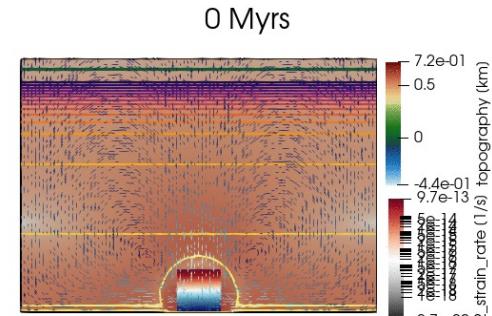
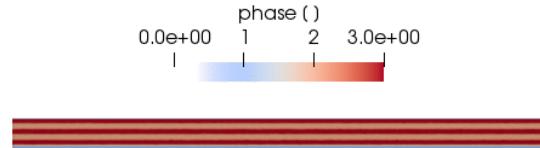
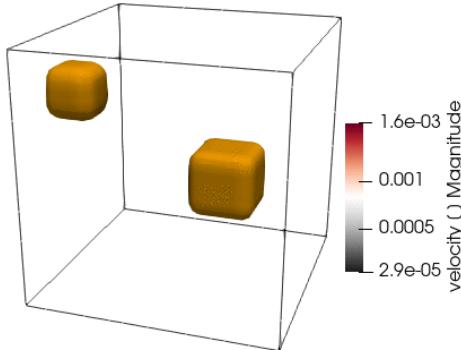


LaMEM short course

17-21 02 2025 Heidelberg
Nicolas Riel- nriel@uni-mainz.de

Objective of the course



- Learn how to use a geodynamic modeling code to simulate a wide range of 2D/3D scenarios
- To do so, you will use
 - A geodynamic modeling code (LaMEM – 90%)
 - A landscape evolution code (FastScape – 10%)
 - A programming environment (Visual studio)
 - A programming language (Julia)
 - Visualization tools (Paraview, and PlotlyJS (a julia package))
 - A tool to generate density diagram used in LaMEM (MAGEMinApp)

Structure/Organization of the short course

Monday

Morning

- Introduction
- Installation
- First model

Afternoon

- Julia introduction
- Create first setup(s)

Tuesday

Morning

- Building up complexity

Afternoon

- Advanced setup: plume

Wednesday

Morning

- Fold and thrust

Afternoon

- Rifting
- GeophysicalModelGenerator

Thursday

Morning

- Landscape evolution modelling

Afternoon

- Advanced setup: subduction

Friday

Morning and afternoon

- build your own setup

Some tips (based on experience)

- Nothing we are going to do is “complicated” on its own
 - *You don't need advanced mathematical background*
 - *Programming experience will help but is not necessary,*
 - *Copy and paste and modify as much as possible!*
- Complexity arises from chaining several simple tasks
- If you don't do it yourself, you are not going to learn
 - *There is not way around having bugs and solving them*
- When there is a bug, try to understand the error message
 - *Start at the beginning of the error message!*
 - *If this still does not help, try to come back to a working version*
 - *Ask for help!*
- Spend a bit of time making your scripts readable for you
 - *Making it look good helps*
 - *Adding comments is important*
- Modeling is like Lego: you can do a lot, but you are constrained by the building blocks!

Some tips (based on experience)

- Use ChatGPT when you get stuck for common task.

Mind that ChatGPT is nothing more than a complex interpolation. If there are a lot of data, the answer may be useful

You
given a 3D array Z, write a julia script that extract all the values > 2

ChatGPT
Certainly! Below is the Julia script to extract all the values greater than 2 from a 3D array:

```
julia
function extract_values_gt_2(arr::Array{Float64, 3})
    return arr[arr .> 2]
end

# Example usage:
# Define a 3D array
Z = rand(-5:5, 3, 3, 3)

# Extract values greater than 2
result = extract_values_gt_2(Z)

println("Values greater than 2 in the 3D array:")
println(result)
```

In this script, `extract_values_gt_2` is a function that takes a 3D array `arr` as input and returns an array containing all the values greater than 2. The `arr .> 2` statement creates a boolean mask indicating which elements of `arr` are greater than 2. This mask is then used to extract the desired values from the original array. Finally, we print the extracted values.

□ ○ ♡

This is how to do!

Mostly useless

Some tips (based on experience)

- Once you have a prompt that is useful update it

For instance:

A You
let's say that I now want to extract all the values > 2, and smaller than 3.6, update the script

ChatGPT
Sure, you can modify the script to extract values greater than 2 and smaller than 3.6 using logical operators. Here's the updated script:

```
julia
function extract_values_between_2_and_3_6(arr::Array{Float64, 3})
    return arr[(arr .> 2) .& (arr .< 3.6)]
end

# Example usage:
# Define a 3D array
Z = rand(-5:5, 3, 3, 3)

# Extract values between 2 and 3.6
result = extract_values_between_2_and_3_6(Z)

println("Values between 2 and 3.6 in the 3D array:")
println(result)
```

In this modified version, the function `extract_values_between_2_and_3_6` now returns values that satisfy both conditions: greater than 2 and smaller than 3.6. The logical operator `.&` is used to perform element-wise AND operation between the boolean arrays generated by the conditions `arr .> 2` and `arr .< 3.6`. This ensures that only elements that satisfy both conditions are selected.

This is how to do!

Mostly
useless

Very useful

Some tips (based on experience)

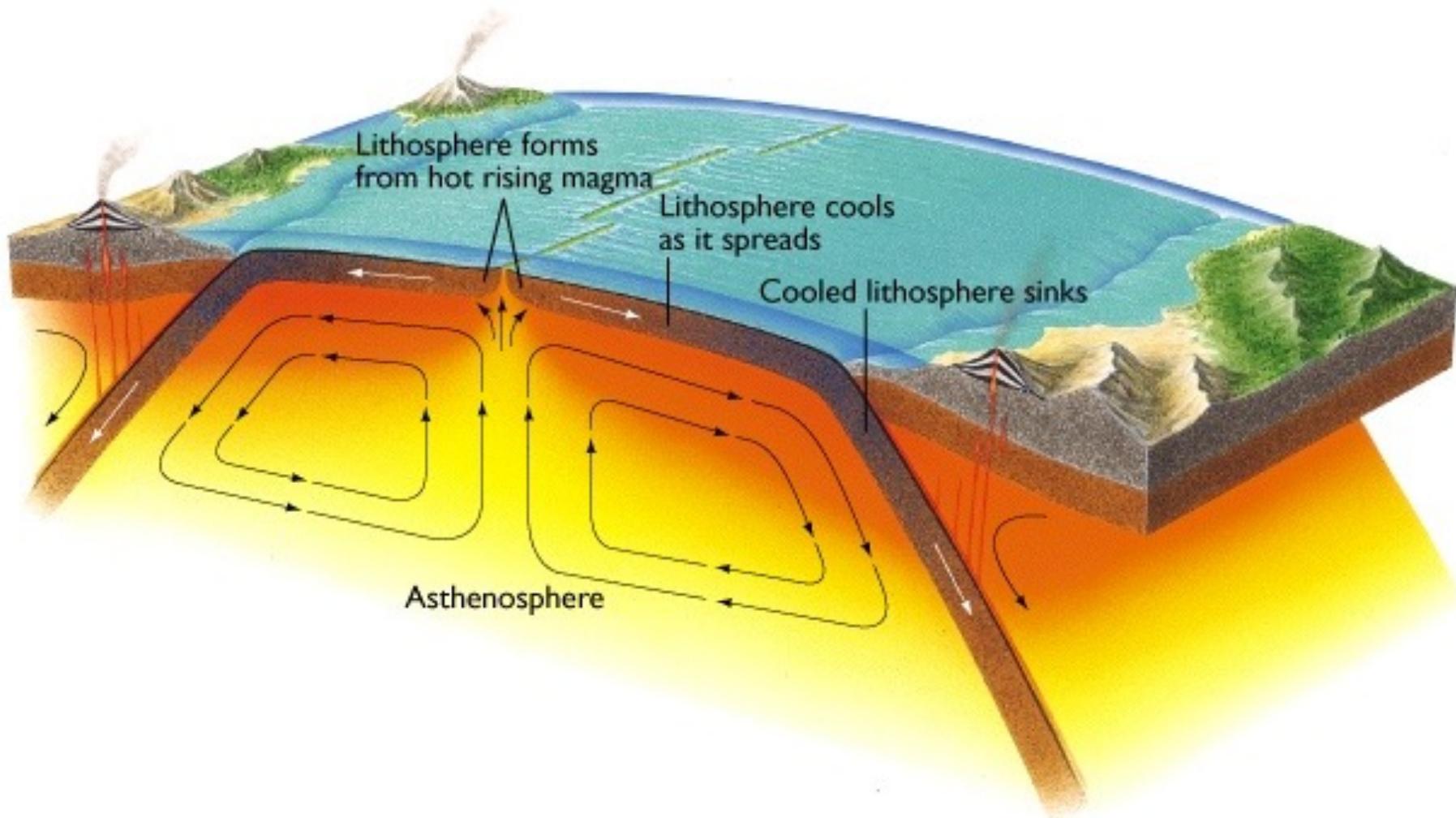
- ChatGPT knows nothing about how to set a LaMEM model up, or any other scientific modelling software in general
- It is however useful to help with Julia programming e.g.
 - How to navigate between different Julia terminal modes such as calculation mode, package manager, shell and help
 - Help for array creation, access and modification which will use extensively
 - May be helpful to better understand Julia related error message (in some cases)
 - Can also potentially help in finding out some syntax mistakes

LaMEM

Lithosphere and Mantle Evolution Model

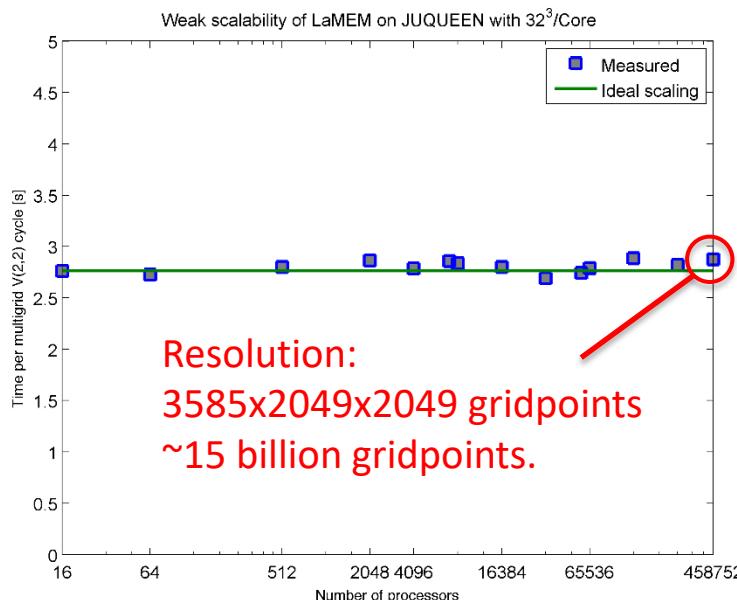
General overview

Geodynamic modelling



What is LaMEM?

- **Lithosphere and Mantle Evolution Model**
- 3D thermo-mechanical code, written in C/PETSc.
- Nonlinear visco-elasto-visco-plastic rheologies
- MPI-parallel; runs on 1-458'752 processors.
- Staggered finite difference method (faster than FEM).
- Can use a large variety of (multigrid) solvers (Galerkin GMG, AMG, Coupled/decoupled)
- Marker-and-cell method, free surface, (coupled to simple erosion model).
- Phase transitions
- Polygonal meshes (geomIO) to create (complex) 3D input geometries.



Equations

$$\frac{\partial u_i}{\partial x_i} = -\frac{1}{K} \cdot \frac{dP}{dt}$$

conservation of mass

$$-\rho g_i = -\frac{\partial P}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j}$$

conservation of momentum

$$\dot{\varepsilon}_{ij} = \frac{1}{2\eta_{eff}} \tau_{ij} + \frac{1}{2G} \frac{D\tau_{ij}}{Dt} + \dot{\lambda} \frac{\partial Q}{\partial \sigma_{ij}} \quad \text{visco-elasto-plastic rheology}$$

$$\rho c \frac{DT}{Dt} = \frac{\partial}{\partial x_i} \left(k \frac{\partial T}{\partial x_i} \right) + H_{sources}$$

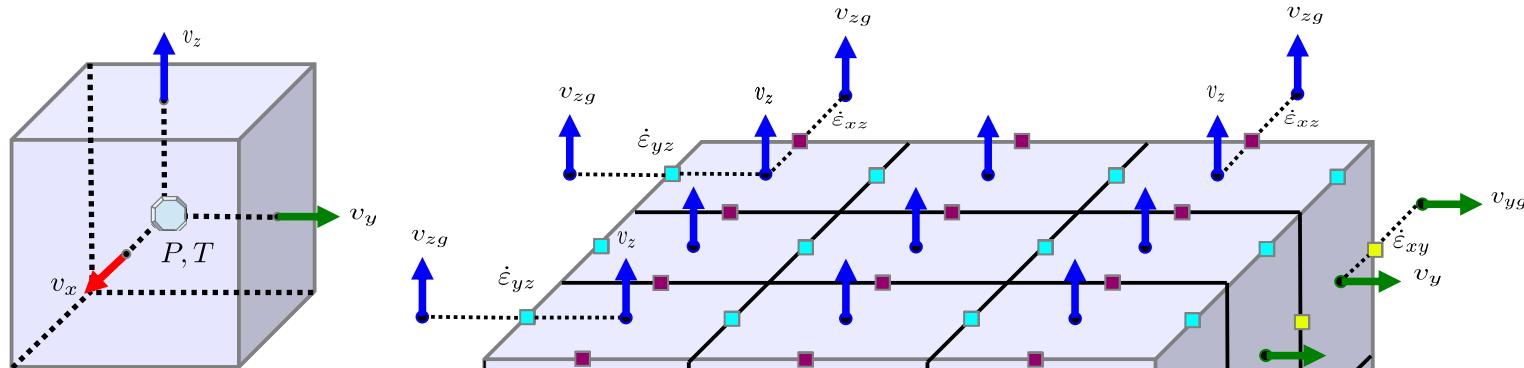
conservation of energy

“Stokes” flow

$$\begin{aligned} \frac{\partial u_i}{\partial x_i} &= 0 \\ -\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j} \left(\eta_{eff} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \right) &= -\rho g_i \end{aligned} \longrightarrow \begin{pmatrix} \mathbf{K} & \mathbf{G} \\ \mathbf{G}^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{p} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{g} \end{pmatrix}$$

- Complexities mainly from the nonlinear rheology.
- Typically need to be solved numerically.
- No time derivative in Stokes equations!
 - For given density and viscosity distribution, velocity is given everywhere

Parallel staggered grid layout implementation



Equations:

$$\frac{\partial \tau_{ij}}{\partial x_j} - \frac{\partial P}{\partial x_i} + \rho g_i = 0$$

$$\frac{1}{\rho} \frac{D\rho}{Dt} + \frac{\partial v_i}{\partial x_i} = 0$$

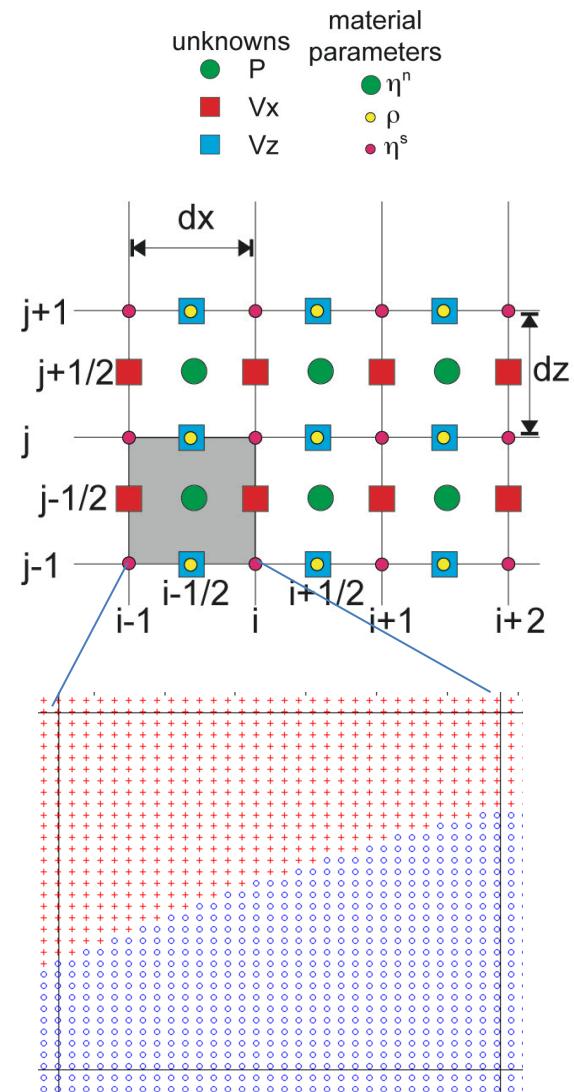
$$\rho C_p \frac{DT}{Dt} = \frac{\partial}{\partial x_i} \left(\lambda \frac{\partial T}{\partial x_i} \right) + H$$

Legend:

- v_x, q_x
- v_y, q_y
- v_z, q_z
- $\dot{\varepsilon}_{xy}, \tau_{xy}, \eta_{xy}$
- $\dot{\varepsilon}_{xz}, \tau_{xz}, \eta_{xz}$
- $\dot{\varepsilon}_{yz}, \tau_{yz}, \eta_{yz}$
- $P, \dot{\varepsilon}_{xx}, \dot{\varepsilon}_{yy}, \dot{\varepsilon}_{zz}$
 $\tau_{xx}, \tau_{yy}, \tau_{zz}, \eta_{cen}$
- T, ρ, C_p, λ, H

Marker & Cell approach

- Rock types are tracked by markers (called “Particles” in LaMEM).
- Each particle:
 - Is of a certain rocktype (or “phase ID”)
 - Contains info such as temperature, stress etc.
- Are advected with the flow
- During every timestep, the particles are mapped back to the computational grid, and vice versa



Nonlinear visco-elasto-plastic rheology

Drucker-Prager yield stress (brittle faulting)

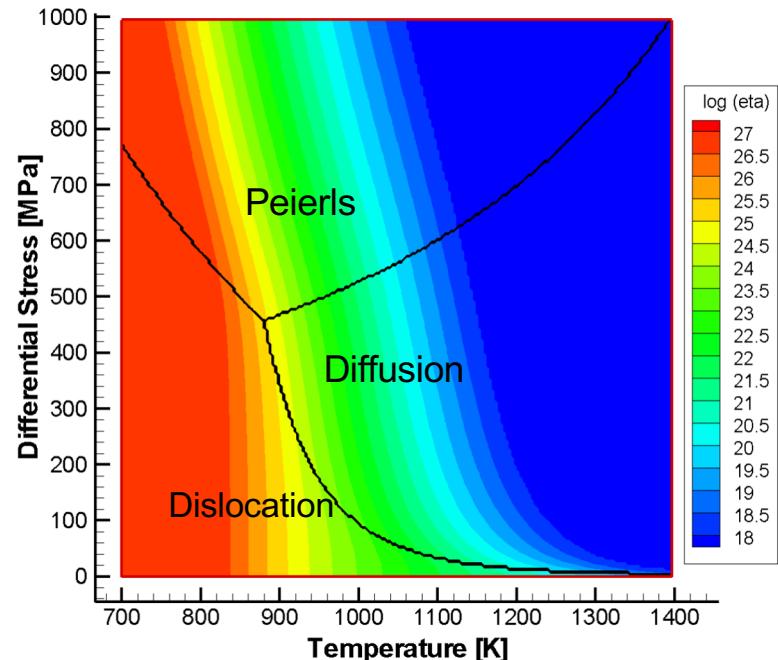
$$\tau_Y = \mu P + c$$

Diffusion, Dislocation and Peierls creep

$$A_l = B_l \exp \left[-\frac{E_l + pV_l}{RT} \right],$$

$$A_n = B_n \exp \left[-\frac{E_n + pV_n}{RT} \right]$$

$$A_p = \frac{B_p}{(\gamma \tau_p)^s} \exp \left[-\frac{E_p + pV_p}{RT} (1-\gamma)^q \right]$$



Effective creep viscosities

$$\eta_l = \frac{1}{2} (A_l)^{-1}$$

$$\eta_n = \frac{1}{2} (A_n)^{-\frac{1}{n}} (\dot{\varepsilon}_{II}^*)^{\frac{1}{n}-1}$$

$$\eta_p = \frac{1}{2} (A_p)^{-\frac{1}{s}} (\dot{\varepsilon}_{II}^*)^{\frac{1}{s}-1}$$

Peierls effective exponent

$$s = \frac{E_p + PV_p}{RT} (1-\gamma)^{q-1} q \gamma$$

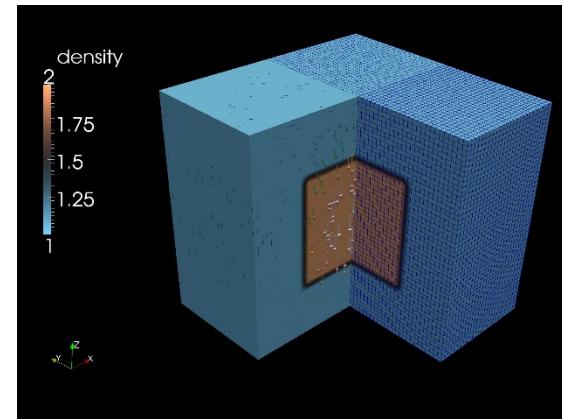
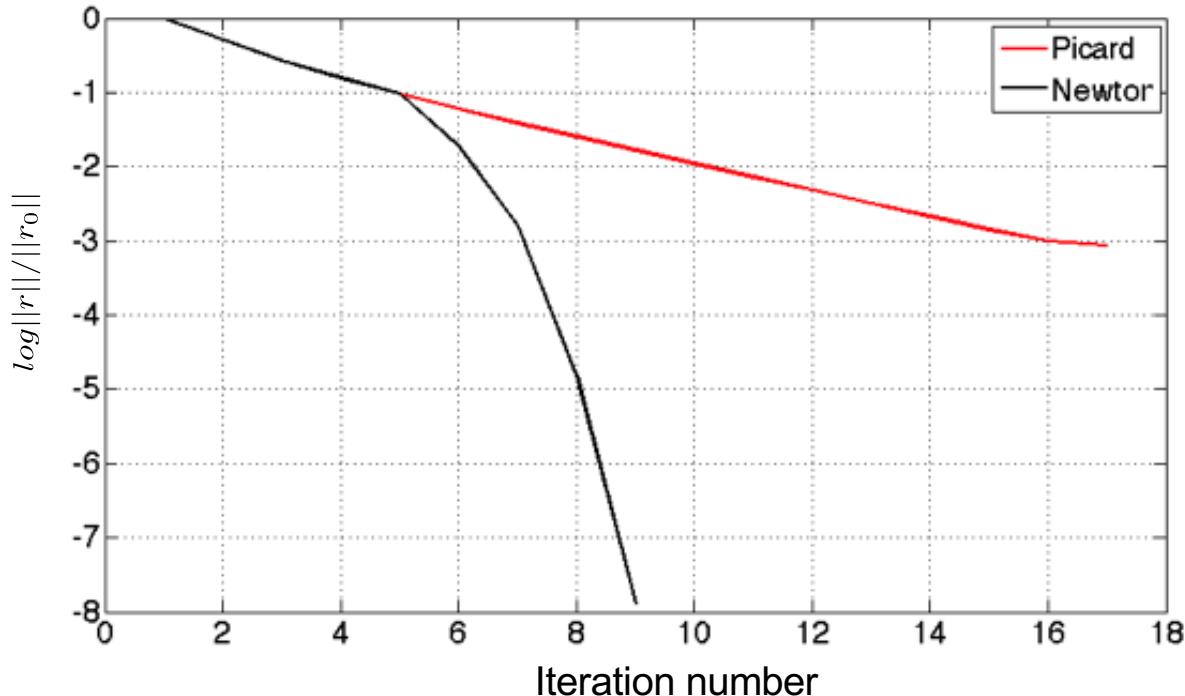
Picard vs. Newton

Quasi-linear residual form:

$$r(x) = A(x)x - b = 0$$

Picard fixed-point iteration:

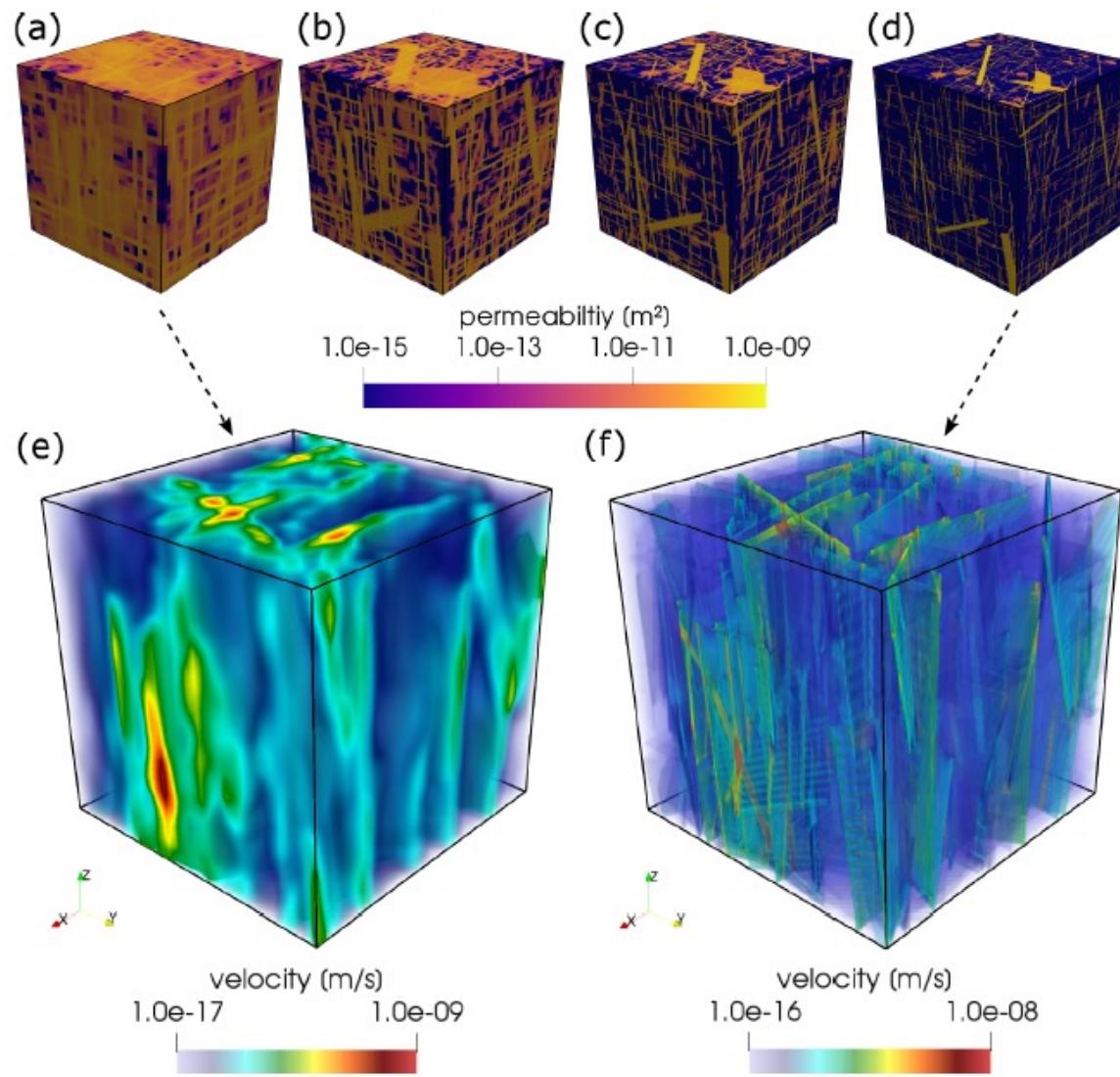
$$J(x) \approx A(x)$$



- Newton much faster than Picard for nonlinear materials
- Picard approximation facilitates convergence at the initial stages
- Switching to Picard can improve Newton convergence

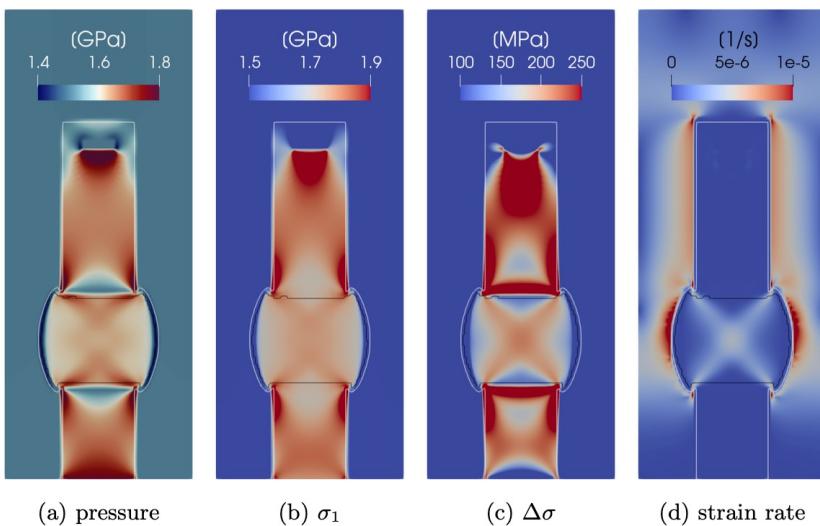
Some recent LaMEM applications

Fracture network permeability study

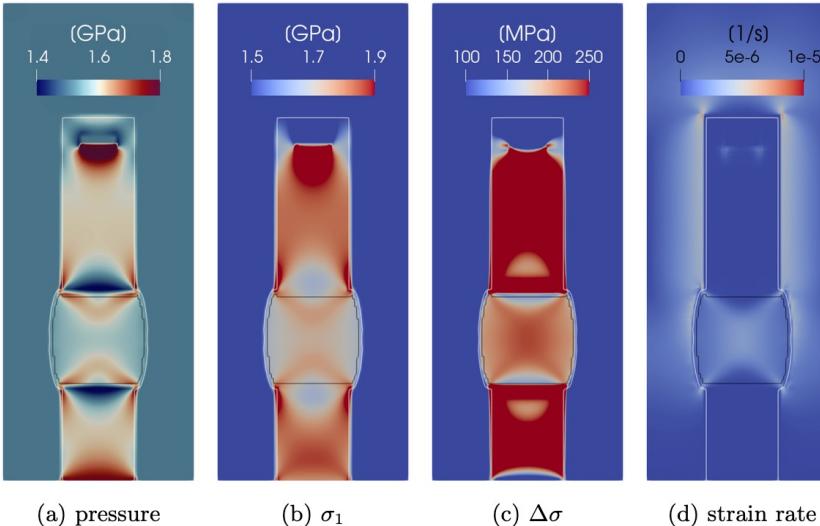


High-pressure rock deformation experiments

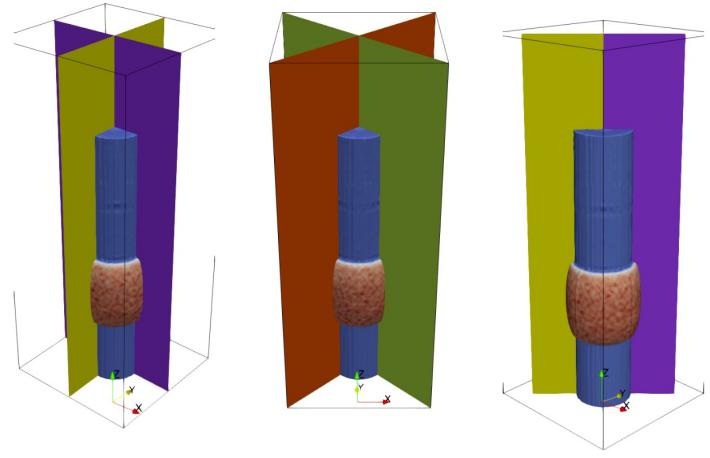
2D



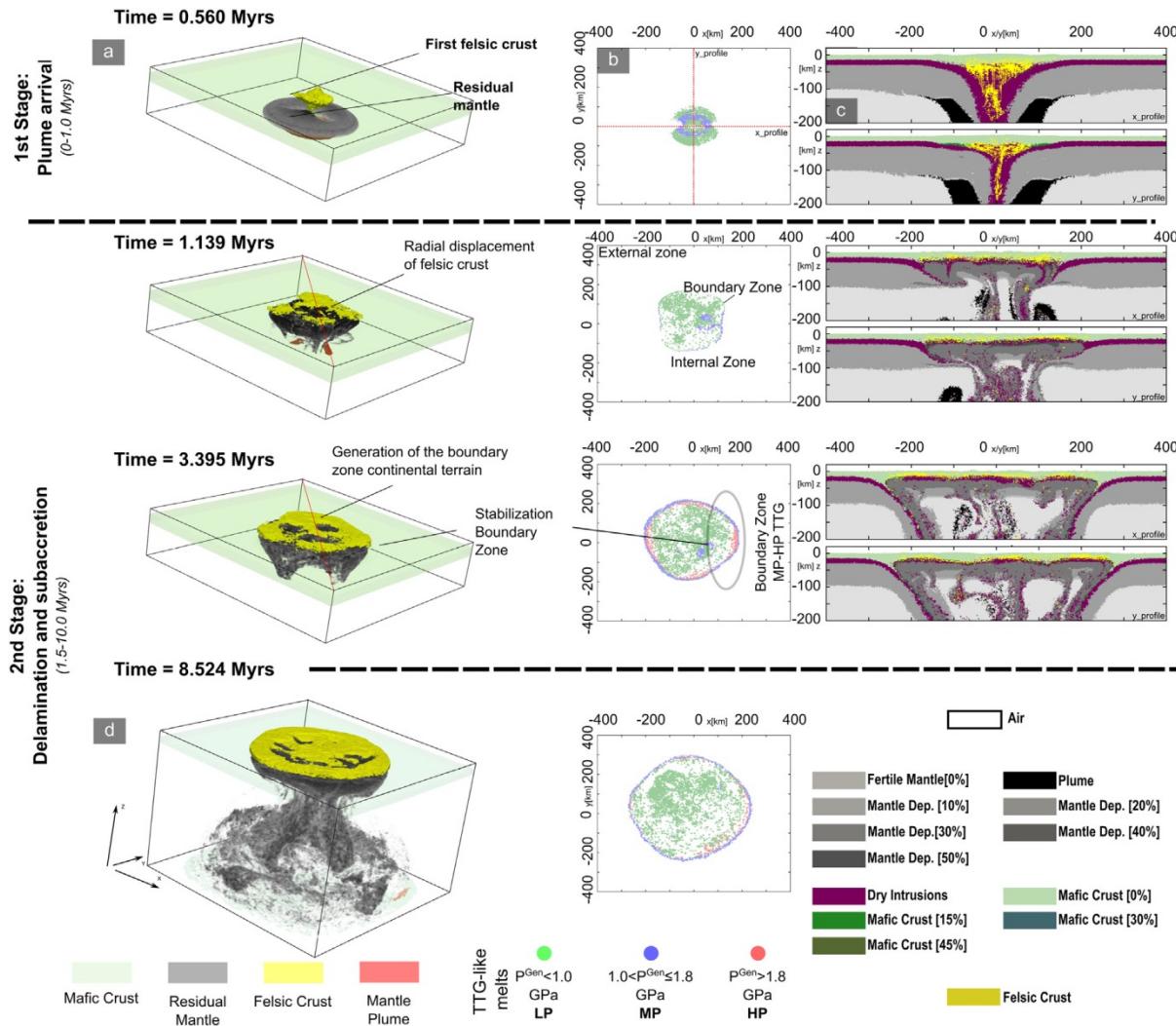
3D



Kilian Herrmanns
(BSc Thesis, Heidelberg)

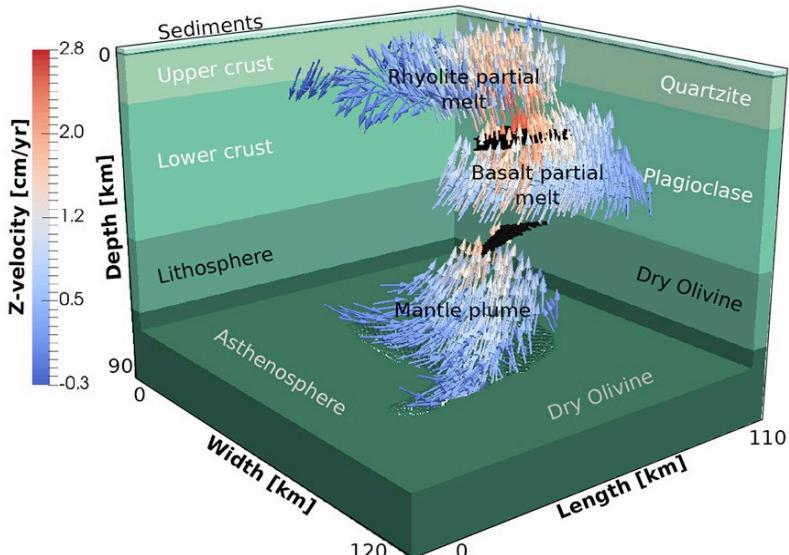
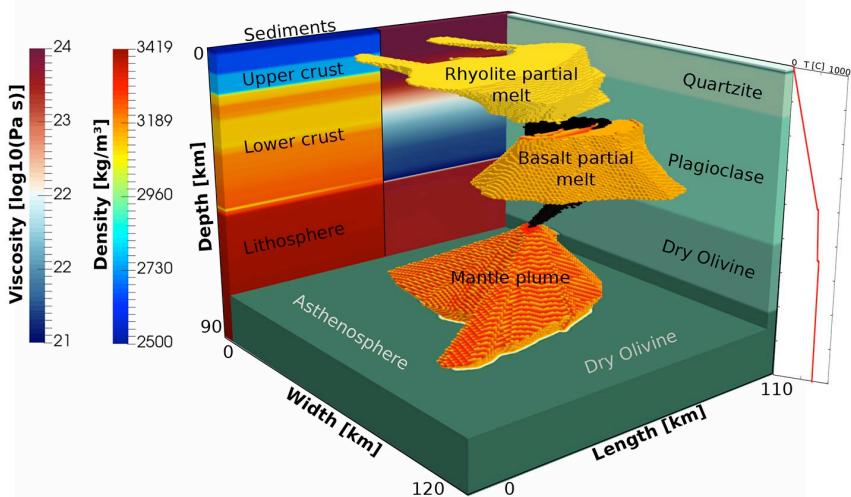


Plume - LID interaction during the Archean

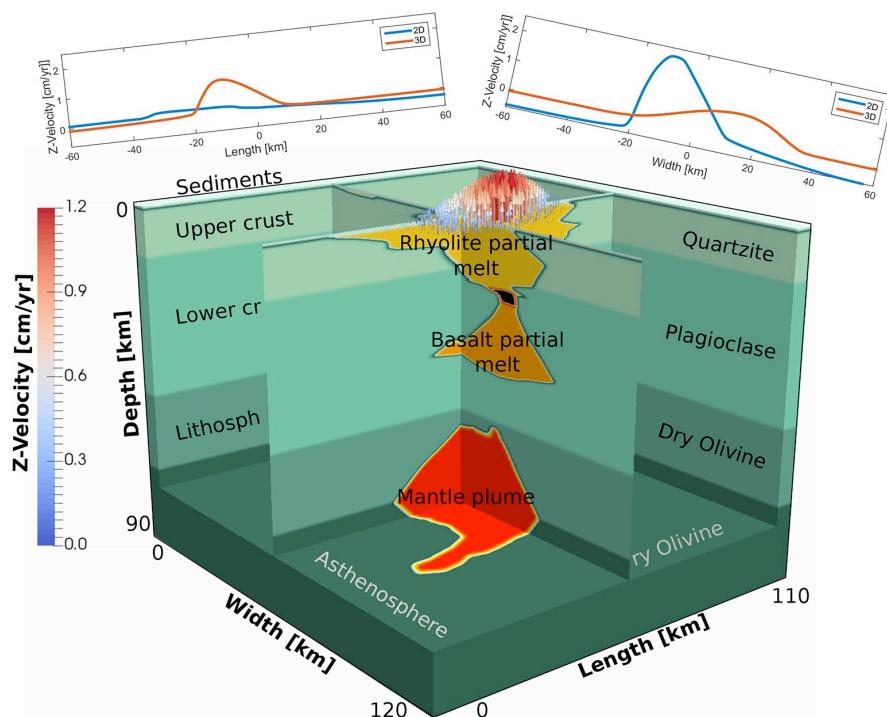


Piccolo et al., 2020 – Gondwana research

Yellowstone Magmatic System

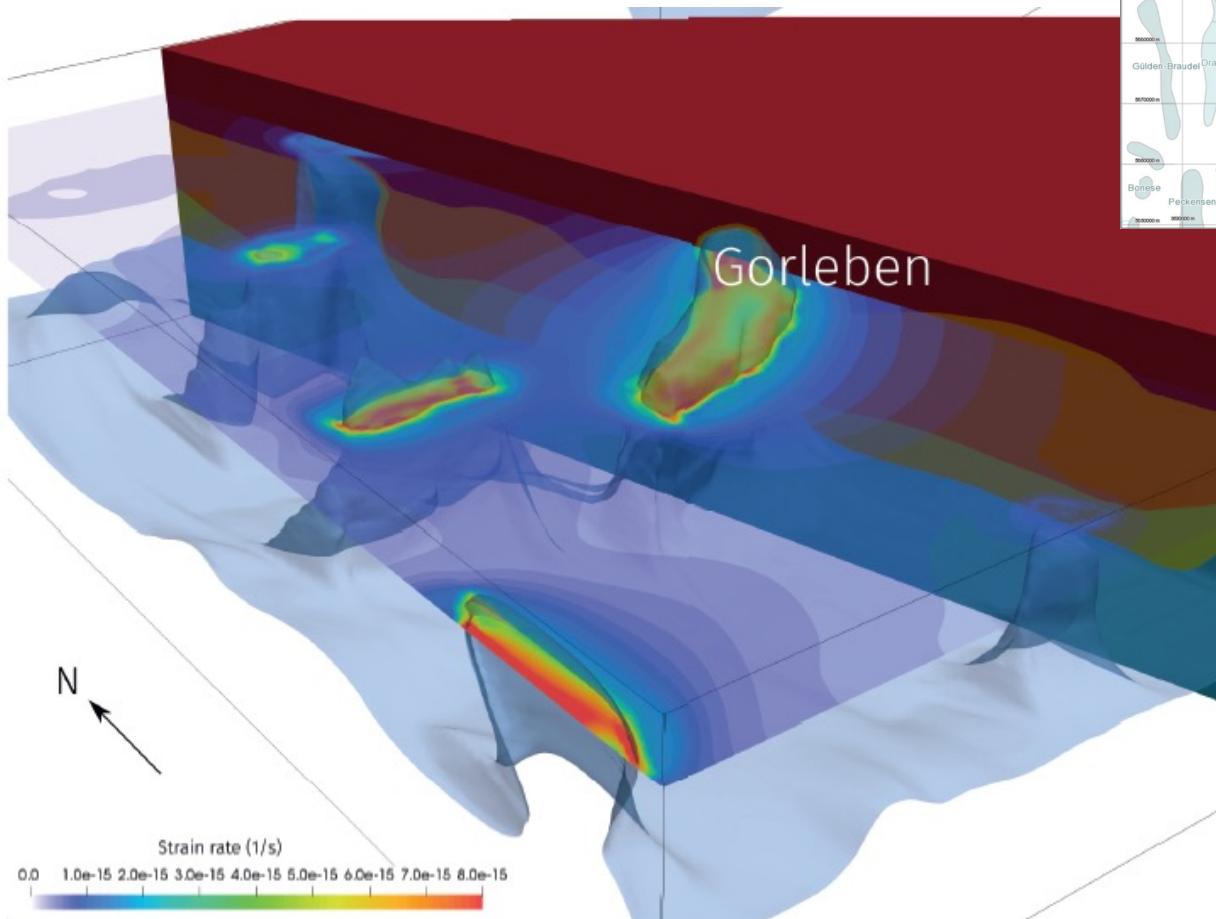


Reuber et al., 2018, Frontiers in Earth Sciences

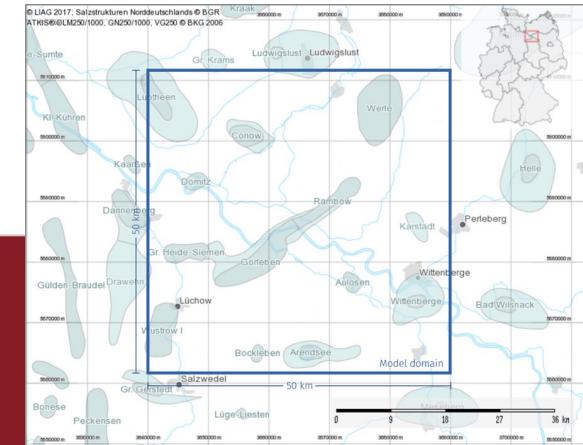


Gorleben salt dome

- Salt cavities as potential storage of radioactive waste?

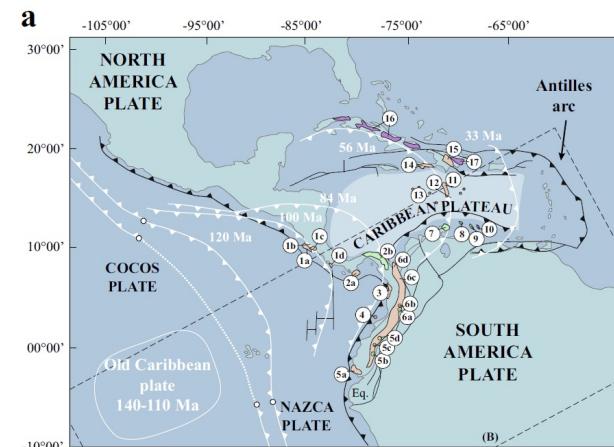
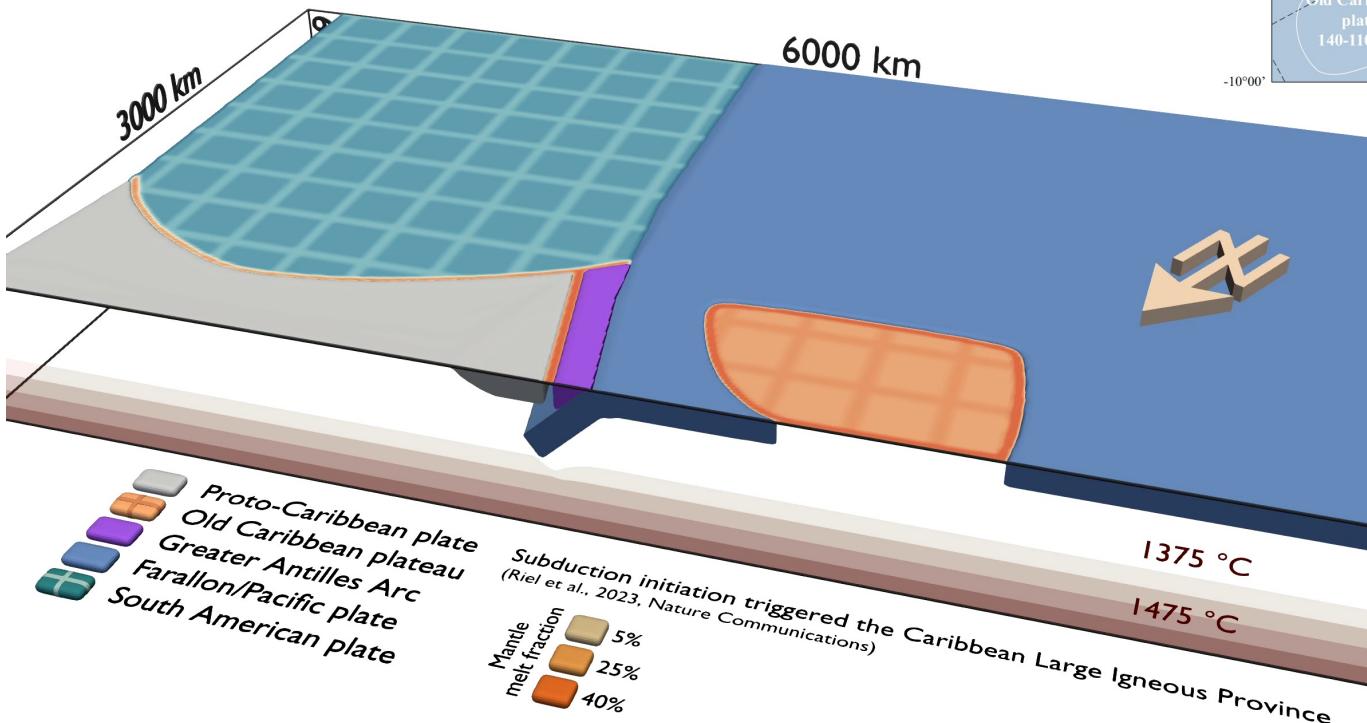


Baumann et al., 2018



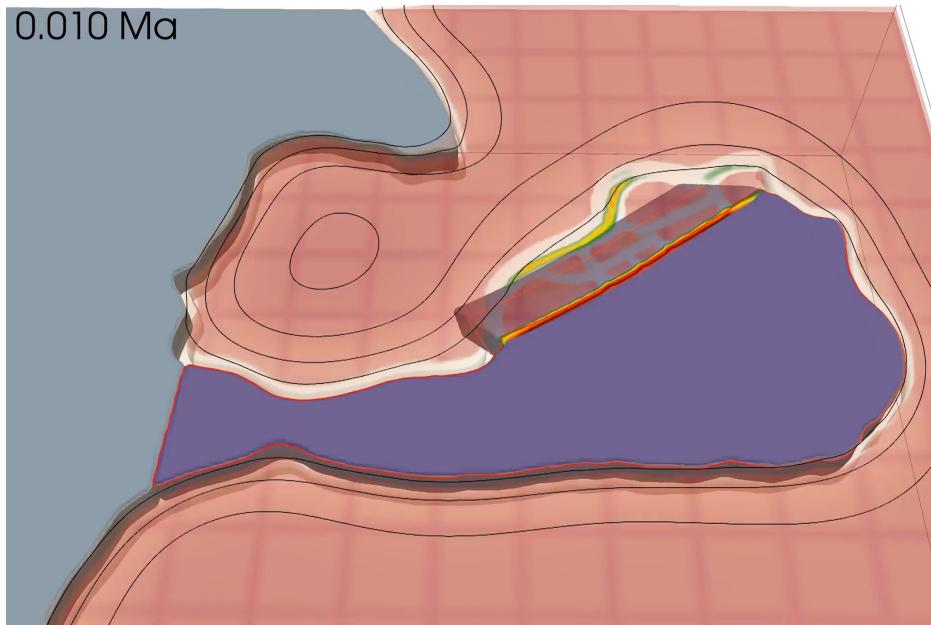
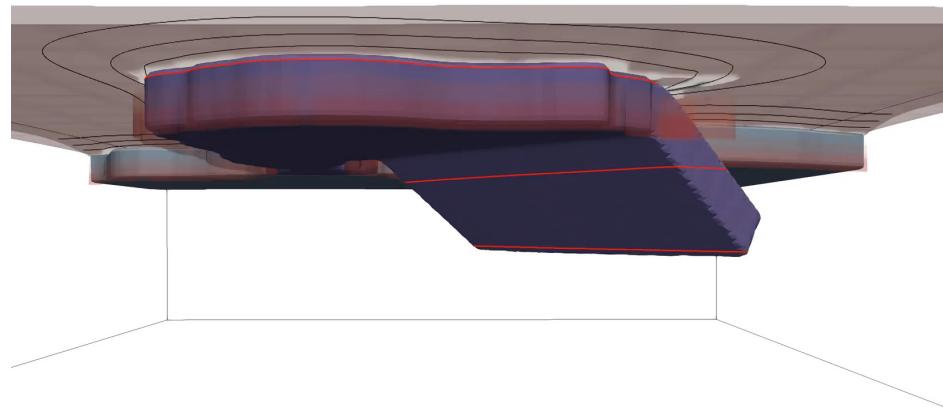
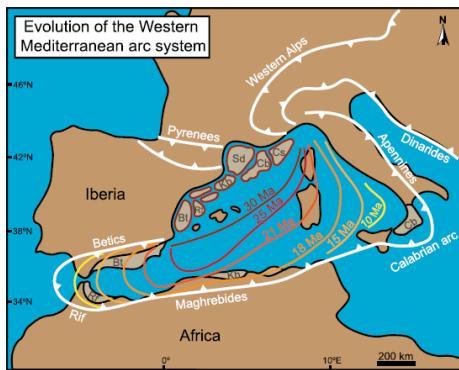
Formation of Caribbean LIP

0.01 Myrs



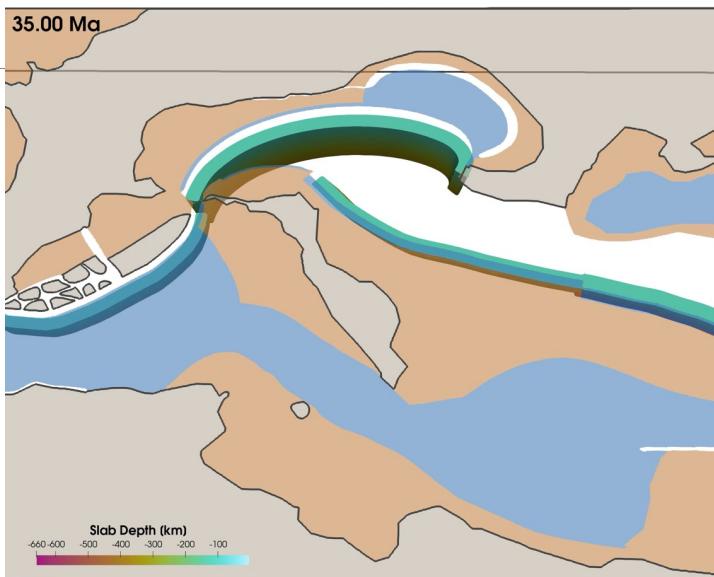
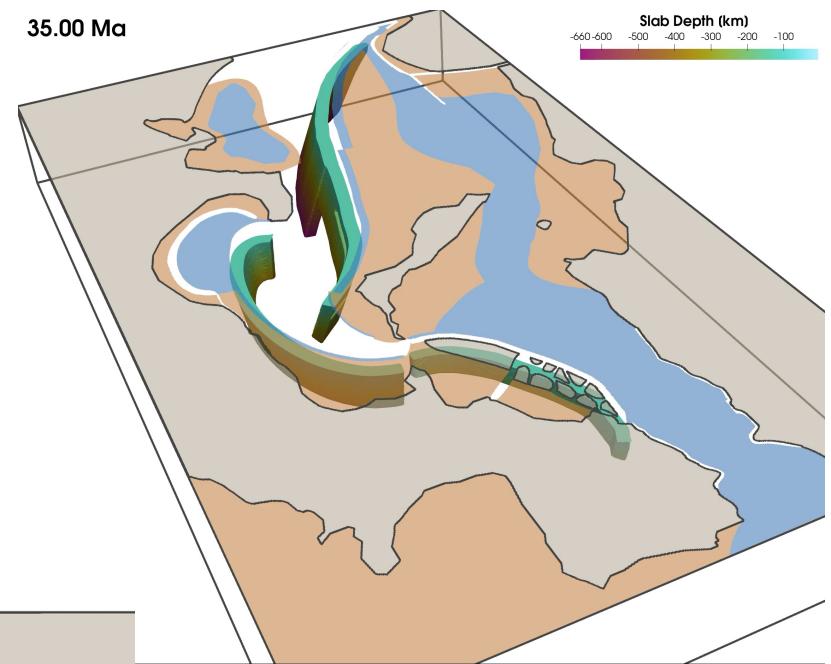
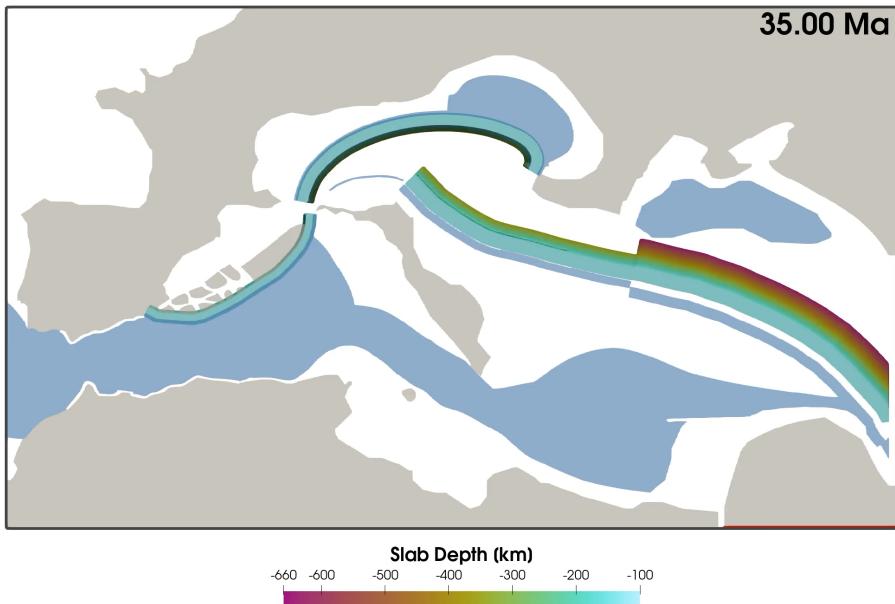
Subduction invasion - Gibraltar

0.010 Ma



Duarte, Riel et al., 2024 – Geology

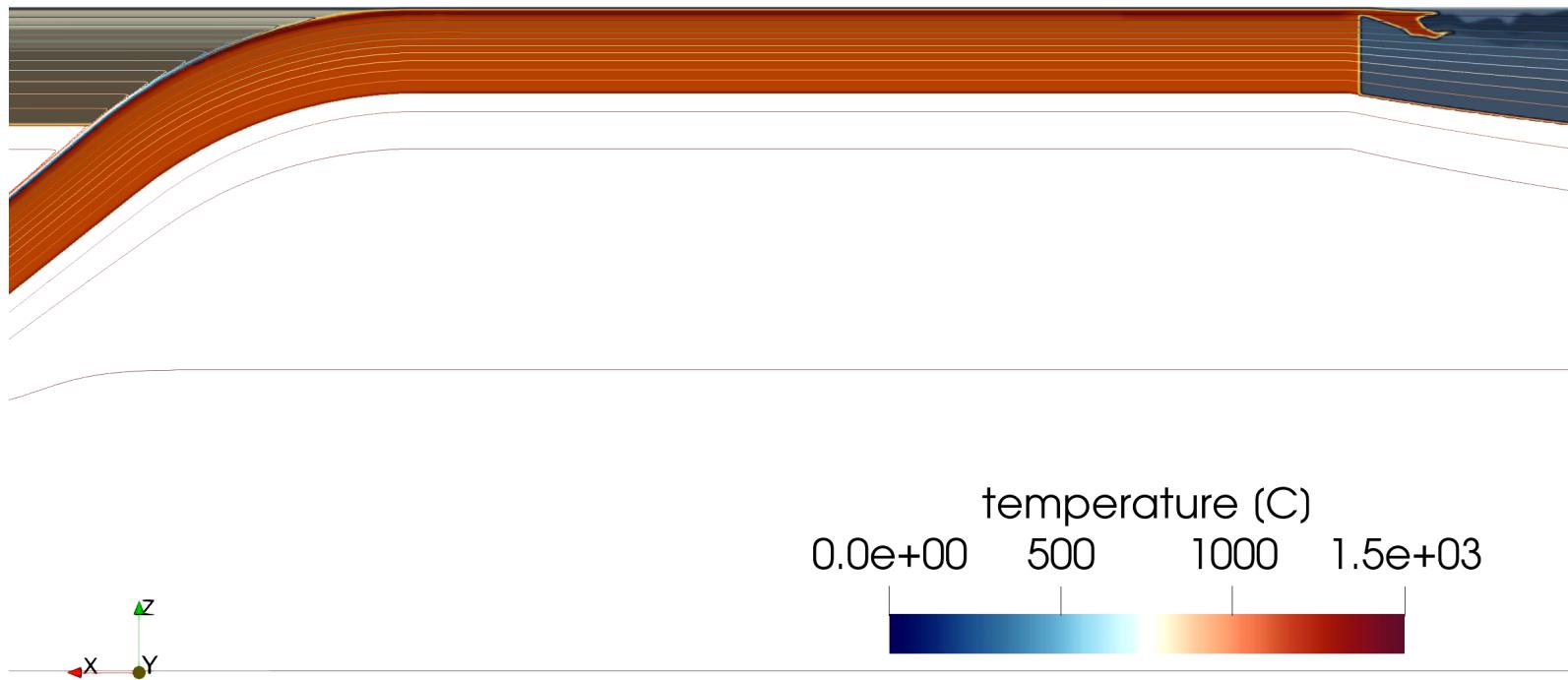
Mantle Dynamics in the Mediterranean and Plate Motion of the Adriatic Microplate



Schuler et al., 2025 G-cubed

Role of lower crust in Orogen build-up

Rodrigues et al, in prep



Ridge opening and mantle window

Rojas-Agramonte, to be submitted

