LaMEM short course

16-20 02 2026 Heidelberg

Nicolas Riel- nriel@uni-mainz.de

How to create a LaMEM model?

```
seph42@DESKTOP-2V82075:~/LaMEM_course/01_falling_block_isoviscous$ julia
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.10.0 (2023-12-25)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

julia> using LaMEM

julia> Model()
LaMEM Model setup
|
|-- Scaling             :  GeoParams.Units.GeoUnits{GeoParams.Units.GEO}
|-- Grid                :  nel=(16, 16, 16); xϵ(-10.0, 10.0), yϵ(-5.0, 5.0), zϵ(-10.0, 0.0)
|-- Time                :  nstep_max=50; nstep_out=1; time_end=1.0; dt=0.05
|-- Boundary conditions :  noslip=[0, 0, 0, 0, 0, 0]
|-- Solution parameters :  eta_min=1.0e18; eta_max=1.0e25; eta_ref=1.0e20; act_temp_diff=0
|-- Solver options      :  direct solver; superlu_dist; penalty term=10000.0
|-- Model setup options :  Type=files;
|-- Output options      :  filename=output; pvd=1; avd=0; surf=0
|-- Materials           :  0 phases;


julia>
```

e.g. BoundaryConditions

Access information by typing:

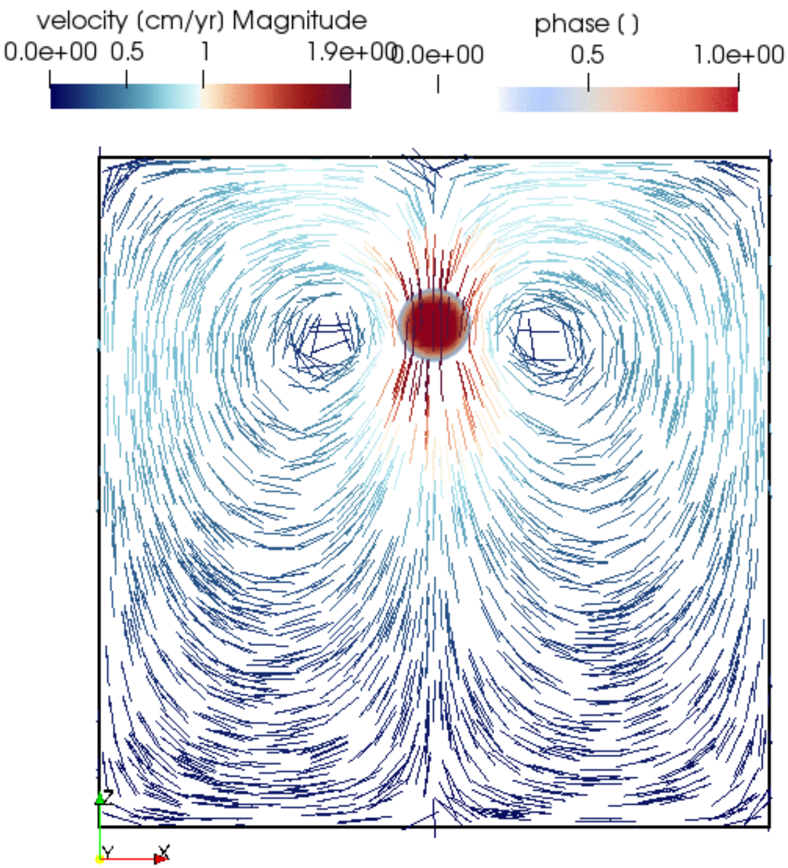
```
julia> BoundaryConditions()
LaMEM Boundary conditions :
  noslip             = [0, 0, 0, 0, 0, 0]
  open_top_bound     = 0
  temp_top           = 0.0
  temp_bot           = 1300.0
  exx_num_periods    = 3
  exx_time_delims    = [0.1, 5.0]
  exx_strain_rates   = [1.0e-15, 2.0e-15, 1.0e-15]
  eyy_num_periods    = 2
  eyy_time_delims    = [1.0]
  eyy_strain_rates   = [1.0e-15, 2.0e-15]
  exy_num_periods    = 2
  exy_time_delims    = [1.0]
  exy_strain_rates   = [1.0e-15, 2.0e-15]
  exz_num_periods    = 2
  exz_time_delims    = [1.0]
  exz_strain_rates   = [1.0e-15, 2.0e-15]
  eyz_num_periods    = 2
  eyz_time_delims    = [1.0]
  eyz_strain_rates   = [1.0e-15, 2.0e-15]
  bg_ref_point       = [0.0, 0.0, 0.0]
  VelocityBoxes      = VelocityBox[]
```

There is a bunch of available parameters, but what do they mean?

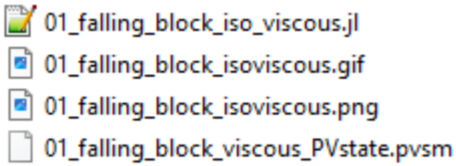
e.g. BoundaryConditions

```
help?> BoundaryConditions
search: BoundaryConditions

  Structure that contains the LaMEM boundary conditions information.

    • noslip::Vector{Int64}: No-slip boundary flag mask (left right front back bottom top)

    • open_top_bound::Int64: Stress-free (free surface/infinitely fast erosion) top boundary flag

    • temp_top::Float64: Constant temperature on the top boundary

    • temp_bot::Float64: Constant temperature on the bottom boundary

    • exx_num_periods::Int64: number intervals of constant background strain rate (x-axis)

    • exx_time_delims::Vector{Float64}: time delimiters (one less than number of intervals, not required for one interval)

    • exx_strain_rates::Vector{Float64}: strain rates for each interval

    • eyy_num_periods::Int64: eyynumperiods

    • eyy_time_delims::Vector{Float64}: eyytimedelims

    • eyy_strain_rates::Vector{Float64}: eyystrainrates

    • exy_num_periods::Int64: exynumperiods

    • exy_time_delims::Vector{Float64}: exytimedelims

    • exy_strain_rates::Vector{Float64}: exystrainrates

    • exz_num_periods::Int64: exznumperiods

    • exz_time_delims::Vector{Float64}: exztimedelims

    • exz_strain_rates::Vector{Float64}: exzstrainrates

    • eyz_num_periods::Int64: eyznumperiods

    • eyz_time_delims::Vector{Float64}: eyztimedelims

    • eyz_strain_rates::Vector{Float64}: eyzstrainrates

    • bg_ref_point::Vector{Float64}: background strain rate reference point (fixed)

    • VelocityBoxes::Vector{VelocityBox}: List of added velocity boxes
```

Falling sphere: introduction



Directory: 01_falling_block_isoviscous



Model setup: overview

model = Model(…)

1. Boundary conditions
2. Set outputted variables
3. Set timestep info

Initial phase and temperature conditions

1. Set rock-type / initial geometry
2. Set starting temperature conditions

Material properties
material = Phase(…)

1. Set rheology
2. Thermal parameters (alpha, cp, density)
3. Add Phases to ‘model’

Model setup: overview

01_falling_block_isoviscous.jl

PART1: package and output directory

Bunch of comments useful to keep track of what does the script

List the packages used for the simulation

Output_directory (relative to REPL position)

```
################################
# 01_falling_block_iso_viscous NR.02-24
# simple setup to simulate a dense viscous sphere falling down a less v
#
#
# if you have blue wavy underline below the model definition lines add
################################


# load needed packages, GeophysicalModelGenerator is used to create sha
using LaMEM, GeophysicalModelGenerator, GMT, Plots


# directory you want your simulation's output to be saved in
out_dir = "output"

```

Model setup: overview

PART2: model setup

<u>Structure holding model infos</u>

Model dimensionalization

Model size and resolution

Thermal and mechanical
Boundary conditions

Time and timestep definition

Other model parameters

Outputted variables

Solver options

```
model = Model(
                    # Scaling paramters, this ensure non-dimensionalisation in LaM
                    Scaling(GEO_units(  temperature     = 1000,
                                        stress          = 1e9Pa,
                                        length          = 1km,
                                        viscosity       = 1e20Pa*s) ),

                    # This is where you setup the size of your model (as km as set
                    Grid(               x               = [-50.0,50.0],
                                        y               = [-1.0,1.0],
                                        z               = [-100.0,0.0],
                                        nel             = (96,1,96) ),

                    # sets the conditions at the walls of the modelled domain
                    BoundaryConditions( temp_bot        = 20.0,
                                        temp_top        = 20.0,
                                        open_top_bound  = 0,
                                        noslip          = [0, 0, 0, 0, 0, 0]),

                    # set timestepping parameters
                    Time(               time_end        = 10.0,
                                        dt              = 0.01,
                                        dt_min          = 0.000001,
                                        dt_max          = 0.1,
                                        nstep_max       = 80,
                                        nstep_out       = 1 ),

                    # set solution parameters
                    SolutionParams(     eta_min         = 1e19,
                                        eta_ref         = 1e20,
                                        eta_max         = 1e22),

                    # what will be saved in the output of the simulation
                    Output(             out_density         = 1,
                                        out_melt_fraction   = 1,
                                        out_j2_strain_rate  = 1,
                                        out_temperature     = 1,
                                        out_surf_velocity   = 1,
                                        out_dir             = out_dir ),

                    # here we define the options for the solver, it is advised to
                    Solver(             SolverType          = "direct",
                                        DirectSolver        = "mumps" )
                )
```

Model setup: overview

PART3: materials properties (shapes and rheologies)

Set background phase

Add ellipsoid shape

Creates mantle phase

Creates eclogite phase

Add phases to the model

```
#================== define phases (different materials) of the model ========================#

model.Grid.Phases                  .= 0;                    # here we first define the bac

add_ellipsoid!(model,   cen           = (0,0,-25),           # defines centre of the spher
                        axes          = (5,5,5),             # define size of the sphere
                        StrikeAngle   = 0,
                        DipAngle      = 0,
                        phase         = ConstantPhase(1),    # we attribute phase id = 1, n
                        T             = ConstantTemp(20));


#===================== define material properties of the phases ==========================#

mantle = Phase(         Name          = "Mantle",            # let's call phase 0 mantle
                        ID            = 0,                   # not that ID here points to p
                        rho           = 3300,                # set mantle density
                        eta           = 1e20,                # set mantle viscosity
                        G             = 5e10 );              # set elastic modulii

eclogite = Phase(       Name          = "eclogite",
                        ID            = 1,
                        rho           = 4000,
                        eta           = 1e24,
                        G             = 5e10 );

add_phase!( model, mantle, eclogite )                              # this adds the phases

plot_cross_section(model, y=0, field=:phase)
savefig("01b_falling_block_isoviscous_with_tracers.png")
#============================== perform simulation ========================================#

run_lamem(model, 2)
```

Model setup: overview

PART3: materials properties (shapes and rheologies)

Set background phase

Add ellipsoid shape

Creates mantle phase

Creates eclogite phase

Add phases to the model

Saves a cross-section through your setup

Perform simulation using 2 cores

```
#================== define phases (different materials) of the model ========================#

model.Grid.Phases                    .= 0;                    # here we first define the bac

add_ellipsoid!(model,    cen           = (0,0,-25),           # defines centre of the spher
                         axes          = (5,5,5),             # define size of the sphere
                         StrikeAngle   = 0,
                         DipAngle      = 0,
                         phase         = ConstantPhase(1),    # we attribute phase id = 1, n
                         T             = ConstantTemp(20));


#==================== define material properties of the phases ==========================#

mantle = Phase(          Name          = "Mantle",            # let's call phase 0 mantle
                         ID            = 0,                   # not that ID here points to p
                         rho           = 3300,                # set mantle density
                         eta           = 1e20,                # set mantle viscosity
                         G             = 5e10 );              # set elastic modulii

eclogite = Phase(        Name          = "eclogite",
                         ID            = 1,
                         rho           = 4000,
                         eta           = 1e24,
                         G             = 5e10 );

add_phase!( model, mantle, eclogite )                                 # this adds the phases

plot_cross_section(model, y=0, field=:phase)
savefig("01b_falling_block_isoviscous_with_tracers.png")
#============================== perform simulation ========================================#

run_lamem(model, 2)
```

Simulation

- Execute the script “01_falling_block_iso_viscous.jl”

option1

Copy and paste
In the julia REPL

```
seph42@DESKTOP-2V82075:~/LaMEM_course/01_falling_block_isoviscous$ julia
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.10.0 (2023-12-25)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

julia> # load needed packages, GeophysicalModelGenerator is used to create shapes
model before running the simulation
       using LaMEM, GeophysicalModelGenerator, GMT, Plots
```

option2

Use include()

```
seph42@DESKTOP-2V82075:~/LaMEM_course/01_falling_block_isoviscous$ julia
^[[A           _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.10.0 (2023-12-25)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

julia> include("01_falling_block_iso_viscous.jl")
Loading GMT routines within GMG
WARNING: using GMT.meshgrid in module GeophysicalModelGenerator conflicts with an existing identifier.
Adding Plots.jl plotting extensions for LaMEM
```

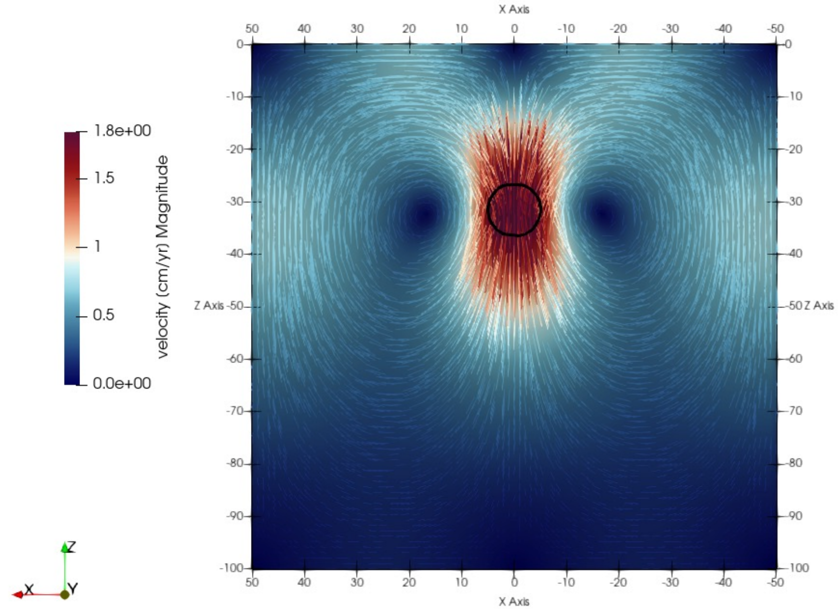
option3

Execute the script
In the terminal

```
seph42@DESKTOP-2V82075:~/LaMEM_course/01_falling_block_isoviscous$ julia 01_falling_block_iso_viscous.jl
Loading GMT routines within GMG
WARNING: using GMT.meshgrid in module GeophysicalModelGenerator conflicts with an existing identifier.
Adding Plots.jl plotting extensions for LaMEM
```

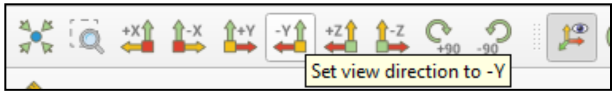
Visualization

- Use Paraview to visualize the results of the simulation and get a similar result:

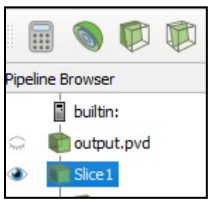


Tips:

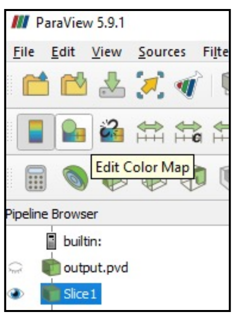
1 Set view direction to –Y



2 Slice "output.pvd" Y normal

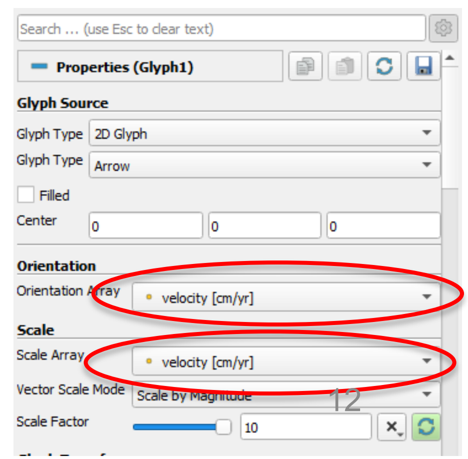


3 Display velocity field on the slice and change colormap



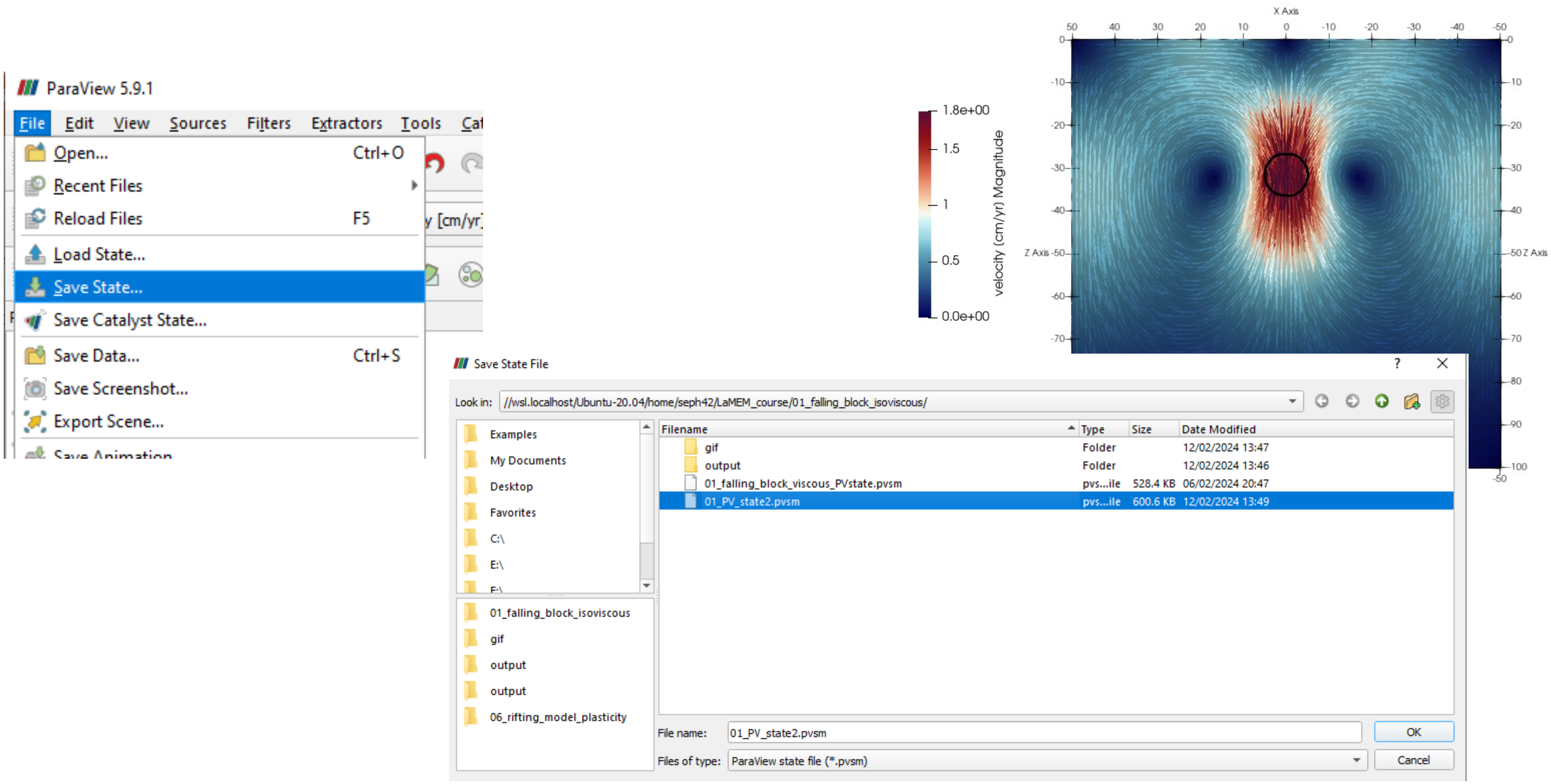
4 Contour phase (select slice)

5 Add glyph use velocity



Visualization - save Paraview state

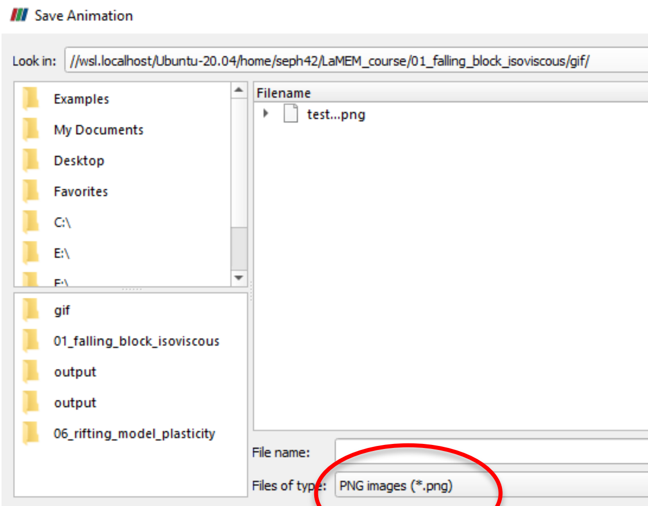
- Saving “Paraview state” allow to re-use it for other (similar) simulation



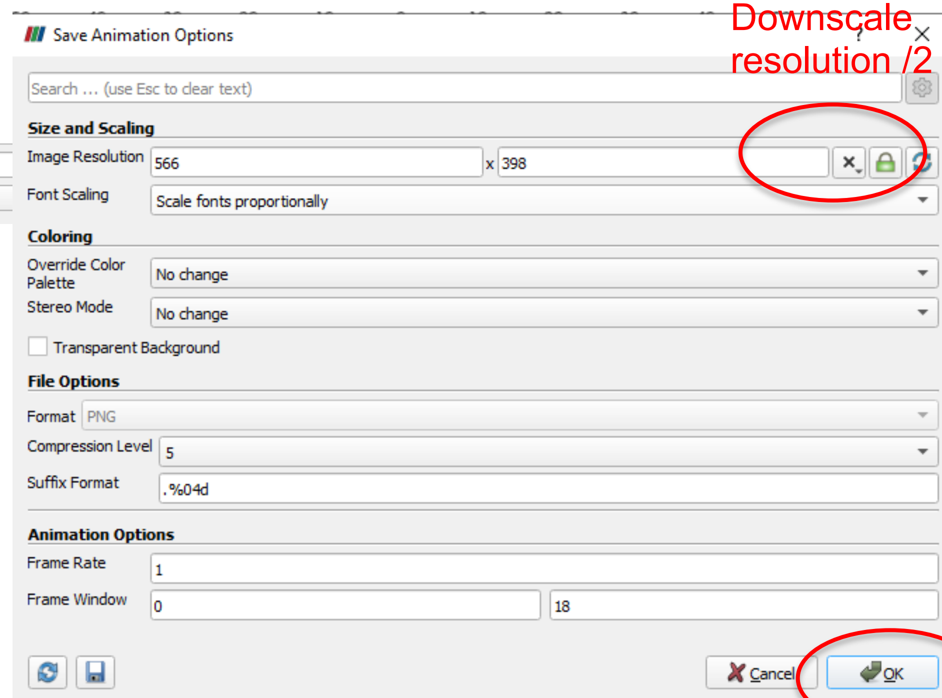
- This is important as you don’t want to redo the visualization process from scratch all the time!

Visualization

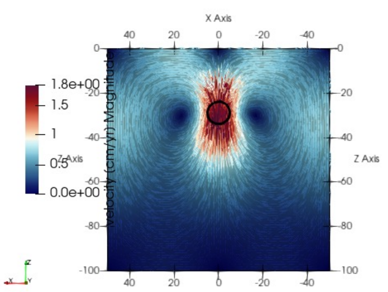
- If you are happy with how it looks like, save an animation
- First create a directory next to “output” called “gif”



Choose png



Downscale resolution /2



Create a gif

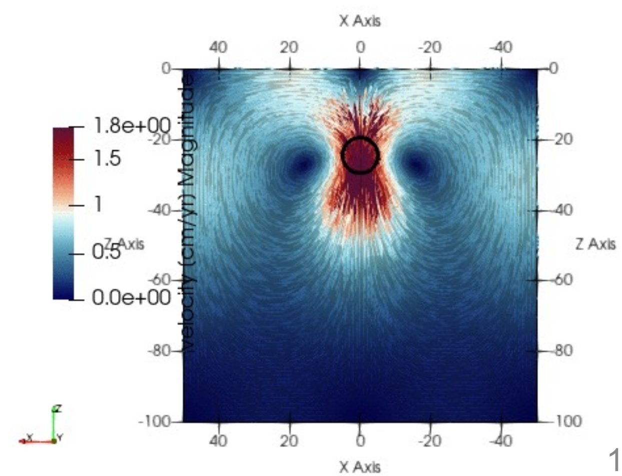
- Install imagemagick (open a terminal in VS-code)

```
sudo apt-get update
sudo apt-get install imagemagick
```

```
brew install imagemagick
```

- Go in gif directory (in the terminal using VS-code)

```
convert -delay 2 -loop 0 *.png ../01_falling_block_isoviscous.gif
```



Getting used to shapes and phases

- Copy and paste the falling sphere setup to a new directory
- Change the shape of the falling sphere to falling square:
  - Instead of using add_ellipsoid!(), use add_box!(). Think about using **Julia> ?add_box!** To get help!

```
add_box!(model;  xlim=(minX, maxX),
                ylim=(minY, maxY),
                zlim=(minZ, maxZ),

                Origin                  = nothing,
                StrikeAngle             = 0,
                DipAngle                = 0,
                phase                   = ConstantPhase(1),
                T                       = ConstantTemp(20) )
```

- Perform the simulation!
- Test out several options:
  - Change size (xlim and zlim) and DipAngle
  - Add a new phase (2) sphere or box, partially overlapping phase 1 (eclogite). Don't forget to add it to the add_phase!()
  - Change the lower half of the box into a lower density material (with respect to the upper half)