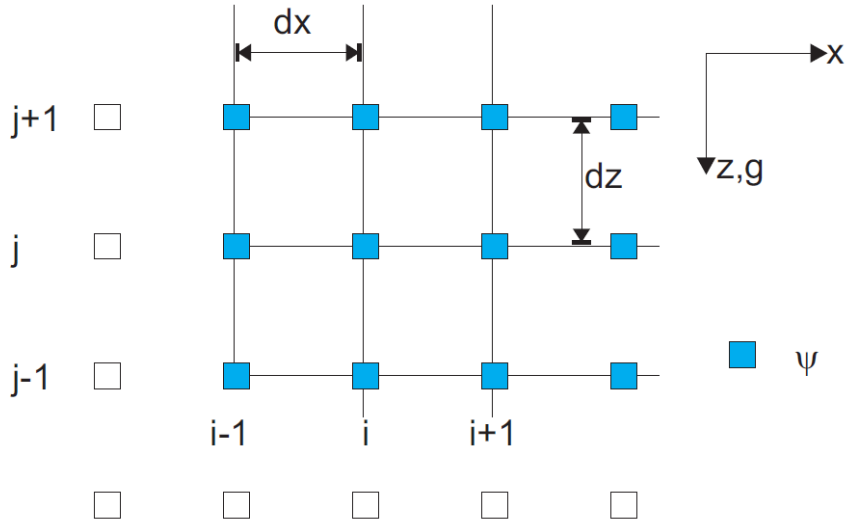

Figure 1: Discretization for the streamfunction approach. The boundary conditions are set through fictious boundary points.

1 Stokes equations with FD on a staggered grid using the stream-function approach.

1.1 Introduction

As was discussed in sec. **??**, the basis of basically all mantle convection and lithospheric dynamics codes are the Stokes equations for slowly moving viscous fluids.

There are several ways to solve those equations, and the goal of this exercise is to use a streamfunction, finite difference approach. Stream function means that there is a potential field which we solve for, and then obtain velocities from the derivatives of this field. The advantage of this approach is that the continuity equation for incompressible flow can be satisfied implicitly, rather than having to use a panelty parameter as for the primitive variable approach of sec. **??**. (It is, however, possible to formulate the stream function method for compressible convection approximations, *e.g.* *Schmeling*, 1989). For a comparison of different finite difference approaches, see *Deubelbeiss and Kaus* (2008), for example.

The main challenges of this project are, 1), having fairly high-order and mixed derivatives (up to 4^{th} order) and, 2), setting of boundary conditions.

1.2 Governing equations

It is assumed that the rheology is incompressible and that the rheology is Newtonian viscous. In this case, the governing equations are (see sec. **??**):

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_z}{\partial z} = 0 \tag{1}$$
$$\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{xz}}{\partial z} = 0 \tag{2}$$
$$\frac{\partial \sigma_{xz}}{\partial x} + \frac{\partial \sigma_{zz}}{\partial z} - \rho g = 0 \tag{3}$$
$$\sigma_{xx} = -p + 2\mu \frac{\partial v_x}{\partial x} \tag{4}$$
$$\sigma_{zz} = -p + 2\mu \frac{\partial v_z}{\partial z} \tag{5}$$
$$\sigma_{xz} = \mu \left(\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x} \right) \tag{6}$$

By substituting eqs. (4)-(6) into eqs. (1)-(3), we obtain (compare sec. **??**)

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_z}{\partial z} = 0 \tag{7}$$
$$-\frac{\partial p}{\partial x} + 2\frac{\partial}{\partial x}\left(\mu \frac{\partial v_x}{\partial x}\right) + \frac{\partial}{\partial z}\left(\mu \left(\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x} \right)\right) = 0 \tag{8}$$
$$-\frac{\partial p}{\partial z} + 2\frac{\partial}{\partial z}\left(\mu \frac{\partial v_z}{\partial z}\right) + \frac{\partial}{\partial x}\left(\mu \left(\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x} \right)\right) = \rho g \tag{9}$$

We can eliminate pressure from eqs. (8) and (9) by taking the derivative of eq. (8) versus z and subtracting eq. (9) derived versus x. This results in:

$$2\frac{\partial^2}{\partial x \partial z}\left(\mu \frac{\partial v_x}{\partial x}\right) - 2\frac{\partial^2}{\partial x \partial z}\left(\mu \frac{\partial v_z}{\partial z}\right) +$$
$$\frac{\partial^2}{\partial z^2}\left(\mu \left(\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x} \right)\right) - \frac{\partial^2}{\partial x^2}\left(\mu \left(\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x} \right)\right) = -\frac{\partial}{\partial x}\rho g. \tag{10}$$

We can also use the incompressibility constraint (7) to simplify things a little bit more:

$$-4\frac{\partial^2}{\partial x \partial z}\left(\mu \frac{\partial v_z}{\partial z}\right) +$$
$$\frac{\partial^2}{\partial z^2}\left(\mu \left(\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x} \right)\right) - \frac{\partial^2}{\partial x^2}\left(\mu \left(\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x} \right)\right) = -\frac{\partial}{\partial x}\rho g \tag{11}$$

Now we introduce a variable Ψ (the stream function) which is defined by its relationship to the velocities as

$$v_x = \frac{\partial \Psi}{\partial z} \tag{12}$$
$$v_z = -\frac{\partial \Psi}{\partial x} \tag{13}$$

Note that Ψ satisfies incompressibility by plugging eqs. (12) and (13) into eq. (1).

By using Ψ, we can write eq. (11) as:

$$\begin{aligned} 4\frac{\partial^2}{\partial x \partial z}\left(\mu \frac{\partial^2 \Psi}{\partial x \partial z}\right) \quad + \\ \frac{\partial^2}{\partial z^2}\left(\mu\left(\frac{\partial^2 \Psi}{\partial z^2} - \frac{\partial^2 \Psi}{\partial x^2}\right)\right) - \frac{\partial^2}{\partial x^2}\left(\mu\left(\frac{\partial^2 \Psi}{\partial z^2} - \frac{\partial^2 \Psi}{\partial x^2}\right)\right) \quad = \quad -\frac{\partial}{\partial x}\rho g. \end{aligned} \tag{14}$$

Note that this equation now has 4^{th} order derivatives for Ψ (easier to see for constant μ, where we can pull the viscosity out of the derivatives.) The challenge is to solve eq. (14) for Ψ given then density gradients.

1.3 Exercise

a) Discretize eq. (14) on a grid as shown on Figure 1.

b) A MATLAB subroutine is shown on Figure 2. The subroutine sets up the grid and the node numbering. Finish the code by programming the discretized eq. (14). To start simple, assume that viscosity is constant.

c) Add free-slip boundary conditions on all sides (which means $v_z = 0, \sigma_{xz} = 0$ on the lower and upper boundaries and $\sigma_{xz} = 0, v_x = 0$ on the side boundaries; you'll have to write these equations in terms of Ψ and employ fictious boundary points).

d) Assume a model domain $x = [0;1], \quad z = [0;1]$, and assume that the density below $z = 0.1\cos(2\pi x) + 0.5$ is 1, whereas the density above it is 2. Compute the velocity, and plot the velocity vectors.

e) Write the code for the case of variable viscosity (which is relevant for the Earth since rock properties are a strong function of temperature).

```
% Solve the 2D Stokes equations on a staggered grid, using the Vx,Vz,P
% formulation.
clear
% Material properties         phase #1 phase #2
mu_vec =  [1 1 ];
rho_vec = [1 2 ];
% Input parameters
Nx = 6;
Nz = 6;
W = 1;H = 1;g = 1;
% Setup the interface
x_int = 0:.01:W;
z_int = cos(x_int*2*pi/W)*1e-2 - 0.5;
% Setup the grids-------------------------------------------------------------
dz = H/(Nz-1);dx = W/(Nx-1);
[X2d,Z2d] = meshgrid(0:dx:W,-H:dz:0);
%--------------------------------------------------------------------------
% Compute material properties from interface----------------------------
% Properties are computed in the center of a control volume
Rho = ones(Nz,Nx)*rho_vec(2);
Mu = ones(Nz,Nx)*mu_vec(2);
z_int_intp = interp1(x_int,z_int,X2d(1,:));
for ix = 1:length(z_int_intp)
  ind = find(Z2d(:,1)<z_int_intp(ix));
  Rho(ind(1:end-1),ix) = mu_vec(1);
  Mu(ind(1:end-1),ix) = rho_vec(1);
  fac = (z_int_intp(ix) - Z2d(ind(end),1))/dz;
  Rho(ind(end),ix) = fac*rho_vec(1) + (1-fac)*rho_vec(2);
  Mu(ind(end),ix) = fac*mu_vec( 2) + (1-fac)*mu_vec( 2);
end
%--------------------------------------------------------------------------
% Setup numbering scheme------------------------------------------------------
Number_ind = zeros(Nz, Nx); % Create the general numbering scheme
num = 1;
for ix=1:Nx
  for iz=1:Nz
    Number_ind(iz,ix) = num;
    num = num+1;
  end
end
num_eqns = num-1;
%--------------------------------------------------------------------------
% Setup the stiffness matrix
A = sparse(num_eqns,num_eqns);Rhs_vec = zeros(num_eqns,1);
% Compute coefficients for mu*d4 Psi/dx4
ind_list = [];ind_val = [];
mu = mu_vec(1);
[ind_list,ind_val] = Add_coeffs(ind_list,ind_val, Number_ind(1:end,5:end ), mu*( 1/dx4));
[ind_list,ind_val] = Add_coeffs(ind_list,ind_val, Number_ind(1:end,4:end-1), mu*(-4/dx4));
[ind_list,ind_val] = Add_coeffs(ind_list,ind_val, Number_ind(1:end,3:end-2), mu*( 6/dx4));
[ind_list,ind_val] = Add_coeffs(ind_list,ind_val, Number_ind(1:end,2:end-3), mu*(-4/dx4));
[ind_list,ind_val] = Add_coeffs(ind_list,ind_val, Number_ind(1:end,1:end-4), mu*( 1/dx4));
% Compute coefficients for d4 Psi/dx2/dz2
% compute coefficients for ....
% Add local equations to global matrix
ii = 1;
for i=1:size(ind_list,2)
  A = A + sparse([ii+1:ii+size(ind_list,1)].,ind_list(:,i),ind_val(:,i),num_eqns,num_eqns);
end
% set rhs
%Rhs_vec([ii+1:ii+size(ind_list,1)]) = Rhs_vec([ii+1:ii+size(ind_list,1)]) + rho(:)*g;
% Set boundary conditions
% Solve system of equations.
```

Figure 2: Code `Streamfunction_Stokes.m` that initializes the grid and node numbering for the 2D streamfunction approach.

```
function [ind_list,ind_val] = Add_coeffs(ind_list,ind_val,ind_add,val_add)% Add coefficients to an array
if (length(val_add(:))==1)
   val_add  =   ones(size(ind_add))*val_add;
end
ind_list    =   [ind_list, ind_add(:)];ind_val     =   [ind_val , val_add(:)];
```

Figure 3: MATLAB script `Add_coeffs.m`, used by `Streamfunction_Stokes.m`.

Bibliography

Deubelbeiss, Y., and B. J. P. Kaus (2008), Comparison of Eulerian and Lagrangian numerical techniques for the Stokes equations in the presence of strongly varying viscosity, *Phys. Earth Planet. Inter.*, *171*, 92–111.

Schmeling, H. (1989), Compressible convection with constant and variable viscosity: The effect on slab formation, geoid, and Topography, *J. Geophys. Res.*, *94*, 12,463–12,481.