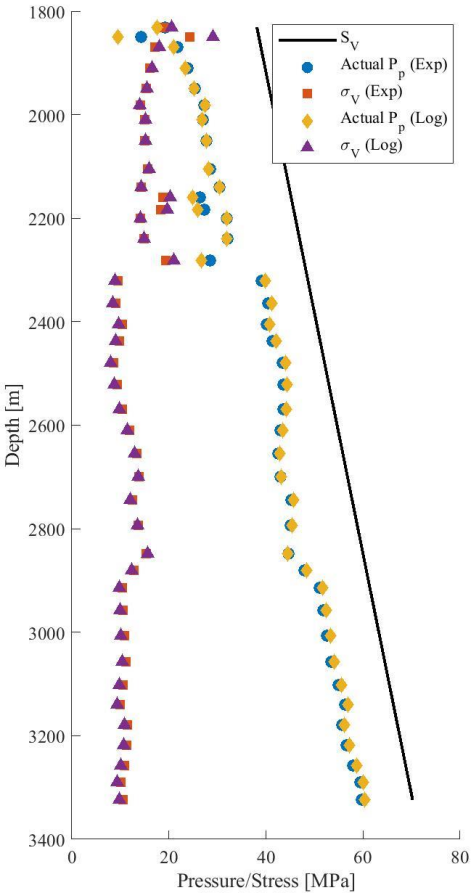
c) By using the fit function in the Matlab, the curve fitting task can be easily carried out. The plot below shows the fitted porosity and void ratio (Matlab codes are attached in Appendix).

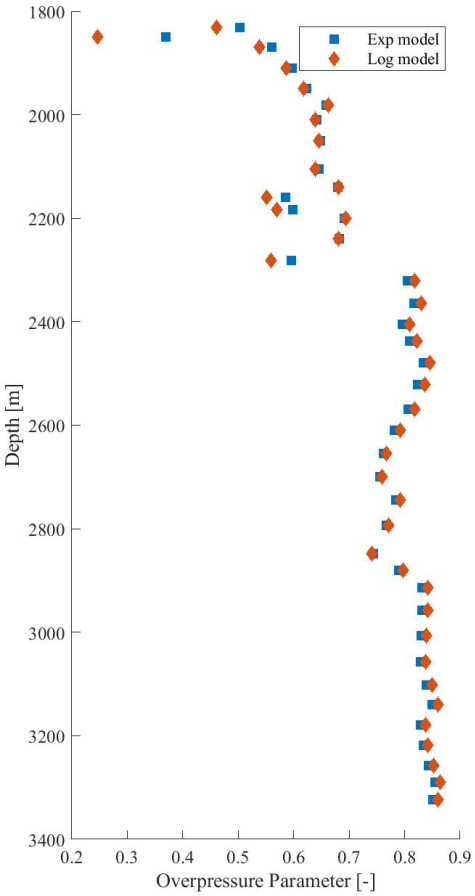




These model fits are considered to be average fittings and should be used with cautions. (It is also actually related to the digitization resolution!!!)

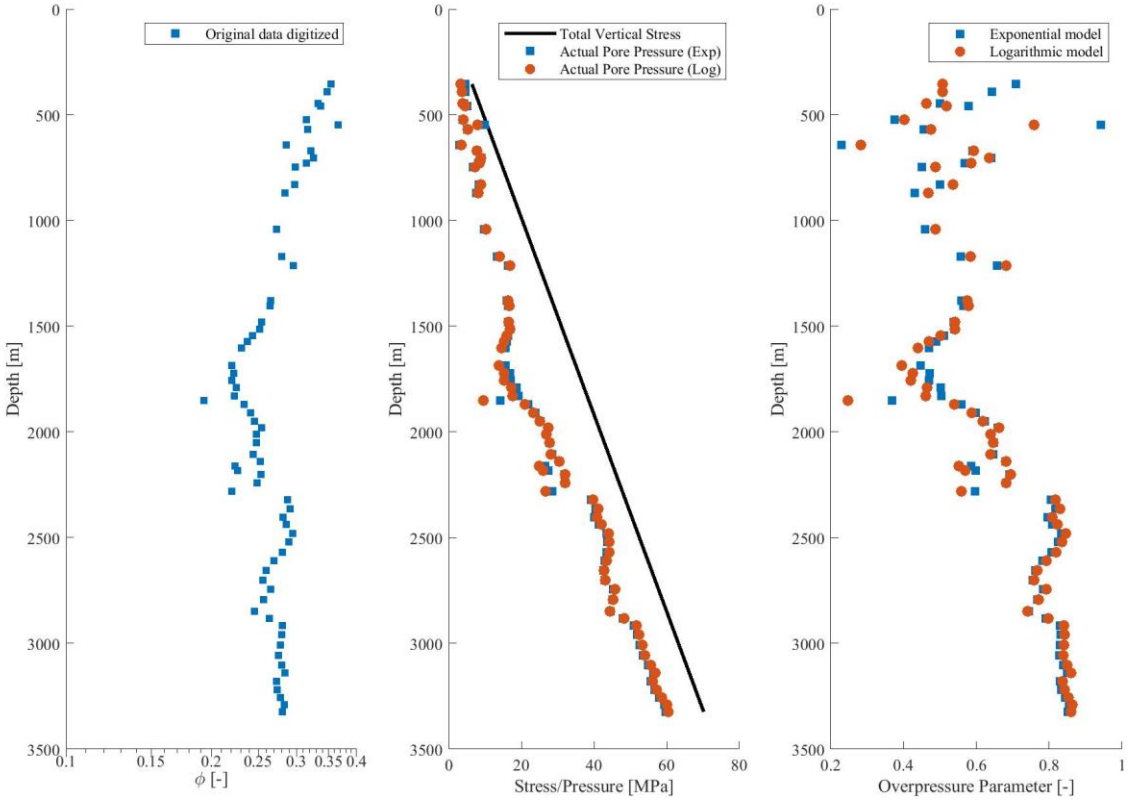
d) Apply the two models obtained in part c (both exponential and logarithm) to the depth between 1800m and 3400m, we obtain the following plot (below & left, Matlab codes in Appendix):





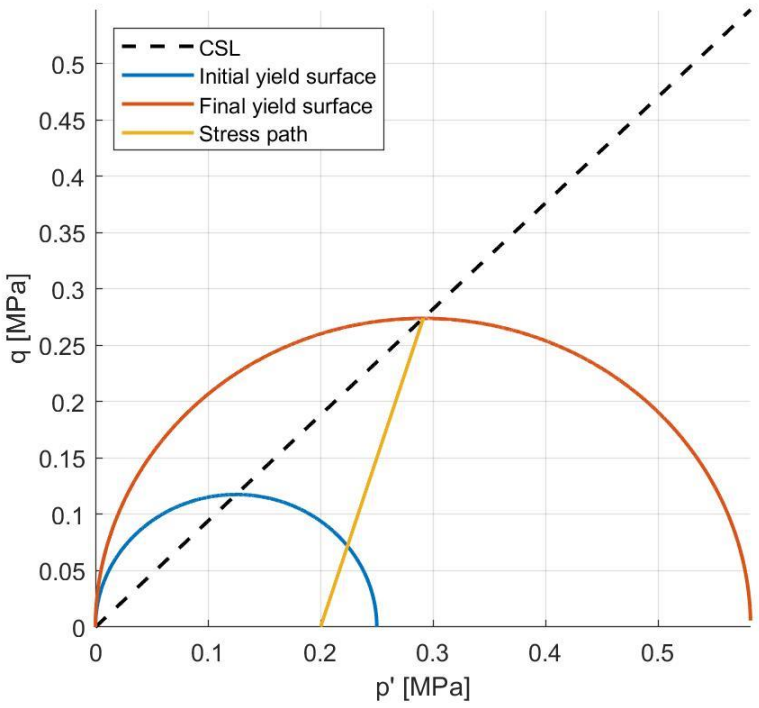
e) The overpressure parameter is simply equal to $\lambda_p = \frac{P_p}{S_V}$. The resulting plot for depth interval between 1800m and 3400m is shown above on the right.

f) The left plot is easy to obtain. If we apply the model developed in part c and apply to all the digitized depths, the resulting plots for actual pore pressure and overpressure parameter can be extrapolated to all depths. The summary plot is shown below:

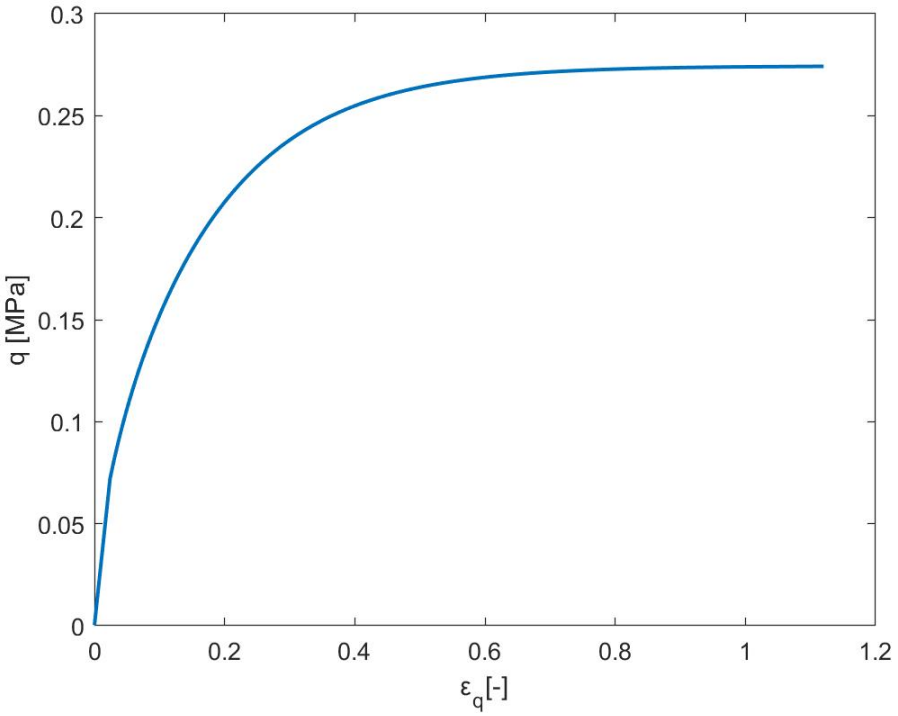


Question 2

For the Cam-clay model, the following steps are followed. First, we declare all the constants and initialize all the parameters' values given the initial conditions. Next, the change for $p', p_0', q,$ void ratio, $\varepsilon_{p'}$ and ε_q are calculated based on the provided equations in the project description/class notes. Then, the updated parameters are obtained by adding the calculated changes to the previous value. $\frac{q}{p'}$ is checked against M. If the ratio is larger than M, the material fails. Otherwise, record all the above-mentioned parameters' updated value to a data structure. These steps are repeated until the material fails. The plot below shows the stress path, yield surfaces obtained by this triaxial test simulation (detailed Matlab code attached in appendix):



It is already clear that the material is strain-hardening, because the stress path crosses over the right half of the initial yield surface, as can be observed above. However, to better visualize whether the material is strain-hardening or strain-softening, the plot of q vs ε_q is made, as shown below:

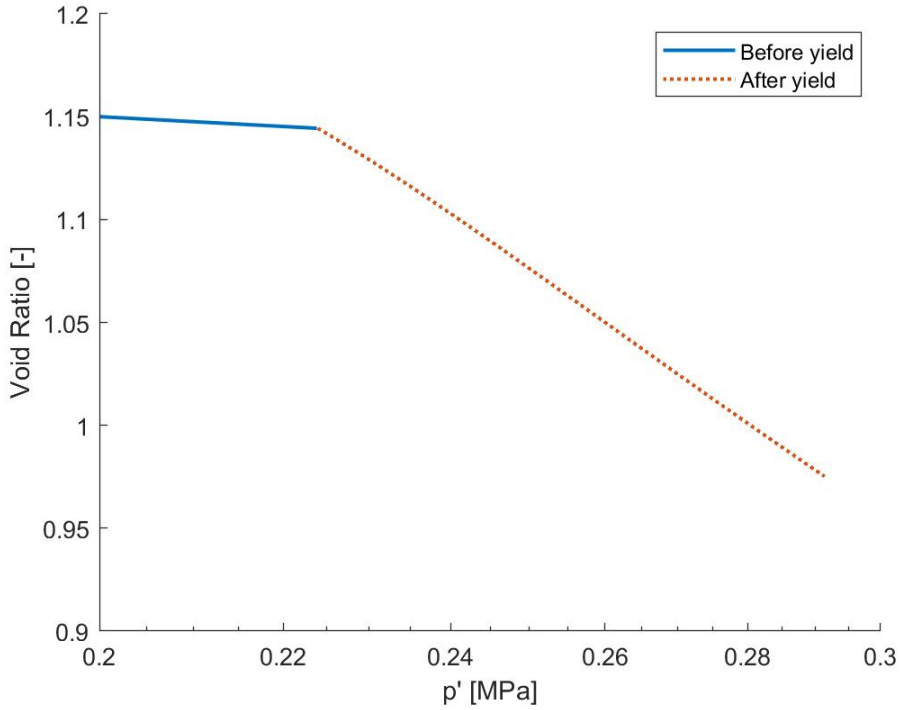


As can be seen from the plot above, the deviatoric stress always increases and approaches asymptotically toward a maximum value. This suggests that the material is strain-hardening indeed. It means that the material is getting stronger and stronger, but this trend approaches a limit as the stress path is near the critical state line (CSL).

According to the third criteria for yield, we have:

$$\begin{cases} q = Mp' \\ p' = \dfrac{p_0'}{2} \end{cases}$$

This implies that if q is larger than $\frac{M \cdot p_0'}{2}$, the material will yield. By selecting the portion of before-yield data and after-yield data and plotting, we obtain the following plot:



Due to the yield, the material will behave differently. Before the yield, the deformation is elastic and thus the slope is equal to $-\kappa$. After the yield, the deformation becomes plastic and the slope is equal to $-\lambda$. Indeed, $\lambda > \kappa$, and this plot above is matching the schematics drawn in the class note.

Appendix A, Question 1 Code

```
clear all; close all; clc


%% Q1 part a+b
conver = 22620.6; % conversion of psi/ft to pa/m
Pp_grad = conver*0.465; % in units of Pa/m
Sv_grad = conver*0.95; % in units of Pa/m
Depth_Pp = 0:1:3500;
sea_flr = 110.6; % this is done by observing the first data of percent sand
S_v = zeros(1,3500);
for k = 1:length(S_v)+1
    if k<sea_flr
        S_v(k) = Depth_Pp(k)*Pp_grad;
    else
```

```
            S_v(k) = S_v(k-1) + Sv_grad;
        end
end
figure; hold on
plot(Pp_grad*Depth_Pp/10^6,Depth_Pp,'b','LineWidth',1.5)
plot(S_v/10^6,Depth_Pp,'r','LineWidth',1.5)
sig_v = (S_v - Pp_grad*Depth_Pp)/10^6; % in MPa
axis ij; xlabel('Pressure/Stress [MPa]'); ylabel('Depth [m]');
x0=100; y0=200; width=350; height=700;
legend('Pore Pressure','Total Vertical Stress')
set(gca, 'FontName', 'Times New Roman')
set(gcf,'position',[x0,y0,width,height])
%% Q1 part c
shale_poro = xlsread('Q1_Shale_poro.csv');
for row = 1:size(shale_poro,1)
        if shale_poro(row,2) <= 400
            start_row = row;
        end
        if shale_poro(row,2) <= 1800
            end_row = row;
        end
end
partc_data = shale_poro(start_row+1:end_row,:);
partc_dep = round(partc_data(:,2));
% shale porosity
y = partc_data(:,1);
x = sig_v(partc_dep)';
fitfun = fittype( @(phi_init,beta,x) phi_init*exp(-beta*(x)) );
[fitted_curve,gof] = fit(x,y,fitfun); %
coefs = coeffvalues(fitted_curve);
phi_0 = coefs(1); beta = coefs(2);
figure; hold on
scatter(x, y, 30, 'ro','filled')
plot(x,fitted_curve(x),'k','LineWidth',2)
xlabel('Effective Vertical Stress [MPa]'); ylabel('Shale Porosity');
set(gca, 'FontName', 'Times New Roman')
legend('Original Data','Function Fit')
dim = [0.2 0.5 0.3 0.3];
str = {strcat('Fitted \beta=',num2str(beta),'MPa^{-1}'),...
            strcat('Fitted \phi_{0}=',num2str(phi_0)),...
            strcat('R^2=',num2str(gof.rsquare))};
annotation('textbox',dim,'String',str,'FitBoxToText','on','Position',[0.2 0.1 0.3 0.3]);
% void ratio
y2 = partc_data(:,1)./(1-partc_data(:,1));
x2 = sig_v(partc_dep)';
fitfun2 = fittype( @(e_coef,C_c,x) e_coef-C_c*log(x));
```

```
[fitted_curve2,gof2] = fit(x2,y2,fitfun2);
coefs = coeffvalues(fitted_curve2);
e_0 = coefs(1); C_c = coefs(2);
figure; hold on
scatter(x2, y2, 30, 'ro','filled')
plot(x2,fitted_curve2(x2),'k','LineWidth',2)
xlabel('Effective Vertical Stress [MPa]'); ylabel('Void Ratio');
set(gca, 'FontName', 'Times New Roman')
legend('Original Data','Function Fit')
dim = [0.2 0.5 0.3 0.3];
str = {strcat('Fitted C_c=',num2str(C_c)),...
        strcat('Fitted e_{0}=',num2str(e_0)),...
        strcat('R^2=',num2str(gof2.rsquare))};
annotation('textbox',dim,'String',str,'FitBoxToText','on','Position',[0.2 0.1 0.3 0.3]);
%% Q1 part d
for row = 1:size(shale_poro,1)
    if shale_poro(row,2) <= 1800
        start_row = row;
    end
    if shale_poro(row,2) <= 3400
        end_row = row;
    end
end
partd_data = shale_poro(start_row+1:end_row,:);
partd_dep = round(partd_data(:,2));
partd_poro = partd_data(:,1);
for row = 1:length(partd_poro)
    Pp_act(row) = S_v(partd_dep(row))/1e6+log(partd_poro(row)/phi_0)/beta;
    sig_v_log(row) = exp((e_0-partd_poro(row)/(1-partd_poro(row)))/C_c);
end
% part d+e
figure; hold on;
plot(S_v(partd_dep)/1e6,partd_dep,'k-','LineWidth',1.5)
scatter(Pp_act,partd_dep,40,'o','filled');
scatter(S_v(partd_dep)/1e6-Pp_act,partd_dep,40,'s','filled');
scatter(S_v(partd_dep)/1e6-sig_v_log,partd_dep,40,'d','filled');
scatter(sig_v_log,partd_dep,40,'^','filled')
axis ij; xlabel('Pressure/Stress [MPa]'); ylabel('Depth [m]');
x0=100; y0=200; width=350; height=700;
legend('S_{V}','Actual P_{p} (Exp)','\sigma_{V} (Exp)',...
    'Actual P_{p} (Log)','\sigma_{V} (Log)','Location','best')
set(gca, 'FontName', 'Times New Roman')
set(gcf,'position',[x0,y0,width,height])
figure; hold on;
scatter(Pp_act./(S_v(partd_dep)/1e6),partd_dep,40,'s','filled')
scatter((S_v(partd_dep)/1e6-sig_v_log)./(S_v(partd_dep)/1e6),partd_dep,...
```

```
    40,'d','filled')
axis ij; xlabel('Overpressure Parameter [-]'); ylabel('Depth [m]');
x0=100; y0=200; width=350; height=700;
legend('Exp model','Log model')
set(gca, 'FontName', 'Times New Roman')
set(gcf,'position',[x0,y0,width,height])
%% Q1 part f
S_v_all = S_v(round(shale_poro(:,2)));
for row = 1:length(S_v_all)
    Pp_act_all(row) = S_v_all(row)/1e6+log(shale_poro(row,1)/phi_0)/beta;
    sig_v_log_all(row) = exp((e_0-shale_poro(row,1)/(1-shale_poro(row,1)))/C_c);
end
figure
x0=100; y0=200; width=350*3; height=700;
set(gcf,'position',[x0,y0,width,height])
subplot(1,3,1)
hold on;
scatter(shale_poro(:,1),shale_poro(:,2),30,'s','filled')
axis ij; xlabel('\phi [-]'); ylabel('Depth [m]'); xlim([0.1 0.4])
a=gca; legend('Original data digitized')
a.XRuler.TickLabelGapOffset = -3;
set(gca, 'FontName', 'Times New Roman'); set(gca,'xscale','log');
hold off;
subplot(1,3,2)
hold on;
plot(S_v_all/1e6,shale_poro(:,2),'k-','LineWidth',2)
scatter(Pp_act_all,shale_poro(:,2),40,'s','filled')
scatter(S_v_all/1e6-sig_v_log_all,shale_poro(:,2),40,'o','filled')
axis ij; xlabel('Stress/Pressure [MPa]'); ylabel('Depth [m]');
set(gca, 'FontName', 'Times New Roman')
legend('Total Vertical Stress','Actual Pore Pressure (Exp)',...
    'Actual Pore Pressure (Log)')
hold off;
subplot(1,3,3)
hold on;
scatter(Pp_act_all./(S_v_all/1e6),shale_poro(:,2),40,'s','filled')
scatter((S_v_all-sig_v_log_all*1e6)./(S_v_all),shale_poro(:,2),40,'o','filled')
axis ij; xlabel('Overpressure Parameter [-]'); ylabel('Depth [m]');
legend('Exponential model','Logarithmic model')
set(gca, 'FontName', 'Times New Roman')
hold off;
```

Appendix B, Question 2 Code

```
clear all; close all; clc


%% Get all the simulation data
```

```
% matrix to store all experiment data
pprm_store = zeros(2,1); q_store = pprm_store; e_store = q_store;
pprm_0_store = e_store; epi_pprm_store = pprm_0_store;
epi_q_store = epi_pprm_store;
% constants initialization
G = 1; % MPa
pprm_0_init = 0.25; %MPa
phi_cs = 24; % degrees
lambda = 0.25; % Loading compressibility
kappa = 0.05; % Unloading compressibility
e_0 = 1.15; % initial void ratio
pprm_init = 0.2; %MPa
q_init = 0;
pprm_step = 0.0001; % simulation step size for x-axis
q_step = 3*pprm_step; % simulation step size for y-axis
phi_ratio = (1+sind(phi_cs))/((1-sind(phi_cs)));
M = 3*(phi_ratio-1)/(2+phi_ratio);
failure = 0; % 0 indicates no failure, 1 indicates failure
e = e_0; pprm_0 = pprm_0_init; q = q_init; pprm = pprm_init;
epi_pprm = 0; epi_q = 0; idx = 1;
while failure < 1
    %calculate all changes
    eta = q/pprm;
    dChi = (lambda-kappa)/((1+e)*pprm*(M^2+eta^2)); % Hardening Parameter
    if (q^2 - M^2*pprm*(pprm_0-pprm))>=0 % checking yield limit
        depi_pprm_Pla = dChi*(M^2-eta^2)*pprm_step+dChi*2*eta*q_step;
        depi_q_Pla = dChi*2*eta*pprm_step+dChi*4*eta^2/(M^2-eta^2)*q_step;
        de = -(1+e)*depi_pprm_Pla;
        dpprm_0 = depi_pprm_Pla*(1+e)*pprm_0/(lambda-kappa);
    else
        depi_pprm_Pla=0; depi_q_Pla=0; dpprm_0=0;
        de = -kappa/pprm*pprm_step;
    end
    depi_pprm_Ela = kappa/(1+e)*pprm_step/pprm;
    depi_q_Ela = 1/(3*G)*q_step;

    %update all parameters
    e = e+de; q = q+q_step; pprm = pprm+pprm_step; pprm_0 = pprm_0+dpprm_0;
    epi_pprm = epi_pprm + depi_pprm_Pla + depi_pprm_Ela;
    epi_q = epi_q + depi_q_Pla + depi_q_Ela;
    %check failure (q>=p'*M or q/p'>=M)
    if q/pprm >= M
        failure = failure+1;
    else
        pprm_store(idx) = pprm; q_store(idx) = q; e_store(idx) = e;
        pprm_0_store(idx) = pprm_0; epi_pprm_store(idx) = epi_pprm;
```

```
            epi_q_store(idx) = epi_q;
            idx = idx+1;
        end
end
%% Part a
figure; hold on; axis equal; grid on;
x_plot = 0:pprm_step:pprm_0_store(end);
init_yield_x = 0:pprm_step:pprm_0_init;
init_yield_y = sqrt(M^2*init_yield_x.*(pprm_0_init-init_yield_x));
final_yield_x = 0:pprm_step:pprm_0_store(end);
final_yield_y = sqrt(M^2*final_yield_x.*(pprm_0_store(end)-final_yield_x));
plot(x_plot,M*x_plot,'k--','LineWidth',1.5) % CSL
plot(init_yield_x,init_yield_y,'LineWidth',1.5) % Initial yield surface
plot(final_yield_x,final_yield_y,'LineWidth',1.5) % Final yield surface
plot(pprm_store,q_store,'LineWidth',1.5) % p', q stress path
legend('CSL','Initial yield surface','Final yield surface',...
    'Stress path','Location','northwest')
xlabel('p'' [MPa]'); ylabel('q [MPa]'); xlim([0 inf]);ylim([0 inf])
%% Part b
figure; plot(epi_q_store,q_store,'LineWidth',1.5);
xlabel([char(949) '_{q}' '[-]']); ylabel('q [MPa]');
%% Part c
for id = 1:length(pprm_store)
    if (q_store(id)^2 - M^2*pprm_store(id)*(pprm_0_init-pprm_store(id)))<0 % q_store(id)<M*pprm_0_init/2
        pprm_beforey(id) = pprm_store(id);
        e_beforey(id) = e_store(id);
    else
        break;
    end
end
e_aftery = setdiff(e_store,e_beforey);
e_aftery = sort(e_aftery,'descend');
pprm_aftery = setdiff(pprm_store,pprm_beforey);
figure; hold on;
plot(pprm_beforey,e_beforey,'-','LineWidth',1.5);
plot(pprm_aftery,e_aftery,':','LineWidth',1.5);
xlabel('p'' [MPa]'); ylabel('Void Ratio [-]'); set(gca, 'XScale', 'log')
legend('Before yield','After yield'); xlim([0.2 0.3]); ylim([0.9 1.2])
```