3D Transient Superstructures in Mantle Convection using Lattice Boltzmann Method

Gabriele Morra[1], Peter Mora[2], and David Yuen[3]

[1]University of Louisiana at Lafayette
[2]King Fahd University of Petroleum & Minerals
[3]Univ of Minnesota

November 24, 2022

Abstract

Recent advances in modeling Rayleigh-Benard convection have demonstrated the existence of turbulent superstructures, whose life and morphology largely varies with Rayleigh (Ra) and Prandtl (Pr) numbers. These structures appear as a two scale phenomena, where small scale rolls organize in larger convection cells, and can be modelled only in 3D on a simulation box characterized by a very large (>10) width/height (W/L) ratio, and sufficiently refined to resolve the boundary layer up to Ra = 10^8 (>100 divisions in height) and to Ra = 10^{10} (>200 divisions). To achieve this goal, we use our own 3D Parallel Python implementation of the Lattice Boltzmann Method, tested to run with linear efficiency on thousands of cores. We show the dependency of horizontal fluctuations of RMS of temperature and vertical velocity in the middle of the box and illustrate how the superstructures emerge for W/L ratios of Terrestrial Planets and Super Earths, and quantify the duration of these superstructures and the likely implications for the evolution of their surface features. The effect of the P/T dependent viscosity and thermal conductivity is finally discussed.

3D Transient Superstructures in Mantle Convection using Lattice Boltzmann Method

Gabriele Morra[(1)], Peter Mora[(2)], David A. Yuen[(3,4)]

(1) University of Louisiana at Lafayette (2) King Fahd University of Petroleum and Minerals (3) Columbia University (4) Ocean University of China

Introduction

Recent advances in modeling Rayleigh-Benard convection have demonstrated the existence of turbulent superstructures, whose life and morphology largely varies with Rayleigh (Ra) and Prandtl (Pr) numbers. These structures appear as a two scale phenomena, where small scale rolls organize in larger convection cells, and can be modelled only in 3D on a simulation box characterized by a very large (>10) width/height (W/L) ratio, and sufficiently refined to resolve the boundary layer up to Ra = 10^8 (>100 divisions in height) and to Ra = 10^{10} (>200 divisions). To achieve this goal, we use our own 3D Parallel Python implementation of the Lattice Boltzmann Method, tested to run with linear efficiency on thousands of cores.

The Lattice Boltzmann Method

The Boltzmann equilibrium distribution

$$f^{eq}(u) = \frac{\rho}{(2\pi RT)^{3/2}} \exp\left(-\frac{(\mathbf{c}-\mathbf{u})^2}{2RT}\right) \qquad \Delta f_\alpha^C = \frac{1}{\tau_f}\left(f_\alpha^{eq} - f_\alpha\right)$$

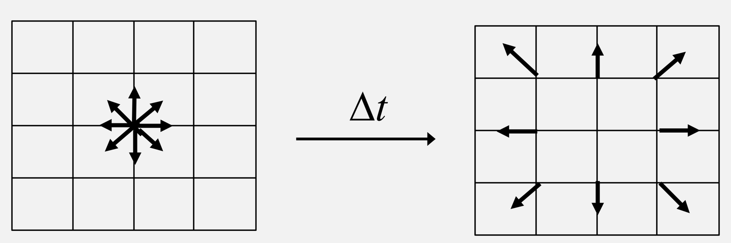
Taylor's expansion around $u \to 0$

$$f^{eq}(u) = \frac{\rho}{(2\pi RT)^{D/2}} \exp\left(-\frac{\mathbf{c}^2}{2RT}\right)\left(1 + \frac{\mathbf{c}\cdot\mathbf{u}}{RT} + \frac{1}{2}\frac{(\mathbf{c}\cdot\mathbf{u})^2}{(RT)^2} - \frac{1}{2}\frac{\mathbf{u}^2}{RT}\right)$$

$$f_\alpha(\mathbf{x}+\Delta\mathbf{x}_\alpha, t+\Delta t) = f_\alpha(\mathbf{x},t)$$

Streaming step

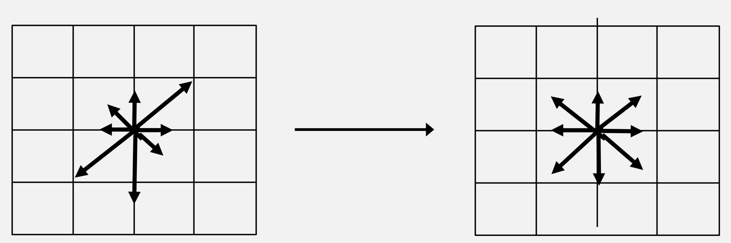
The streaming step movies particle densities



$$f_\alpha^*(\mathbf{x},t) \to f_\alpha(\mathbf{x},t) + \Delta f_\alpha^C(\mathbf{x},t)$$

Collision step

The collision step adjusts particle densities such that momentum is conserved



$$c_s = \sqrt{RT} = 1/\sqrt{3} \Rightarrow f_\alpha^{eq} = \rho w_\alpha\left(1 + 3(\mathbf{c}\cdot\mathbf{u}) + \frac{9}{2}(\mathbf{c}\cdot\mathbf{u})^2 - \frac{3}{2}\mathbf{u}^2\right)$$

$$\Delta f_\alpha^C = (f_\alpha^{eq} - f_\alpha)/\tau_f \quad where \quad \tau_f = \upsilon_f/(c_s^2\Delta t) + 0.5$$

Relaxation time *Kinematic viscosity* *Speed of sound*

$$f_\alpha(\mathbf{x}+\Delta\mathbf{x}_\alpha, t+\Delta t) = f_\alpha(\mathbf{x},t) + \Delta f_\alpha^C(\mathbf{x},t) + \Delta f_\alpha^B(\mathbf{x},t)$$

Collision term *Boussinesq term*

$$\Delta f_\alpha^B = -[w_\alpha 3(\mathbf{c}_\alpha)_z \rho\beta\Delta T g]/\tau_B$$

energy density

$$g_\alpha(\mathbf{x}+\Delta\mathbf{x}_\alpha, t+\Delta t) = g_\alpha(\mathbf{x},t) + \Delta g_\alpha^C(\mathbf{x},t)$$

Streaming *Collision*

$$\Delta g_\alpha^C = (g_\alpha^{eq} - g_\alpha)/\tau_g \qquad \tau_g = \upsilon_g/(c_s^2\Delta t) + 0.5 \quad where \quad \kappa = \upsilon_g$$

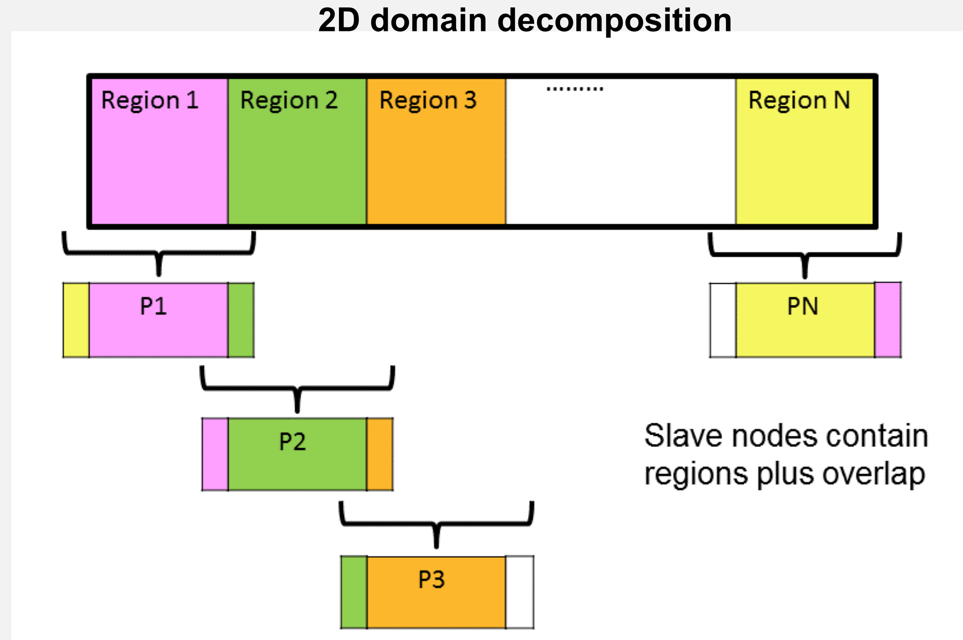
Relaxation time *Thermal diffusivity*

Equilibrium distribution

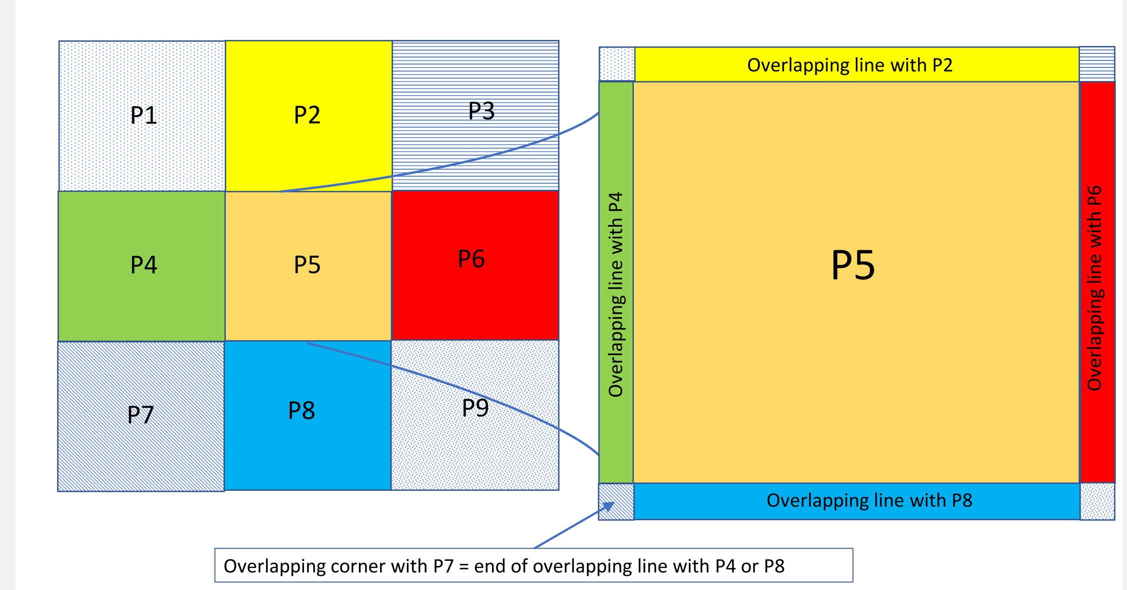
$$g_\alpha^{eq} = \rho\varepsilon w_\alpha \exp\left(1 + 3(\quad) + \frac{9}{2}(\mathbf{c}\cdot\mathbf{u})^2 - \frac{3}{2}\mathbf{u}^2\right) \qquad \rho\varepsilon = \rho DRT/2 = \sum_\alpha g_\alpha$$

Parallel Implementation

2D domain decomposition



3D domain decomposition




```
def get_edges():
    if nr == 2:
        if rank == 1:
            f[0:na,0:nz,0        ] = f[0:na,0:nz,mx[1]]
            f[0:na,0:nz,mx[1]+1] = f[0:na,0:nz,   1 ]
    else:
        if rank > 0 and rank < nr:
            edgeR = np.zeros(nz)
            edgeL = np.zeros(nz)

            rr = rank+1 if rank<nr-1 else 1  #right block
            rl = rank-1 if rank>1 else nr-1  #left block

            for a in range(na):
                comm.Send([ (f[a,0:nz,mx[rank]]).copy(), MPI.FLOAT ] ,dest=rr)
                comm.Send([ (f[a,0:nz,1]).copy(), MPI.FLOAT ], dest=rl)

                comm.Recv(edgeR,source=rr); f[a,0:nz,mx[rank]+1] = edgeR[0:nz]
                comm.Recv(edgeL,source=rl); f[a,0:nz,0] = edgeL[0:nz]
```

Too Few cores: Periodic BC without MPI

Store edges to pass

Identifies last and first blocks for periodic BC

Send and receive right and left edges

Blocking Communication of numpy arrays does not reduce performance

Macroscopic properties

```
rho = 0; for a in range(na): rho += f[a]
eta = 0; for a in range(na): eta += g[a]
eta /= rho

Pi = np.zeros((D,nz,nx_i))
for d in range(D):
   for a in range(na): Pi[d] += (f[a] * c[a][d])
   u[d] = Pi[d]/rho
```

Equilibrium distributions

```
u2 = 0; for d in range(D): u2 += u[d]**2
for a in range(na):
   cu = np.zeros((nz,nx_i))
   for d in range(D): cu += c[a][d]*u[d]
   f_eq[a] = rho * w[a] * (c1 + c2*cu + c3*cu**2 + c4*u2)
   g_eq[a] = eta * f_eq[a]
```

Boussinesq term

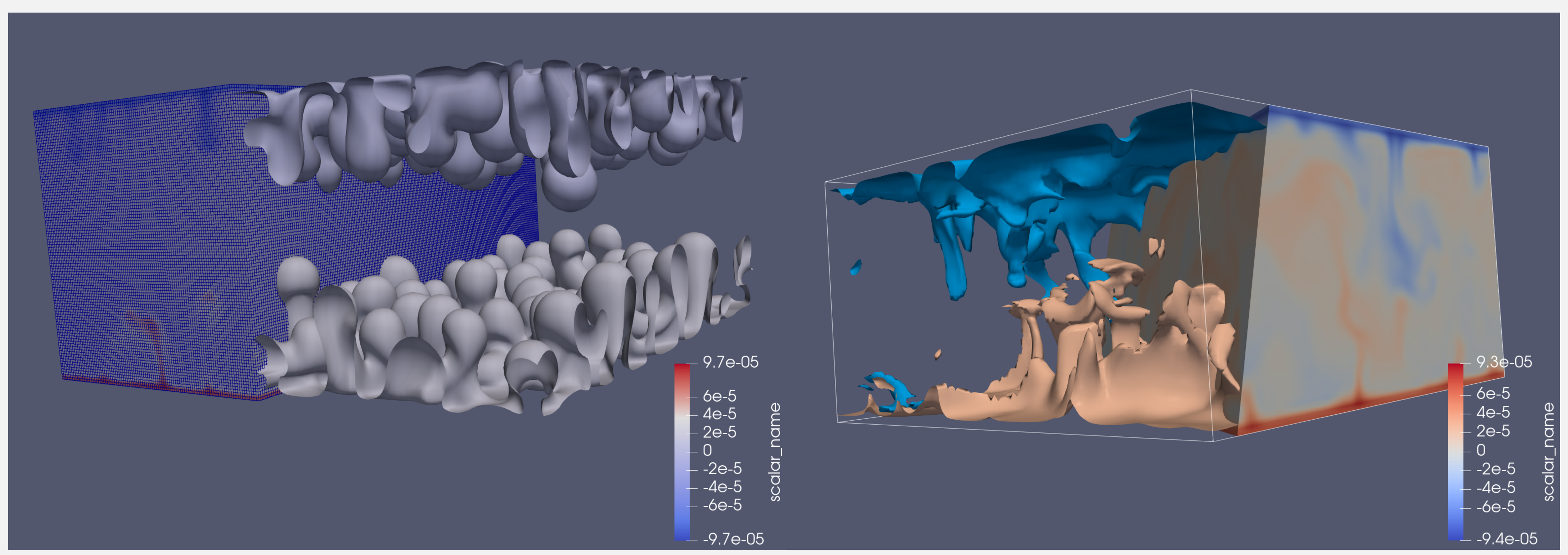
```
gi = -rho * beta * eta * G
for a in range(1,na): Delta_b[a] = (w[a] * c2 * c[a][1] * gi)
```

Collision terms

```
Delta_b /= tau_b # Collision term
Delta_f = (f_eq - f)/tau_f
Delta_g = (g_eq - g)/tau_g
f = f + Delta_f + Delta_b
g = g + Delta_g
```

Modern languages such as Python allow writing compact, easy to understand codes that can model exceptionally sophisticated physics. Here above, are the cores of two LBM implementations. On the left, the main MPI communication routine, which makes use of the mpi4py module. On the right is the core of the implementation of the Thermal Dependent Lattice Boltzmann Method used for modeling convection..

3D Simulations



Related References

H.R. Tufo, D.A. Yuen, **G. Morra**, M.G. Knepley, Bei Zhang张贝 and Shi Chen陈石. **2021**, Application of HPC and Big Data in Post-Pandemic Times, ***Earthquake Research Advances,***

G. Morra, E. Bozdag, M. Knepley, L. Räss, and V. Vesselinov, **2021**, A tectonic shift in analytics and computing is coming, ***EOS, Transactions of the American Geophysical Union***, 102,

P. Mora, **G. Morra**, D.A. Yuen, R. Juanes, **2021**, Influence of wetting on viscous fingering via 2D Lattice Boltzmann simulations, ***Transport in Porous Media,*** DOI 10.1007/s11242-021-01629-8

P. Mora, **G. Morra**, D.A. Yuen, **2021,** Optimized surface tension isotropy in the Rothman-Keller colour gradient Lattice Boltzmann Method for multi-phase flow, ***Physical Review E*,** 103, 033302

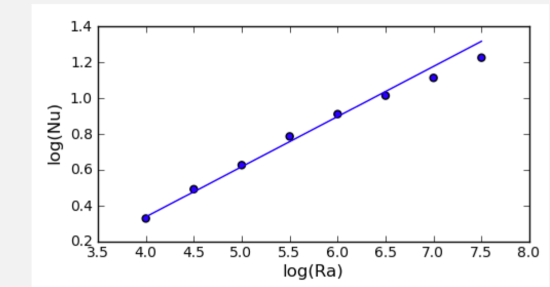
P. Mora, **G. Morra**, D.A. Yuen, R. Juanes, **2021**, Optimal wetting angles in Lattice Boltzmann simulations of viscous fingering, ***Transport in Porous Media,*** 136, no. 3: 831-842,

G. Morra, D.A. Yuen, H.R. Tufo, M.G. Knepley, **2020**, Fresh Outlook in Numerical Methods for Geodynamics. Part 1: Introduction and Modeling, ***Encyclopedia of Geology, 2e***. Pages 826-840

G. Morra, D.A. Yuen, H.R. Tufo, M.G. Knepley, **2020**, Fresh Outlook in Numerical Methods for Geodynamics. Part 2: Big Data, HPC, Education, ***Encyclopedia of Geology, 2e***. Pages 841-855

G. Morra. **2018**. Pythonic geodynamics. ***Lecture Notes in Earth System Sciences.*** Springer Verlag.

Mantle Convection



2D scaling (Mora and Yuen, 2018)

Existing GL theory predicts a complex relationship between Pr, Ra, Re and Nu numbers:

$$(Nu-1)RaPr^{-2} = c_1\frac{Re^2}{g(\sqrt{Re_L/Re})} + c_2 Re^3,$$

$$Nu - 1 = c_3 Re^{1/2}Pr^{1/2}\left\{f\left[\frac{2aNu}{\sqrt{Re_L}}g\left(\sqrt{\frac{Re_L}{Re}}\right)\right]\right\}^{1/2} + c_4 PrRef\left[\frac{2aNu}{\sqrt{Re_L}}g\left(\sqrt{\frac{Re_L}{Re}}\right)\right],$$

which at a first order, this implies a power law scaling:

Nu ~ Ra^γ

Pr = 1

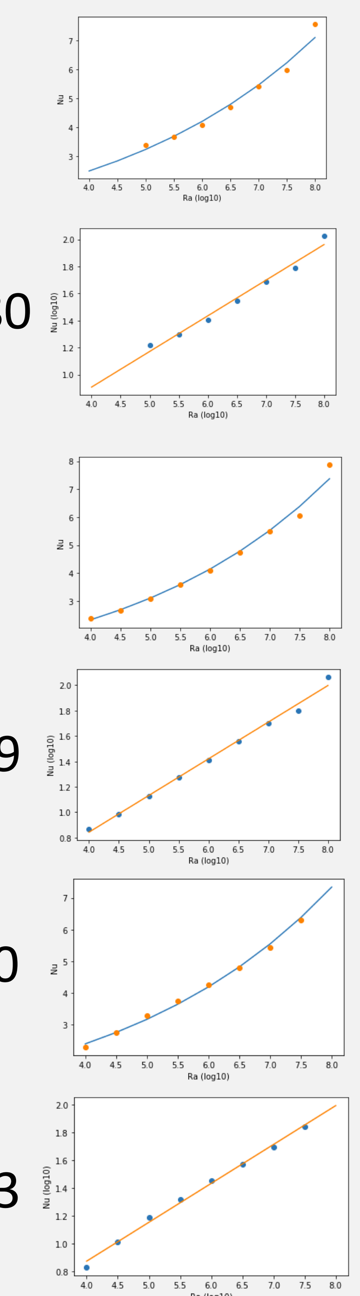
γ = 0.280

Pr = 10

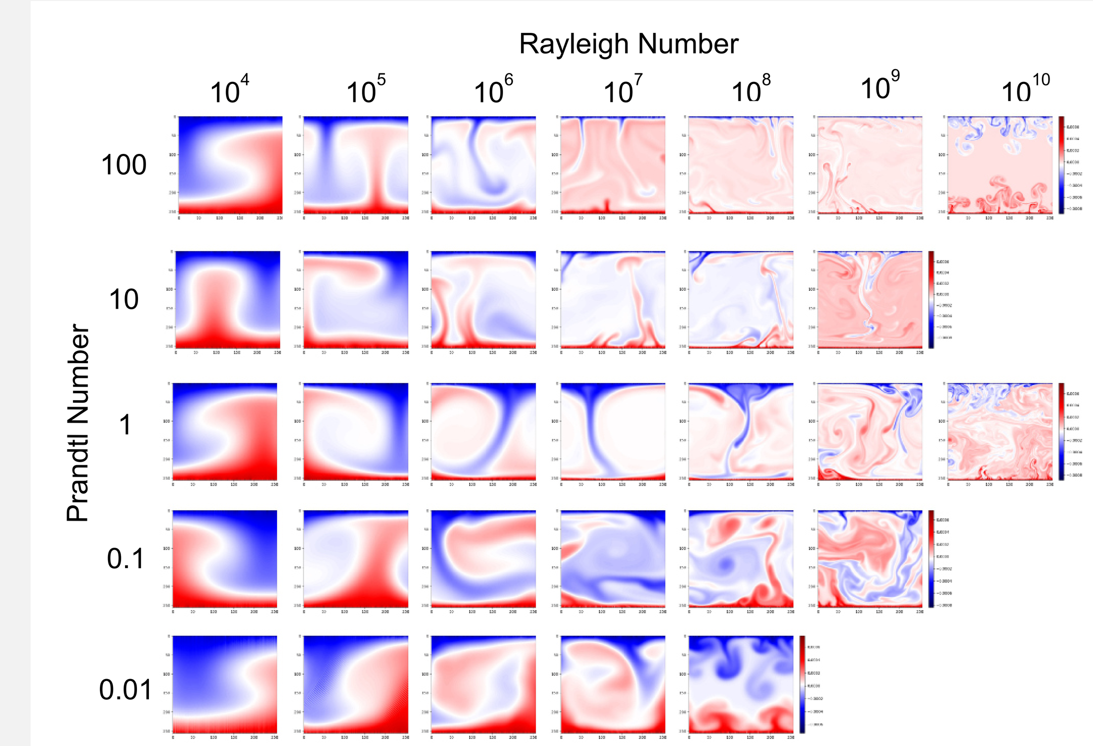
γ = 0.289

Pr = 100

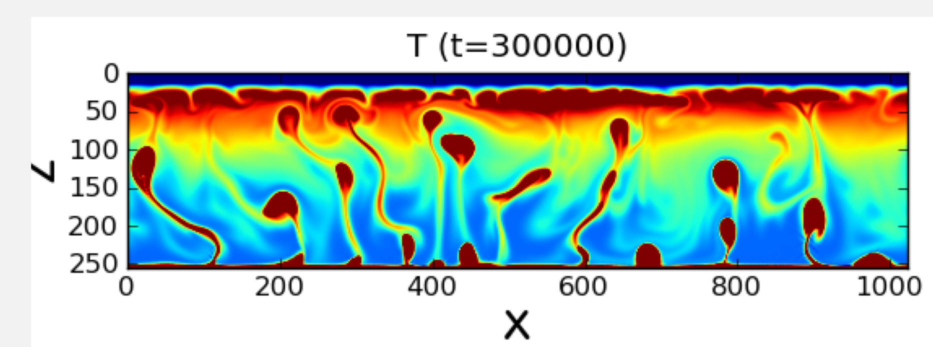
γ = 0.263



Wide spectrum of Ra and Pr



Non-linear rheologies



Conclusions and Future Work

CONCLUSIONS: Scaling between Nu and Ra follows the same power law from Pr = 1 up to Pr = 100. This confirms the prediction of the GL theory.

FUTURE WORK

1) Ongoing work will aims at increasing the horizontal vs vertical ratio up to 25. Pandey et al, (2018) has shown that macrostructures appear soon after the onset of convection. Scaling shows that these structure are long term for very high Pr such as Earth and Planetary mantle. We are running large scale simulations to reproduce them in the specific case of large Ra and Pr.

2) Use the ability to model low Pr numbers to model magma oceans convection and solidification. As shown in parallel publications, modeling surface tension allows this transition as well.