```
disp(datetime)
```

```
28-Jul-2020 15:42:41
```

Buckling Venus Lithosphere

This script documents the models and results reported in the manuscript entitled "Felsic Tesserae on Venus Permitted by Lithospheric Deformation Models," by Resor et al., submitted to GRL. The script accesses mineral creep data and rock compositions in the Excel file *Creep_database.xlsx.* These data are used to generate lithosphereic strength envelopes (AKA Christmas tree diagrams) and viscous buckling plots for single mineral lithospheres and polymineralic lithospheres.

Monomineralic lithosphere

Read in creep parameters

Estimates of dry and wet creep parameters are avaiable for olivine (Hirth and Kolstead, 2003), diopside (Dimanov and Dressen, 2005), and anorthite (Rybacki et al., 2006). The dry quartz flow law is modified from the we flow law of Hirth et al., 2001 using the pre-expoential term from Barbery, 2017.

```
% read mineral creep parameters from file
% columns: dry olivine, dry diopside, dry anorthite, quartz
% rows: A* n Q V (m^3/mol) rho
params = xlsread('./Resoretal_minrx_param.xlsx',...
    'minerals','B2:F5');

% seperate into individal parameters
Astar = params(:,1);
n = params(:,2);
Q = params(:,3);
V = params(:,4);
rho = params(:,5);
name = {'olivine', 'diopside', 'anorthite', 'quartz'};

% assumed pressure for laws with activation volume effect
P = 300e6; % Pa

% adjust Q for activation volume effect, if included
Q = Q + P*V;

% adjust Astar values for triaxial deformation (see Ranalli, 95, p. 76
% or Burov 2015, eq. 18)
Astar = Astar .* 3.^((n+1)./2) ./2;
```

Define additional relevant constants

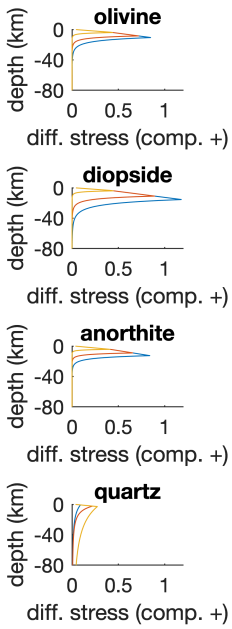
```
% define constants for Venus
g=8.87; % m/s^2
Ts=740; % deg K

% define tectonic regime for strength envelope calculation
phi='r'; % reverse-faulting
```

```
% define dominant fold wavelength and uncertainty bounds
lam = [14.4 10.1 18.7];
```

Plot Strength Envelopes

Strength envelopes for end-member parameters are plotted to illustrate the effects of strain rate and geotherm.

```
% create christmas tree plot for representative values
% set up ranges of strain rate and thermal gradient to iterate over
% values based on B&G (1-2) and this study (3-6)
logexx1 = [-17, -15, -15,  -15,  -15,  -15];
thKkm1 = [10, 20, 48, 66, 46, 5];

fig2 = figure(2);
fig2.Units = 'centimeters';
fig2.Position = [0, 0, 9.5, 23];
fig2.PaperUnits = 'centimeters';
fig2.PaperPosition = [0, 0, 9.5, 23];

% Create empty array to store numerical results
MinResults = zeros(length(Astar)*3, 5);

% loop over range of materials
for j = 1: length(Astar)
    subplot(7, 3, (3*j-2))
    hold on
    th = [thKkm1(1:2) thKkm1(j+2)]./1000;
    Exx = 10.^[logexx1(1:2) logexx1(j+2)];
    for k = 1:length(th)
        [H, gam, eta, z, env] = bdt_plot(th(k), Ts, g, rho(j), Astar(j), n(j), Q(j), ab
        plot(-env/1000, -z/1000)
        dx = (z(2)-z(1));
        IntStrength = sum(dx.*(env(1:end-1) + env(2:end))./2) * 1e6; %N/m
        MinResults((j-1)*length(th)+k,:) = [log10(Exx(k)), th(k)*1000, H/1000, env(z==H
    end
    hold off
    xlabel('diff. stress (comp. +)')
    ylabel('depth (km)')
%     legend([num2str(log10(Exx(1))),', ',num2str(th(1)*1000)],...
%         [num2str(log10(Exx(2))),', ',num2str(th(2)*1000)],...
%         [num2str(log10(Exx(3))),', ',num2str(th(3)*1000)],...
%         'Location', 'southeast')
    title(name(j));
    axis([0 1.2 -80 0])
    yticks([-80, -40, 0])
    xticks([0, 0.5, 1])
end
```




```
% pos = fig1.Position;
% axesj.Units = 'centimeters';
% axesj.OuterPosition = [0, -(j-1)*2.875, 3, 2.75];
```

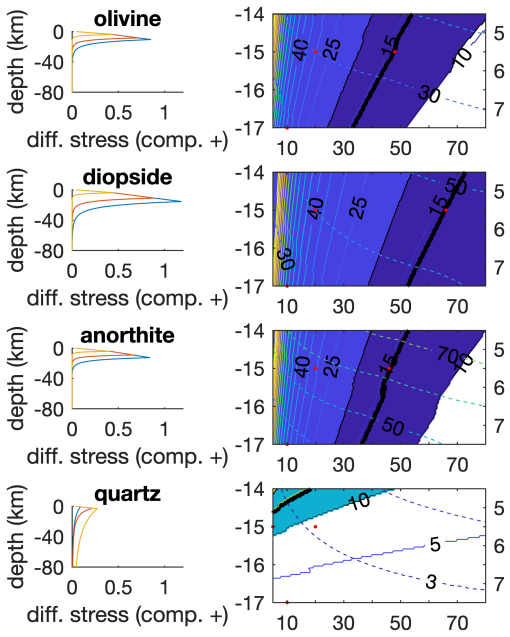
Figure 1. Strength envelopes for monomineralic lithospheres. Plots are for monomineralic a) olivine, b) diopside, c)anorthite, and d) quartz lithospheres.

Plot buckling instability

```
% plot buckling instabilities
% set up ranges of strain rate and thermal gradient to iterate over
% values used in 2010 LPSC poster
logexx = -17:0.05:-14;
```

```
thKkm = 5:0.5:80;

% time bounds 1% strain
logexxtime = log10(0.01/(365.24219*86400) * 1./(10.^[-14 -17]));

% create variable to store numerical results
MinLvalsBG = zeros(length(Astar), 2);
MinLvals = zeros(length(Astar)*4, 3);

for j = 1: length(Astar)
    % create empty matrices for solutions
    qvals=zeros(length(logexx),length(thKkm));
    Lvals=zeros(length(logexx),length(thKkm));
    Hvals=zeros(size(Lvals));

    % iterate over range of strain rates
    for u=1:length(logexx)
        Exx=-10^logexx(u);
        % iterate over range of thermal gradients
        for v=1:length(thKkm)
            th=thKkm(v)/1000; % convert to deg/m
            % Calculate buckling instability
            [qvals(u,v), Lvals(u,v), Hvals(u,v)] = perturb2(th, Ts, g, rho(j), Astar(j)
        end
    end

    subplot(7, 3, [3*j-1 3*j])
    [TH, E]=meshgrid(thKkm, logexx);

    % add mean and 1-sigma range as line and colored swath
    [~,~]=contourf(TH, E, Lvals/1000, lam(2:3));
    hold on
    [~,~]=contour(TH, E, Lvals/1000, [lam(1) lam(1)], 'k', 'LineWidth',2);

    % plot color contours for all lvals
    [c,h]=contour(TH, E, Lvals/1000, (0:5:100));
    clabel(c,h, [5, 10, 15, 25, 40])

    % add contours of qvals
    [c,h]=contour(TH, E, imgaussfilt(qvals,3), [ 3 4 10:10:100], '--');
    clabel(c,h, [3 10:20:90])
%     xlabel('thermal gradient (K/km)')
%     ylabel('log strain rate (s^-^1)')
%     title(name(j))

    % add points for envelope values
    plot([thKkm1(1:2) thKkm1(j+2)], [logexx1(1:2) logexx1(j+2)], 'r.')

    % set x axis labels to every 10 K/km
    ticks = 10:10:80;
    labels = cell(1,length(ticks));
    labels(1:2:length(ticks)) = num2cell(ticks(1:2:length(ticks)));
    xticks(ticks)
    xticklabels(labels)
```

```
    ax = gca;
    pos = ax.Position;
    ax.Position = [pos(1)+0.2*pos(3), pos(2), 0.8*pos(3), pos(4)];

    % add second y axis with strain times on log scale
    ax2 = axes('Position', ax.Position,...
    'YLim', logexxtime,...
    'Ydir', 'reverse',...
    'YTick', [5 6 7],...
    'YTicklabel', [5 6 7],...
    'XTick',[],...
    'YAxisLocation','right',...
    'XaxisLocation','top',...
    'Color','none');

    % store results for present-day and B&G Tessera
    MinLvalsBG(j,:) = [Lvals(E==-17&TH==10)/1000 Lvals(E==-15&TH==20)/1000];

    % store dominant wavelength and upper/lower bounds values for specific strain rates
    F = griddedInterpolant(TH', E', Lvals'/1000);
    Es = min(logexx):1:max(logexx);
    options = optimset('Display','off'); % hide output
    for m = 1:length(Es)
        fun = @(th)F(th,Es(m))-lam(1);
        med = fzero(fun,40,options);
        fun = @(th)F(th,Es(m))-lam(3);
        iqr1 = fzero(fun,40,options);
        fun = @(th)F(th,Es(m))-lam(2);
        iqr2 = fzero(fun,40,options);
        MinLvals(m+(j-1)*length(Es),:) = [med iqr1 iqr2];
    end
end
```




```
% save as vector pdf
% fig2.Renderer='Painters';
% saveas(fig2,'fig2','pdf')
```

Figure 2. Contour plots of dominant wavelength (solid contours) and growth factor (dashed contours) for single layer buckling instability models. Plots are for monomineralic a) olivine, b) diopside, c) anorthite, and d) quartz lithospheres.

Polymineralic lithosphere

Dry whole rock data are only available for Basaltic compositions (Columbia and Maryland diabase, Mackwell et al., 1998). We simulate whole rock deformatin of polymineralic rocks using the Minimized Power Geometric model of Huet et al., 2014).

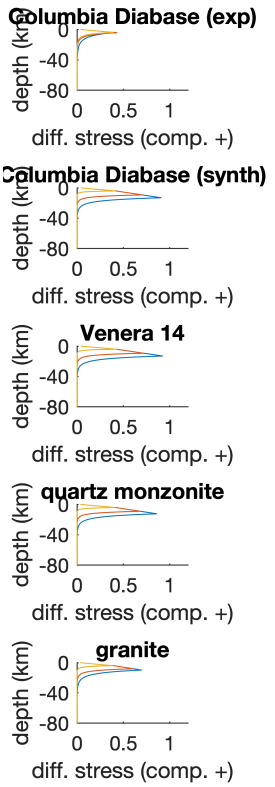
We model a suite of mafic to felsic compositions from the Laramie Anorthosite Complex (Anderson, 1995; Anderson et al., 2003). Dry mineral data are limited. We group minerals based on observation of their relative strengths when data are not available.

```
% create empty matrix for rock mineralogy and then load in values from
% excel spreadsheet

% define number of synthetic rocks to be analyzed
NumRocks = 4;

% create empty vector for modal mineral values
rocks = zeros(14,NumRocks);
name2 = {'Columbia Diabase (exp)', 'Columbia Diabase (synth)', 'Venera 14',...
    'quartz monzonite', 'granite'};
rocks(:,1) = xlsread('./Resoretal_minrx_param.xlsx',...
    'compositions','AU3:AU16'); % Columbia diabase, Mackwell et al., 1998
rocks(:,2) = xlsread('./Resoretal_minrx_param.xlsx',...
    'compositions','AW3:AW16'); % Venera 14 CIPW,
rocks(:,3) = xlsread('./Resoretal_minrx_param.xlsx',...
    'compositions','AO3:AO16'); % Quartz Monzonite MPK81, Anderson et al., 1995, 2003
rocks(:,4) = xlsread('./Resoretal_minrx_param.xlsx',...
    'compositions','AS3:AS16'); % Granite MPK78, Anderson et al., 1995, 2003

% combine similar minerals
simple = zeros(size(params,1),size(rocks,2));
simple(3,:) = rocks(1,:) + rocks(2,:); % combine all feldspars
simple(1,:) = rocks(3,:); % olivine
simple(2,:) = rocks(4,:) + rocks(5,:) + rocks(6,:)+ rocks(9,:); % pyroxene and hornbler
simple(4,:) = rocks(7,:); % quartz

% renormalize
simple = simple./repmat(sum(simple,1),4,1);

% calculate apparent (model) density
rhobar = zeros(size(simple,2)+1,1); % add one for expreimental sample
rhobar(2:end) = sum(simple.*repmat(rho,1,NumRocks),1);

% Calculate whole rock creep parameters from mineral data using MPGe mixing

% set up empty vectors for creep paramaters
nbar = zeros(size(simple,2)+1,1); % add one for expreimental sample
Qbar = nbar;
Abar = nbar;

% loop over rocks and calculate average values
for j = 2: size(simple,2)+1 % add one for expreimental sample
    [nbar(j), Qbar(j), Abar(j)] = MPGe(simple(:,j-1), n, Q, Astar);
end
```

```
% add in experimental values for Columbia diabase
Columbia = xlsread('./Resoretal_minrx_param.xlsx',...
    'rocks','B2:D2');
nbar(1) = Columbia(2);
Qbar(1) = Columbia(3);
Abar(1) = Columbia(1);
Abar(1) = Abar(1) .* 3.^((nbar(1)+1)./2) ./2;
rhobar(1) = 3000;
```

We use the modeled creep parameters to construct lithospheric strength profiles

```
% create christmas tree plot for representative values
% set up ranges of strain rate and thermal gradient to iterate over
% values based on B&G (1-2) and this study (3-7)
logexx2 = [-17, -15, -15, -15, -15, -15, -15];
thKkm2 = [10, 20, 24, 51, 51, 49, 46];

fig3 = figure(3);
fig3.Units = 'centimeters';
fig3.Position = [0, 0, 9.5, 23];
fig3.PaperUnits = 'centimeters';
fig3.PaperPosition = [0, 0, 9.5, 23];

% Create empty array to store numerical results
RockResults = zeros(length(Abar)*3, 5);

% loop over range of materials
for j = 1: length(Abar)
    subplot(7, 3, (3*j-2))
    hold on
    th = [thKkm2(1:2) thKkm2(j+2)]./1000;
    Exx = 10.^[logexx2(1:2) logexx2(j+2)];
    for k = 1:length(th)
        [H, gam, eta, z, env] = bdt_plot(th(k), Ts, g, rhobar(j), Abar(j), nbar(j), Qba
        plot(-env/1000, -z/1000)
        dx = (z(2)-z(1));
        IntStrength = sum(dx.*(env(1:end-1) + env(2:end))./2) * 1e6; %N/m
        RockResults((j-1)*length(th)+k,:) = [log10(Exx(k)), th(k)*1000, H, env(z==H), I
    end
    hold off
    xlabel('diff. stress (comp. +)')
    ylabel('depth (km)')
%     legend([num2str(log10(Exx(1))),', ',num2str(th(1)*1000)],...
%         [num2str(log10(Exx(2))),', ',num2str(th(2)*1000)],...
%         [num2str(log10(Exx(3))),', ',num2str(th(3)*1000)],...
%         'Location', 'southeast')
    title(name2(j));
    axis([0 1.2 -80 0])
    yticks([-80, -40, 0])
    xticks([0, 0.5, 1])
end
```



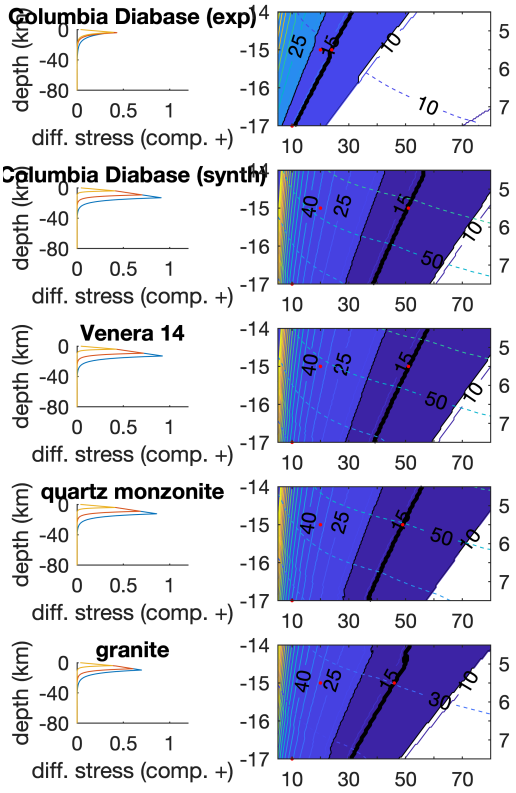

```
% pos = fig2.Position;
% fig2.Position = [pos(1:3), 400];
```

And calculate buckling instability

```
% plot buckling instabilities
% set up ranges of strain rate and thermal gradient to iterate over
% values used in 2010 LPSC poster
logexx = -17:0.05:-14;
thKkm = 5:0.5:80;

% create variable to store numerical results
RockLvalsBG = zeros(length(Astar), 2);
RockLvals = zeros(length(Astar)*4, 3);
```

```
for j = 1: length(Abar)
    % create empty matrices for solutions
    qvals=zeros(length(logexx),length(thKkm));
    Lvals=zeros(length(logexx),length(thKkm));
    Hvals=zeros(size(Lvals));

    % iterate over range of strain rates
    for u=1:length(logexx)
        Exx=-10^logexx(u);
        % iterate over range of thermal gradients
        for v=1:length(thKkm)
            th=thKkm(v)/1000; % convert to deg/m
            % Calculate buckling instability
            [qvals(u,v), Lvals(u,v), Hvals(u,v)] = perturb2(th, Ts, g, rhobar(j), Abar
        end
    end

    subplot(7, 3, [3*j-1 3*j])
    [TH, E]=meshgrid(thKkm, logexx);

    % add mean and 1-sigma range as line and colored swath
    [~,~]=contourf(TH, E, Lvals/1000, lam(2:3));
    hold on
    [~,~]=contour(TH, E, Lvals/1000, [lam(1) lam(1)], 'k', 'LineWidth',2);

    % plot color contours for all lvals
    [c,h]=contour(TH, E, Lvals/1000, (0:5:100));
    clabel(c,h, [5, 10, 15, 25, 40])

    % add contours of qvals
    [c,h]=contour(TH, E, imgaussfilt(qvals,3), (0:10:100), '--');
    clabel(c,h, 10:20:90)
%     xlabel('thermal gradient (K/km)')
%     ylabel('log strain rate (s^-^1)')
%     title(name2(j))

    % add points for envelope values
    plot([thKkm2(1:2) thKkm2(j+2)], [logexx2(1:2) logexx2(j+2)], 'r.')

    % set x axis labels to every 10 K/km
    ticks = 10:10:100;
    labels = cell(1,length(ticks));
    labels(1:2:length(ticks)) = num2cell(ticks(1:2:length(ticks)));
    xticks(ticks)
    xticklabels(labels)
    ax = gca;
    pos = ax.Position;
    ax.Position = [pos(1)+0.2*pos(3), pos(2), 0.8*pos(3), pos(4)];

    % add second y axis with strain times on log scale
    ax2 = axes('Position', ax.Position,...
    'YLim', logexxtime,...
    'Ydir', 'reverse',...
```

```
    'YTick', [5 6 7],...
    'YTicklabel', [5 6 7],...
    'XTick',[],...
    'YAxisLocation','right',...
    'XaxisLocation','top',...
    'Color','none');

    % store results for present-day and B&G Tessera
    RockLvalsBG(j,:) = [Lvals(E==-17&TH==10)/1000 Lvals(E==-15&TH==20)/1000];

    % store median and IQR values for specific strain rates
    F = griddedInterpolant(TH', E', Lvals'/1000);
    Es = min(logexx):1:max(logexx);
    for m = 1:length(Es)
        fun = @(th)F(th,Es(m))-lam(1);
        med = fzero(fun,40,options);
        fun = @(th)F(th,Es(m))-lam(3);
        iqr1 = fzero(fun,40,options);
        fun = @(th)F(th,Es(m))-lam(2);
        iqr2 = fzero(fun,40,options);
        RockLvals(m+(j-1)*length(Es),:) = [med iqr1 iqr2];
    end
end
```




```
% pos = fig.Position;
% fig.Position = [pos(1:2), 400, pos(3)];

% fig3.Renderer='Painters';
% saveas(fig3,'fig3','pdf')
```

Internal functions

Strength Envleope

```
function [H, gam, eta, z, env] = bdt_plot(th, Ts, g, rho, Astar, n, Q, Exx, phi)
% define brittle-ductile transition for given density (rho), gravity (g)
% thermal gradient (th), surface temperature (Ts),
```

```
% power-law rheology (Astar, n, Q), strain rate (Exx),
% and tectonic regime (phi = 'n' for normal, 'r' for reverse, 's' for
% strike slip)

% output is thickness of brittle layer (H), inverse e-folding depth (gam),
% and viscosity at the bdt (eta)

% Christmas tree plot is based on scripts written by John Townend (2001)

% define basic friction parameters for rock based on Byerlee (1978)
mu1=0.85;  %coefficient of friction (0-200 MPa)
mu2=0.6;        %coefficient of friction (>200 MPa)
F1=(sqrt(1+mu1^2)+mu1)^2;%frictional failure parameter (0-114 MPa)
F2=(sqrt(1+mu2^2)+mu2)^2;%frictional failure parameter (>114 MPa)

R=8.31446; % J/K/mol (NIST 2014)

% calculate depth to bdt

% calculate brittle strength
Ps=9.2; % surface pressure in MPa

% calculate depth of change in friction coefficient equivalent to 200 MPa
zt=(1.14e8-Ps*1e6)/(rho*g);

% set up two depth ranges with break at 200 MPa
z1=(zt:-100:0);
z1=fliplr(z1);
z2=(zt:100:100000);

% Calculate stress based on frictional strength in MPa
% Result depends on tectonic regime (phi)

if phi=='n' %normal faulting
    brit_S1_S31=(rho*g*z1*1e-6+Ps)*(F1-1)/F1;
    brit_S1_S32=(rho*g*z2*1e-6+Ps)*(F2-1)/F2;
    brit_S1_S32=brit_S1_S32+(brit_S1_S31(end)-brit_S1_S32(1)); % shift segments to meet

elseif phi=='r' %reverse faulting
    brit_S1_S31=(rho*g*z1*1e-6+Ps)*(F1-1);
    brit_S1_S32=(rho*g*z2*1e-6+Ps)*(F2-1);
    brit_S1_S32=brit_S1_S32+(brit_S1_S31(end)-brit_S1_S32(1)); % shift segments to meet

elseif phi=='s' %strike-slip faulting
    brit_S1_S31=2*(rho*g*z1*1e-6+Ps)*(F1-1)/(1+F1);
    brit_S1_S32=2*(rho*g*z1*1e-6+Ps)*(F2-1)/(1+F2);
    brit_S1_S32=brit_S1_S32+(brit_S1_S31(end)-brit_S1_S32(1)); % shift segments to meet

% if no tectonic regime is specified return error
else
    errordlg('Sorry! - The stress regime must be normal (n), strike-slip (s), or revers
end

% append brittle strength envelope to create single vector with compression
```

```
% negative
brit_S1_S3=-[brit_S1_S31 brit_S1_S32(2:end)];
z=[z1 z2(2:end)];

% calculate ductile strength
duct_S1_S3=-(abs(Exx)/Astar)^(1/n)*exp(Q./(n*R*(Ts+th*z)));

% find depth where ductile strength is less negative than brittle strength
% (crossover point)
[I]=find(duct_S1_S3>=brit_S1_S3, 1);
% set BDT to this point
H=z(I);

% calcualte inverse e-folding depth (gam) from the BDT depth, temperature,
% and rock rheology parameters
gam=Q*th/(n*R*(Ts+th*H)^2);

% calculate the viscosity at the BDT
eta=1e6*1/2*Astar^(-1/n)*abs(Exx)^-(1-1/n)*exp(Q/(n*R*(Ts+th*H)));

% store the envelope as the maximum of the [negative] strength values
env=max(duct_S1_S3, brit_S1_S3);
end
```

Stability analysis

```
function [qmax, Lmax, H]=perturb2(th, Ts, g, rho, Astar, n, Q, Exx, phi)
% function [qmax, kmax]=perturb(H, theta, Ts, Astar, n, Q, Exx)
% Implementation of Fletcher and Hallet (1983) necking/buckling model
% after implementation of Dombard and McKinnon (2001)
% qmax is the maximum growth rate of the instability of wave number kmax.
% th is the linear thermal gradient, Ts is the surface temperature, g is
% the gravitational acceleration, rho is the density, Astar, n, and Q are power-law rhe
% material constants, Exx is the horizontal strain rate.

% h=1;
qmax=-10^10; % very small initial value
n1=1e4; % plastic layer viscosity (approximating infinite)

% call subroutine to determine brittle layer thickness (H), inverse
% e-folding depth (gam) and interface viscosity (eta)

% [H, gam, eta] = bdt(th, Ts, g, rho, Astar, n, Q, Exx);
[H, gam, eta, ~, ~] = bdt_plot(th, Ts, g, rho, Astar, n, Q, abs(Exx), phi);

% Calculate S parameter from F&H
% tauy = 2*eta*abs(Exx);
% S = rho*g*H./(2*tauy);

% create vector of k values and empty q values
ks=1.4:0.05:4;
q = zeros(size(ks));
```

```
% iterate over wavenumber, k
for j = 1 : length(ks)
    k = ks(j);

    % initiaqlize P matrix that holds equation coefficients
    P=zeros(6,6);

    % make wavenmuber dimensionless
    lam=k/H;

    % initial small amplitude of topography (scaled to wavelength)
    h = .01*2*pi/lam;

    % define common terms for substrate
    m=gam/lam;
    a=sqrt((m^2/4+2/n-1+sqrt(m^4/16+m^2*(2/n+1)/2+1))/2);
    rr=sqrt(m^2/4+(n-1)/n^2);
    al=a-m/2;
    be=rr/a;

    % define common terms for surface layer
    al1=sqrt(1/n1);
    be1=sqrt(1-1/n1);

    % terms for stress equations
    M1=(4/n-1)*al-m*(1+al^2-be^2)-al*(al^2-3*be^2);
    M2=be*((4/n-1)-2*al*m+be^2-3*al^2);

    C=cos(be1*k);
    S=sin(be1*k);

    % set up matrics

    % first two rows are shear and normal stresses at surface due to
    % topography
    P(1,1)=(C-be1/al1*S)*exp(al1*k);
    P(1,2)=(be1/al1*C+S)*exp(al1*k);
    P(1,3)=(C+be1/al1*S)*exp(-al1*k);
    P(1,4)=-(be1/al1*C-S)*exp(-al1*k);
    P(2,:)=[C*exp(al1*k) S*exp(al1*k) -C*exp(-al1*k) -S*exp(-al1*k) 0 0];

    % next four are continuity equations for displacements (horizontal,
    % veritcal) and stresses (shear, normal) across interface
    P(3,:)=[al1 be1 -al1 be1 -al -be];
    P(4,:)=[1 0 1 0 -1 0];
    P(5,:)=[-2/n1 -2*al1*be1 -2/n1 2*al1*be1 1+al^2-be^2 2*al*be];
    P(6,:)=[2*al1 0 -2*al1 0 -M1 -M2];

    % vector of boundary conditions
    % (Sxz and Szz at surface; u, w, Sxz, Szz at interface)
    % Perturbation at surface
    BC=[2*n1*Exx*h -rho*g*h*H/(2*eta*al1*k) 0 0 0 0]';
```

```
    % solve for unkown coefficients using MATLAB mldivide
    A=P\BC;

%     % calculate vertical and horizontal velocities on grid
%     [X,Z] = meshgrid(-pi:pi/4:pi, 0:H/4:H);
%     W=((A(1)*cos(be1*lam*Z)+A(2)*sin(be1*lam*Z)).*exp(al1*lam*Z)+...
%         (A(3)*cos(be1*k)+A(4)*sin(be1*k))*exp(-al1*k)).*cos(lam*X);

    % calculate vertical velocity at (0 or pi,H)
    W0=sign(Exx)*((A(1)*C+A(2)*S)*exp(al1*k)+(A(3)*C+A(4)*S)*exp(-al1*k));

    % calculate growth rate factor
    q(j)=W0/(abs(Exx)*h); % corrected for contraction

    % check for max and if true store values
    if q(j)>qmax
        qmax=q(j);
        kmax=k;
    end
end

% calculate wavelength for maximum growth rate
Lmax=2*pi*H/kmax;
end
```

Mixing model

```
function [nbar, Qbar, Abar] = MPGe(phi, n, Q, A)
% function [nbar, Qbar, Abar] = MPGe(phi, n, Q, A)
%
% function form of Minimize Power Geometric Mean of Huet et al., 2014
% inputs are vectors of volume fraction (phi) and creep parameters
% (n, Q, A) for a polyphase rock composed. Output variables are averaged
% creep parameters.

% determine number of phases
nphase = length(phi);

% initialize ID variable with number for each phase
ID = 1:nphase;

% initialize "a" parameter (cumulative product of stress exponents of all
% other phases +1)
a = zeros(size(phi));

% loop over phases and calculate a
for j=1:nphase
    a(j)=prod(n(j~=ID)+1);
end

% calculate creep parameters

% average power law exponent (n)
nbar=(sum(phi.*a.*n))./(sum(phi.*a));
```

```
Qbar=(sum(phi.*a.*Q))./(sum(phi.*a));

Abar=prod((A).^((phi.*a)./sum(phi.*a)))...
    .*(sum((phi.*n)./(n+1))).^-nbar...
    .*prod((n./(n+1)).^((phi.*a.*n)./sum(phi.*a)));
end
```