

# Bridging the Gap: Harnessing advanced Computational Libraries for coupled multi-physics problems

**Marc Spiegelman<sup>1,2</sup> and Cian R. Wilson<sup>1</sup>**

Plus enormous contributions from PETSc, FEniCS and AMCG groups

<sup>1</sup>Lamont-Doherty Earth Observatory, Columbia University, New York

<sup>2</sup>DEES/DAPAM, Columbia University, New York

BTG, Princeton, 1–2 October 2012

## Motivation

- Computation is essential for exploring non-linear multi-physics problems in Geosciences
- Major advances in hardware, software and algorithms make complex problems more accessible.
- However, still a considerable gap between Geosciences and Computational Sciences/Math.
- Some Existing Barriers:
  - Overall Complexity of both software libraries and models
  - Lack of transparency/reproducibility/reusability of current models
  - Language/Training issues between computation and Earth Sciences

## Motivation

- Computation is essential for exploring non-linear multi-physics problems in Geosciences
- Major advances in hardware, software and algorithms make complex problems more accessible.
- However, still a considerable gap between Geosciences and Computational Sciences/Math.
- Some Existing Barriers:
  - Overall Complexity of both software libraries and models
  - Lack of transparency/reproducibility/reusability of current models
  - Language/Training issues between computation and Earth Sciences
- *We need better tools for exploring complex (and simple) models and making advanced computation accessible to more general users.*

# Target Application : multi-physics models of magmatic plate boundaries

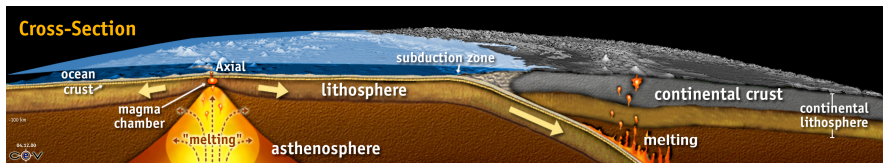


image courtesy of the Neptune project ([www.neptune.washington.edu](http://www.neptune.washington.edu))

- **Central to Solid Earth Geosciences:** Essential for understanding
  - Seismic, Volcanic and Tsunamagenic Natural Hazards
  - global tectonics and geochemical cycling

# Target Application : multi-physics models of magmatic plate boundaries

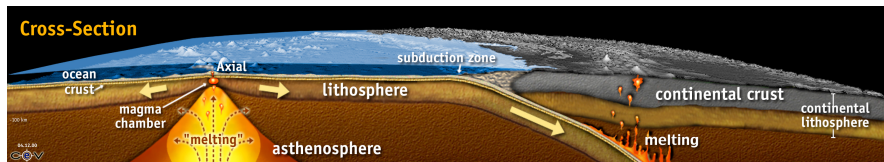


image courtesy of the Neptune project ([www.neptune.washington.edu](http://www.neptune.washington.edu))

- **Central to Solid Earth Geosciences:** Essential for understanding
  - Seismic, Volcanic and Tsunamagenic Natural Hazards
  - global tectonics and geochemical cycling
- **Multi-physics:** Tightly coupled non-linear systems
  - Coupled fluid-solid mechanics
  - Complex solid rheologies
  - Coupled thermodynamics/geodynamics

# Target Application : multi-physics models of magmatic plate boundaries

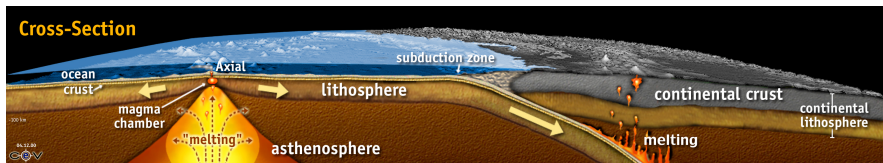


image courtesy of the Neptune project ([www.neptune.washington.edu](http://www.neptune.washington.edu))

- **Central to Solid Earth Geosciences:** Essential for understanding
  - Seismic, Volcanic and Tsunamagenic Natural Hazards
  - global tectonics and geochemical cycling
- **Multi-physics:** Tightly coupled non-linear systems
  - Coupled fluid-solid mechanics
  - Complex solid rheologies
  - Coupled thermodynamics/geodynamics
- **Considerable uncertainty** in equations, constitutive relations and coupling

## User Flexibility:

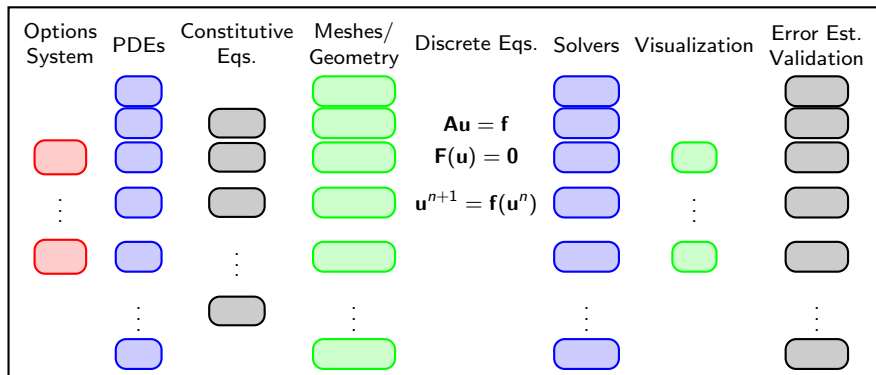
- Choice of **equations**, geometry, elements, constitutive relations, coupling
- Wide choice of solvers/preconditioners for **coupled non-linear** problems
- Ability to rapidly compose a range of models from simple process models to regional geodynamic models
- All choices available at or near run time

## Infrastructure:

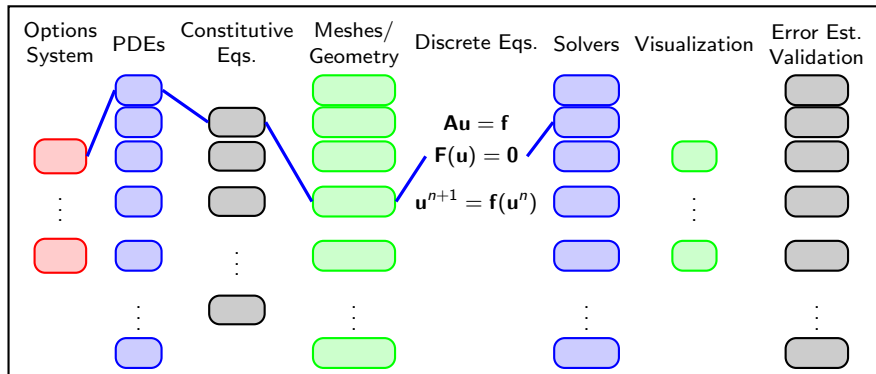
- Residual monitoring for the *full coupled problem*
- Model reproducibility:
  - Transparent options system
  - Regression tested
- Generic model services:
  - Standardized I/O
  - Checkpointing
  - Monitoring and detectors
- Parallel, scalable, open source, version controlled, regression tested... free



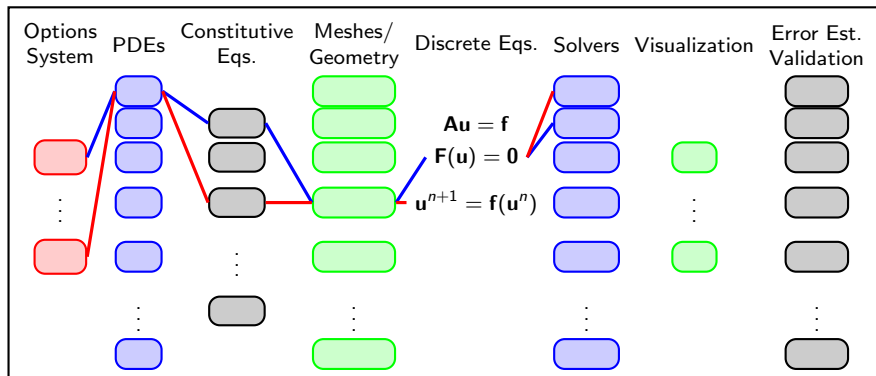
# Generic Structure of PDE Based Numerical Models



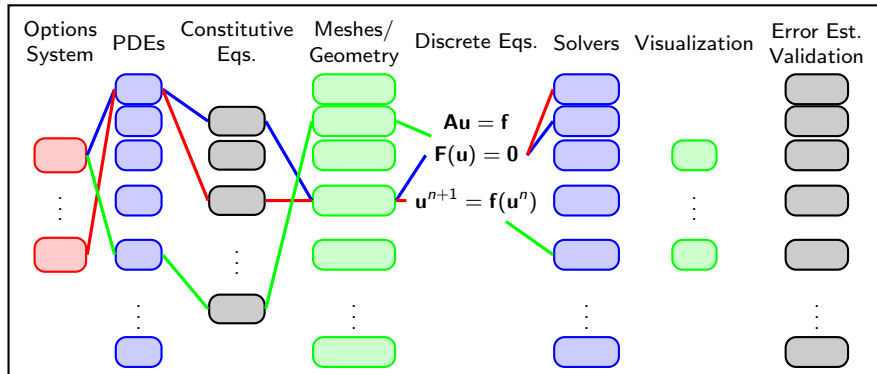
# Generic Structure of PDE Based Numerical Models



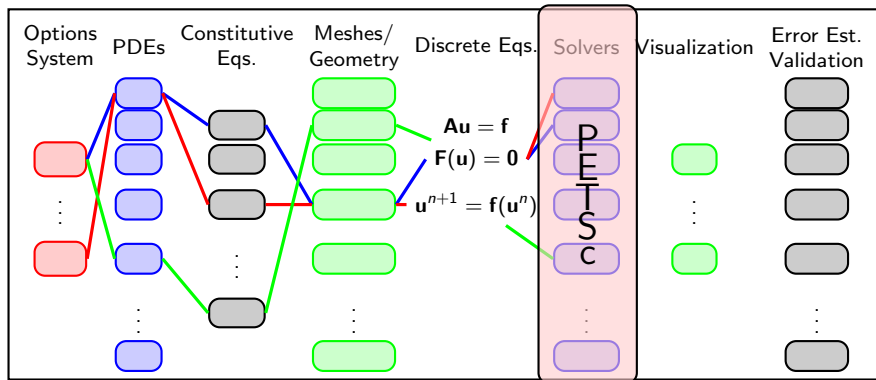
# Generic Structure of PDE Based Numerical Models



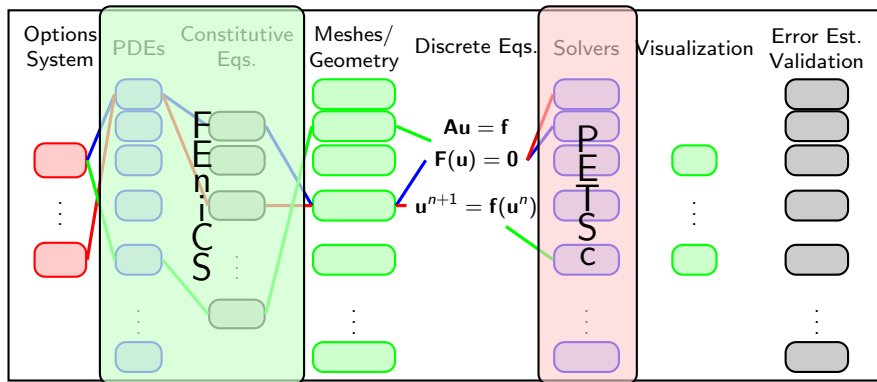
# Generic Structure of PDE Based Numerical Models



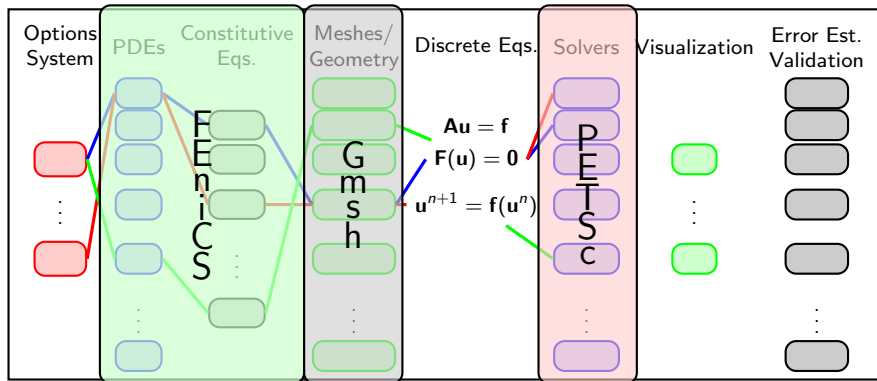
# Generic Structure of PDE Based Numerical Models



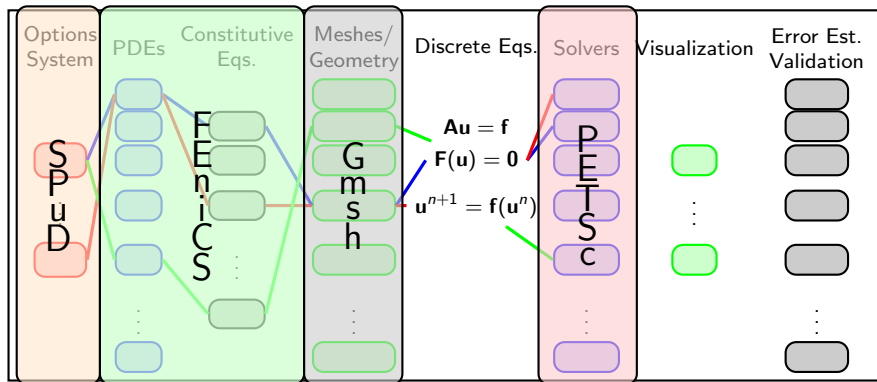
# Generic Structure of PDE Based Numerical Models



# Generic Structure of PDE Based Numerical Models

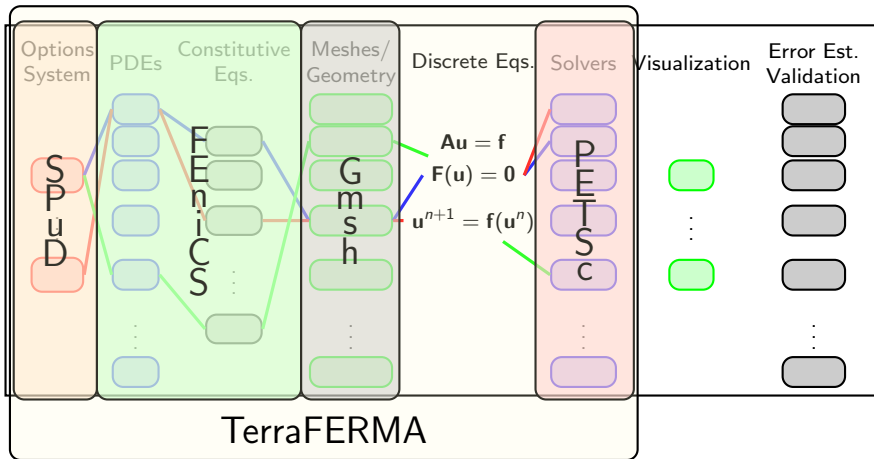


# Generic Structure of PDE Based Numerical Models



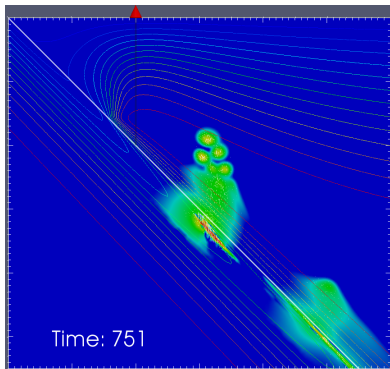


# Generic Structure of PDE Based Numerical Models



Transparent Finite Element Rapid Model Assembler

# Application: Fluid Migration in Subduction Zones



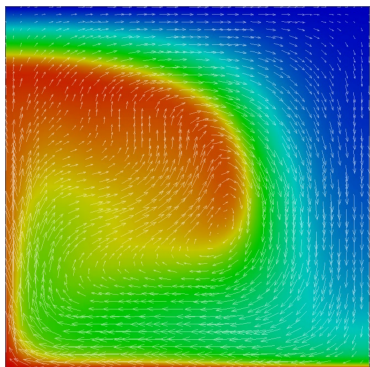
$$\nabla \cdot \left( 2\eta \left( \frac{\nabla \mathbf{v}_s + \nabla \mathbf{v}_s^T}{2} \right) \right) = \nabla p^* + \varphi_0 \varphi \mathbf{k}$$

$$\nabla \cdot \mathbf{v}_s = \varphi_0 \frac{\mathcal{P}}{\zeta}$$

$$-\nabla \cdot \frac{K}{\mu} \nabla \mathcal{P} + \frac{\mathcal{P}}{\zeta} = -\nabla \cdot \left( \frac{K}{\mu} \mathbf{k} - \nabla p^* \right) + \Gamma \frac{\Delta \rho}{\rho_f \varphi_0}$$

$$\frac{D\varphi}{Dt} = (1 - \varphi_0 \varphi) \frac{\mathcal{P}}{\zeta} + \frac{\Gamma}{\varphi_0}$$

# Example: Infinite Prandtl Thermal Convection

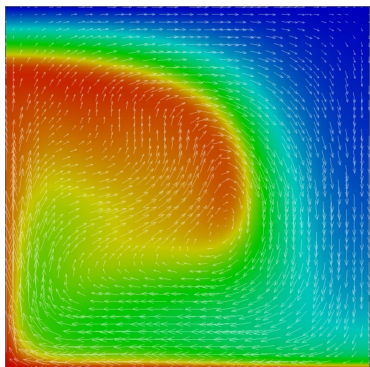


$$-\nabla \cdot \left[ 2 \mu \left( \frac{\nabla \mathbf{v} + \nabla \mathbf{v}^T}{2} \right) \right] + \nabla p - T k = 0,$$

$$\nabla \cdot \mathbf{v} = 0,$$

$$\frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T + \frac{1}{\text{Ra}} \nabla^2 T = 0$$

# Example: Infinite Prandtl Thermal Convection

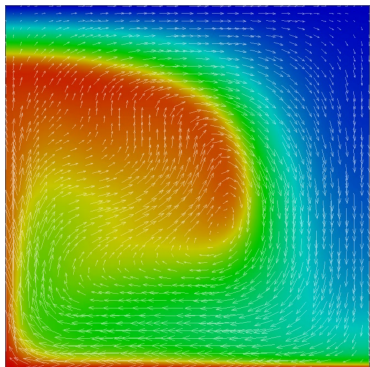


$$-\nabla \cdot \left[ 2 \mu \left( \frac{\nabla \mathbf{v} + \nabla \mathbf{v}^T}{2} \right) \right] + \nabla p - T k = 0,$$

$$\nabla \cdot \mathbf{v} = 0,$$

$$\frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T + \frac{1}{\text{Ra}} \nabla^2 T = 0$$

# Example: Infinite Prandtl Thermal Convection

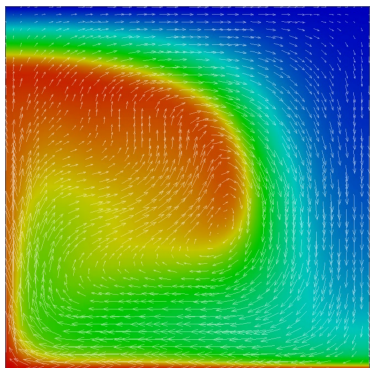


$$-\nabla \cdot \left[ 2\mu \left( \frac{\nabla \mathbf{v} + \nabla \mathbf{v}^T}{2} \right) \right] + \nabla p - T\mathbf{k} = 0,$$

$$\nabla \cdot \mathbf{v} = 0,$$

$$\frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T + \frac{1}{\text{Ra}} \nabla^2 T = 0$$

# Example: Infinite Prandtl Thermal Convection



$$-\nabla \cdot \left[ 2\mu \left( \frac{\nabla \mathbf{v} + \nabla \mathbf{v}^T}{2} \right) \right] + \nabla p - T\mathbf{k} = 0,$$

$$\nabla \cdot \mathbf{v} = 0,$$

$$\frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T + \frac{1}{\text{Ra}} \nabla^2 T = 0$$

Let  $u = (\mathbf{v}, p, T)$

$$u_{i+1} = u_i - \alpha J(u_i)^{-1} r(u_i)$$

## Example: FEniCS Equation Description

Given function  $u = (v, p, T)$ , test function  $u_t = (v_t, p_t, T_t)$  and trial function  $u_a = (v_a, p_a, T_a)$ :

Weak form of residual,  $r(u)$ :

$$r_V = \int_{\Omega} \left[ \left( \frac{\nabla v_t + \nabla v_t^T}{2} \right) : 2\mu \left( \frac{\nabla v + \nabla v^T}{2} \right) - \nabla \cdot v_t p - (v_t)_z T \right],$$

$$r_p = \int_{\Omega} p_t \nabla \cdot v,$$

$$r_T = \int_{\Omega} \left[ T_t ((T - T_n) + \Delta t v_{\theta} \cdot \nabla T_{\theta}) + \frac{\Delta t}{\text{Ra}} \nabla T_t \cdot \nabla T_{\theta} \right]$$

$$r = r_V + r_p + r_T$$

Weak form of Jacobian,  $J(u)$ :

$$J = r'(u)$$

## Example: FEniCS Equation Description

Given function  $u = (v, p, T)$ , test function  $u_t = (v_t, p_t, T_t)$  and trial function  $u_a = (v_a, p_a, T_a)$ :

Weak form of residual,  $r(u)$ :

```
r_v = (inner(sym(grad(v_t)), 2.*mu*sym(grad(v)))  
      - div(v_t)*p - T*v_t[1] )*dx  
r_p = p_t*div(v)*dx  
r_T = (T_t*((T - T_n) + dt*inner(v_theta, grad(T_theta)))  
      + (dt/Ra)*inner(grad(T_t), grad(T_theta)) )*dx  
  
r = r_v + r_p + r_T
```

Weak form of Jacobian,  $J(u)$ :

```
J = derivative(r, u, u_a)
```



## Example: PETSc Block Preconditioning & Solvers

$$J(u_i)\delta u = -r(u_i)$$

$$J = \begin{pmatrix} K_{11} & K_{12} & G_1 & C_1 \\ K_{21} & K_{22} & G_2 & C_2 \\ G_1^T & G_2^T & \mathbf{0} & \mathbf{0} \\ B_1 & B_2 & \mathbf{0} & A \end{pmatrix}$$

# Example: PETSc Block Preconditioning & Solvers

$$J(u_i)\delta u = -r(u_i)$$

$$J = \begin{pmatrix} K_{11} & K_{12} & G_1 & C_1 \\ K_{21} & K_{22} & G_2 & C_2 \\ G_1^T & G_2^T & \mathbf{0} & \mathbf{0} \\ B_1 & B_2 & \mathbf{0} & A \end{pmatrix}$$

Stokes

Advection-Diffusion

# Example: PETSc Block Preconditioning & Solvers

$$\tilde{J}_{\text{PC}}(u_i)^{-1} J(u_i) \delta u = -\tilde{J}_{\text{PC}}(u_i)^{-1} r(u_i)$$

$$J = \begin{pmatrix} K_{11} & K_{12} & G_1 & C_1 \\ K_{21} & K_{22} & G_2 & C_2 \\ G_1^T & G_2^T & \mathbf{0} & \mathbf{0} \\ B_1 & B_2 & \mathbf{0} & A \end{pmatrix}$$

$$J_{\text{PC}} = \begin{pmatrix} K_{11} & K_{12} & G_1 & C_1 \\ K_{21} & K_{22} & G_2 & C_2 \\ G_1^T & G_2^T & M & \mathbf{0} \\ B_1 & B_2 & \mathbf{0} & A \end{pmatrix}$$

# Example: PETSc Block Preconditioning & Solvers

$$\tilde{J}_{\text{PC}}(u_i)^{-1} J(u_i) \delta u = -\tilde{J}_{\text{PC}}(u_i)^{-1} r(u_i)$$

$$J = \begin{pmatrix} K_{11} & K_{12} & G_1 & C_1 \\ K_{21} & K_{22} & G_2 & C_2 \\ G_1^T & G_2^T & \mathbf{0} & \mathbf{0} \\ B_1 & B_2 & \mathbf{0} & A \end{pmatrix}$$

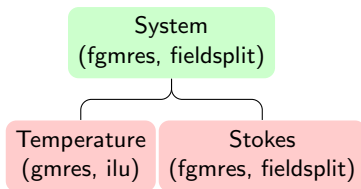
System  
(fgmres, fieldsplit)

$$J_{\text{PC}} = \begin{pmatrix} K_{11} & K_{12} & G_1 & C_1 \\ K_{21} & K_{22} & G_2 & C_2 \\ G_1^T & G_2^T & M & \mathbf{0} \\ B_1 & B_2 & \mathbf{0} & A \end{pmatrix}$$

# Example: PETSc Block Preconditioning & Solvers

$$\tilde{J}_{PC}(u_i)^{-1} J(u_i) \delta u = -\tilde{J}_{PC}(u_i)^{-1} r(u_i)$$

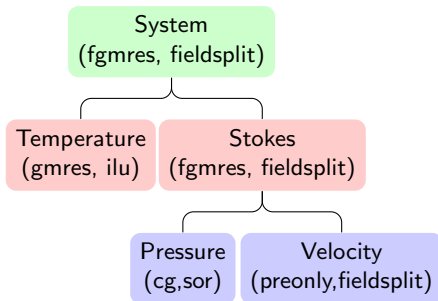
$$J = \begin{pmatrix} K_{11} & K_{12} & G_1 & C_1 \\ K_{21} & K_{22} & G_2 & C_2 \\ G_1^T & G_2^T & \mathbf{0} & \mathbf{0} \\ B_1 & B_2 & \mathbf{0} & A \end{pmatrix}$$
$$J_{PC} = \begin{pmatrix} K_{11} & K_{12} & G_1 & C_1 \\ K_{21} & K_{22} & G_2 & C_2 \\ G_1^T & G_2^T & M & \mathbf{0} \\ B_1 & B_2 & \mathbf{0} & A \end{pmatrix}$$



# Example: PETSc Block Preconditioning & Solvers

$$\tilde{J}_{PC}(u_i)^{-1} J(u_i) \delta u = -\tilde{J}_{PC}(u_i)^{-1} r(u_i)$$

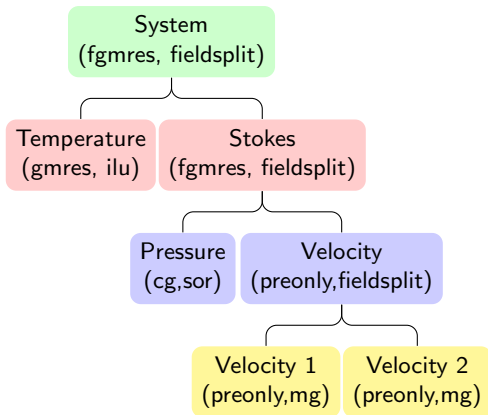
$$J = \begin{pmatrix} K_{11} & K_{12} & G_1 & C_1 \\ K_{21} & K_{22} & G_2 & C_2 \\ G_1^T & G_2^T & \mathbf{0} & \mathbf{0} \\ B_1 & B_2 & \mathbf{0} & A \end{pmatrix}$$
$$J_{PC} = \begin{pmatrix} K_{11} & K_{12} & G_1 & C_1 \\ K_{21} & K_{22} & G_2 & C_2 \\ G_1^T & G_2^T & M & \mathbf{0} \\ B_1 & B_2 & \mathbf{0} & A \end{pmatrix}$$



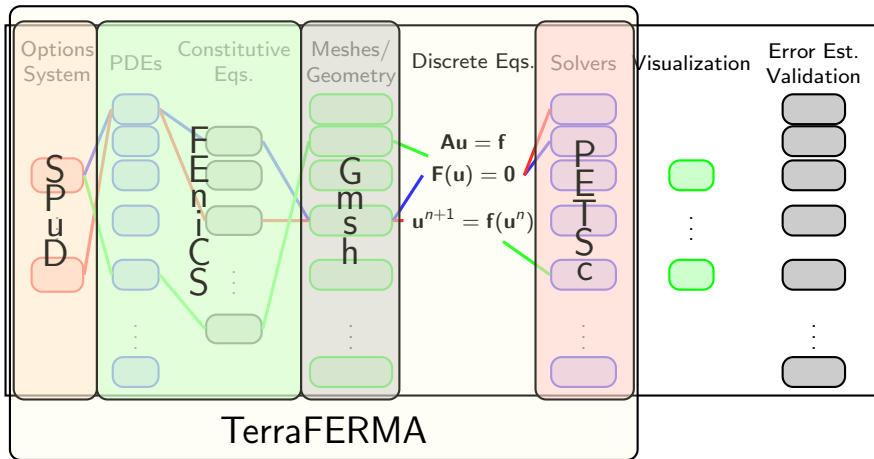
# Example: PETSc Block Preconditioning & Solvers

$$\tilde{J}_{PC}(u_i)^{-1} J(u_i) \delta u = -\tilde{J}_{PC}(u_i)^{-1} r(u_i)$$

$$J = \begin{pmatrix} K_{11} & K_{12} & G_1 & C_1 \\ K_{21} & K_{22} & G_2 & C_2 \\ G_1^T & G_2^T & \mathbf{0} & \mathbf{0} \\ B_1 & B_2 & \mathbf{0} & A \end{pmatrix}$$
$$J_{PC} = \begin{pmatrix} K_{11} & K_{12} & G_1 & C_1 \\ K_{21} & K_{22} & G_2 & C_2 \\ G_1^T & G_2^T & M & \mathbf{0} \\ B_1 & B_2 & \mathbf{0} & A \end{pmatrix}$$



# Generic Structure of PDE Based Numerical Models



Transparent Finite Element Rapid Model Assembler



# Example: TerraFERMA Options

File Edit View Validate Tools Help

- Node
- ▼ bucket\_options
    - ▶ geometry
    - ▶ io
    - ▶ timestepping
      - nonlinear\_systems
    - ▶ global\_parameters
    - ▶ system (Convection)
    - system

## Option Properties

### Description

Options describing a system.

A system consists of a DOLFIN functionspace, the fields on this functionspace, the forms describing the solvers and preconditioners that act on that functionspace and the coefficients that appear in those forms.

The system name must be unique amongst any other systems.

### Attributes

Name	Value
name	Convection

### Data

No data

### Comment

No comment

# Example: TerraFERMA Options

File Edit View Validate Tools Help

Node

- ▼ bucket\_options
  - ▶ geometry
  - ▶ io
  - ▶ timestepping
    - nonlinear\_systems
  - ▶ global\_parameters
  - ▼ system (Convection)
    - mesh (Mesh)
    - ufl\_symbol
    - ▶ field (Velocity)
    - ▶ field (Pressure)
    - ▶ field (Temperature)
    - field
    - ▶ coefficient (RayleighNumber)
    - coefficient
    - boundary\_condition
    - ▶ nonlinear\_solver (Preliminary)
    - ▶ nonlinear\_solver (Solver)
    - nonlinear\_solver
    - solve (in\_timestep)
    - system

## Option Properties

### Description

Options describing a system.

A system consists of a DOLFIN functionspace, the fields on this functionspace, the forms describing the solvers and preconditioners that act on that functionspace and the coefficients that appear in those forms.

The system name must be unique amongst any other systems.

### Attributes

Name	Value
name	Convection

### Data

No data

### Comment

No comment

/system:Convection

# Example: Setting Equations in TerraFERMA

File Edit View Validate Tools Help

Node

▼ bucket\_options

▶ geometry

▶ io

▶ timestepping

nonlinear\_systems

▶ global\_parameters

▼ system (Convection)

mesh (Mesh) ↓

ufl\_symbol

▶ field (Velocity)

▶ field (Pressure)

▶ field (Temperature)

field

▶ coefficient (RayleighNumber)

coefficient

boundary\_condition

▶ nonlinear\_solver (Preliminary)

▼ nonlinear\_solver (Solver)

▼ type (SNES) ↓

preamble

▶ form (Residual)

▶ form (Jacobian)

▶ form (JacobianPC)

quadrature\_degree

▶ snes\_type (ls) ↓

## Option Properties

### Description

ufl code form describing a linear residual form (must return a linear form). Any system, field or coefficient ufl symbols defined in this options file may be used in this form as well as any symbols defined in the preamble above.

### Data

```
1 b = 6.9077552789821368
2 mu = exp(-b*T_i)
3
4 rv = (inner(sym(grad(v_t)), 2.*mu*sym(grad(v_i))) \
5       - div(v_t)*p_i - T_i*v_t[1] )*dx
6 rp = p_t*div(v_i)*dx
7 rT = (T_t*((T_i - T_n) + dt*inner(v_theta, grad(T_theta))) \
8       + (dt/Ra)*inner(grad(T_t), grad(T_theta)) )*dx
9
10 r = rv + rp + rT
```

Revert data

Store data

### Comment

(string)

# Example: Setting Equations in TerraFERMA

File Edit View Validate Tools Help

Node

▼ bucket\_options

▶ geometry

▶ io

▶ timestepping

nonlinear\_systems

▶ global\_parameters

▼ system (Convection)

mesh (Mesh) ↓

ufl\_symbol

▶ field (Velocity)

▶ field (Pressure)

▶ field (Temperature)

field

▶ coefficient (RayleighNumber)

coefficient

boundary\_condition

▶ nonlinear\_solver (Preliminary)

▼ nonlinear\_solver (Solver)

▼ type (SNES) ↓

preamble

▶ form (Residual)

▶ form (Jacobian)

▶ form (JacobianPC)

quadrature\_degree

▶ snes\_type (ls) ↓

## Option Properties

### Description

ufl code form describing a jacobian bilinear form (must return a bilinear form). Any system, field or coefficient ufl symbols defined in this options file may be used in this form as well as any symbols defined in the preamble and residual linear form above.

### Data

```
1 a = derivative(r, u_i, u_a)
```

Revert data

Store data

### Comment

(string)

/system: Convection/nonlinear\_solver::Solver/type::SNES/form::Jacobian

# Example: Solver Options in TerraFERMA

File Edit View Validate Tools Help

Node

▼ linear\_solver

▶ iterative\_method (fgmres) ⚙

▼ preconditioner (fieldsplit) ⚙

composite\_type (multiplicative) ⚙

▼ fieldsplit (Temperature) ❌

▶ field (Temperature) ❌

field ⚙

▶ monitors

▼ linear\_solver

▶ iterative\_method (gmres) ⚙

preconditioner (ilu) ⚙

remove\_null\_space ⚙

▼ fieldsplit (Stokes) ❌

▶ field (Pressure) ❌

▶ field (Velocity) ❌

field ⚙

▶ monitors

▼ linear\_solver

▶ iterative\_method (fgmres) ⚙

▼ preconditioner (fieldsplit) ⚙

composite\_type (multiplicative) ⚙

▶ fieldsplit (Velocity) ❌

▶ fieldsplit (Pressure) ❌

fieldsplit ⚙

remove\_null\_space ⚙

fieldsplit ⚙

/system::Stokes/nonlinear\_solver::Solver/type::SNES/linear\_solver

## Option Properties

### Description

Options describing a linear solver.

### Data

No data

### Comment

No comment

# Example: Solver Options in TerraFERMA

File Edit View Validate Tools Help

- Node
- ▼ linear\_solver
    - ▶ iterative\_method (fgmres) ⚙
    - ▼ preconditioner (fieldsplit) ⚙
      - composite\_type (multiplicative) ⚙
    - ▼ fieldsplit (Temperature)
      - ▶ field (Temperature)
        - field
      - ▶ monitors
    - ▼ linear\_solver
      - ▶ iterative\_method (gmres) ⚙
      - preconditioner (ilu) ⚙
        - remove\_null\_space
    - ▼ fieldsplit (Stokes)
      - ▶ field (Pressure)
      - ▶ field (Velocity)
        - field
      - ▶ monitors
    - ▼ linear\_solver
      - ▶ iterative\_method (fgmres) ⚙
      - ▼ preconditioner (fieldsplit) ⚙
        - composite\_type (multiplicative) ⚙
        - ▶ fieldsplit (Velocity) ✖
        - ▶ fieldsplit (Pressure) ✖
          - fieldsplit ⚙
        - remove\_null\_space ⚙
        - fieldsplit ⚙

## Option Properties

### Description

Options describing a linear solver.

System  
(fgmres, fieldsplit)

### Comment

No comment

/system::Stokes/nonlinear\_solver::Solver/type::SNES/linear\_solver

# Example: Solver Options in TerraFERMA

File Edit View Validate Tools Help

Node

- ▼ linear\_solver
  - ▶ iterative\_method (fgmres)
  - ▼ preconditioner (fieldsplit)
    - composite\_type (multiplicative)
  - ▼ fieldsplit (Temperature)
    - ▶ field (Temperature)
      - field
    - ▶ monitors
  - ▼ linear\_solver
    - ▶ iterative\_method (gmres)
    - preconditioner (ilu)
    - remove\_null\_space
  - ▼ fieldsplit (Stokes)
    - ▶ field (Pressure)
    - ▶ field (Velocity)
      - field
    - ▶ monitors
  - ▼ linear\_solver
    - ▶ iterative\_method (fgmres)
    - ▼ preconditioner (fieldsplit)
      - composite\_type (multiplicative)
      - ▶ fieldsplit (Velocity)
      - ▶ fieldsplit (Pressure)
      - fieldsplit
      - remove\_null\_space
      - fieldsplit

## Option Properties

### Description

Options describing a linear solver.

System  
(fgmres, fieldsplit)

Temperature  
(gmres, ilu)

Stokes  
(fgmres, fieldsplit)

### Comment

No comment

/system::Stokes/nonlinear\_solver::Solver/type::SNES/linear\_solver

# Example: Solver Options in TerraFERMA

File Edit View Validate Tools Help

Node

▼ linear\_solver

▶ iterative\_method (fgmres)

▼ preconditioner (fieldsplit)

composite\_type (multiplicative)

▼ fieldsplit (Temperature)

▶ field (Temperature)

field

▶ monitors

▼ linear\_solver

▶ iterative\_method (gmres)

preconditioner (ilu)

remove\_null\_space

▼ fieldsplit (Stokes)

▶ field (Pressure)

▶ field (Velocity)

field

▶ monitors

▼ linear\_solver

▶ iterative\_method (fgmres)

▼ preconditioner (fieldsplit)

composite\_type (multiplicative)

▶ fieldsplit (Velocity)

▶ fieldsplit (Pressure)

fieldsplit

remove\_null\_space

fieldsplit

/system:Stokes/nonlinear\_solver::Solver/type::SNES/linear\_solver

Option Properties

Description

Options describing a linear solver.

System  
(fgmres, fieldsplit)

Temperature  
(gmres, ilu)

Stokes  
(fgmres, fieldsplit)

Pressure  
(cg,sor)

Velocity  
(preonly,fieldsplit)

Comment

No comment

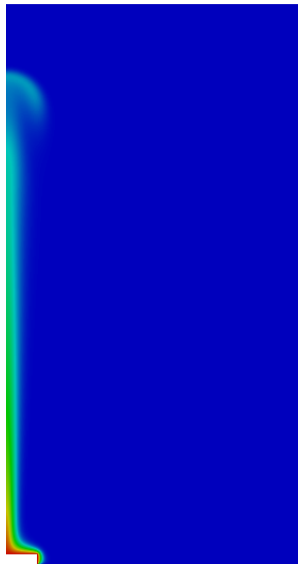


# Example: Benchmark (Blankenbach, 1989)

		Nu	$v_{rms}$	$q_1$	$q_2$	$T_e$	$z_e$
$Ra = 10^4$ isoviscous	32×32	4.887	42.865	8.060	0.589	0.422	0.226
	64×64	4.885	42.865	8.059	0.589	0.422	0.226
	128×128	4.885	42.865	8.059	0.589	0.422	0.225
	Benchmark	4.884	42.865	8.059	0.589	0.422	0.225
$Ra = 10^5$ isoviscous	32×32	10.539	193.222	19.081	0.722	0.428	0.114
	64×64	10.535	193.215	19.080	0.723	0.428	0.111
	128×128	10.534	193.215	19.080	0.723	0.428	0.112
	Benchmark	10.534	193.214	19.079	0.723	0.428	0.112
$Ra = 10^6$ isoviscous	32×32	21.982	834.024	46.008	0.877	0.432	0.059
	64×64	21.971	833.990	45.972	0.877	0.432	0.058
	128×128	21.972	833.989	45.967	0.877	0.432	0.058
	Benchmark	21.972	833.990	45.964	0.877	0.432	0.058
$Ra = 10^4$ $T$ -dep viscosity	32×32	10.069	479.951	17.533	1.007	0.739	0.062
	64×64	10.066	480.385	17.531	1.008	0.740	0.063
	128×128	10.064	480.257	17.528	1.008	0.740	0.063
	Benchmark	10.066	480.433	17.531	1.009	0.741	0.062

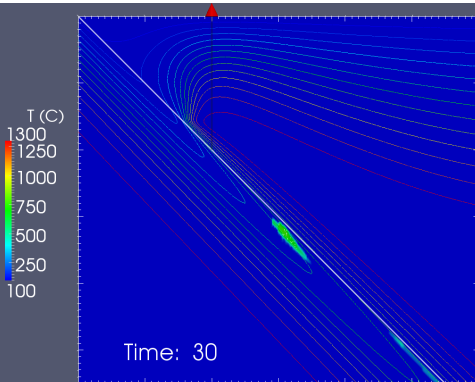
# Current Benchmarks

- Incompressible 2D Convection (Blankenbach et al, 1989)
- Incompressible 3D Convection (Busse et al., 1994)
- Cylindrical Laminar Plumes (Vatteville et al., 2009)
- Compressible 2D Convection (King et al., 2009)
- Kinematic Subduction Zones (van Keken et al., 2008)
- Linearized Free Surface Evolution (Kramer et al., 2012)
- Non-linear magma waves (Simpson and Spiegelman, 2011)
- Cylindrical Convection (Rhodri Davies:  
rhodri.davies@imperial.ac.uk)

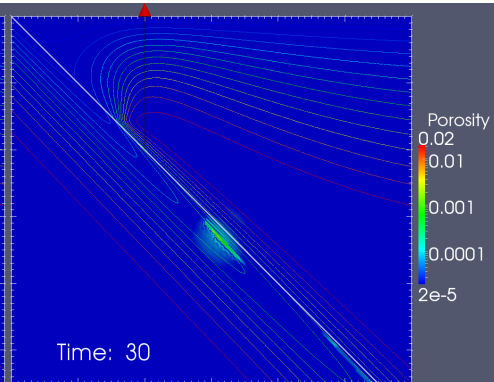


# Fluid Migration

Buoyancy



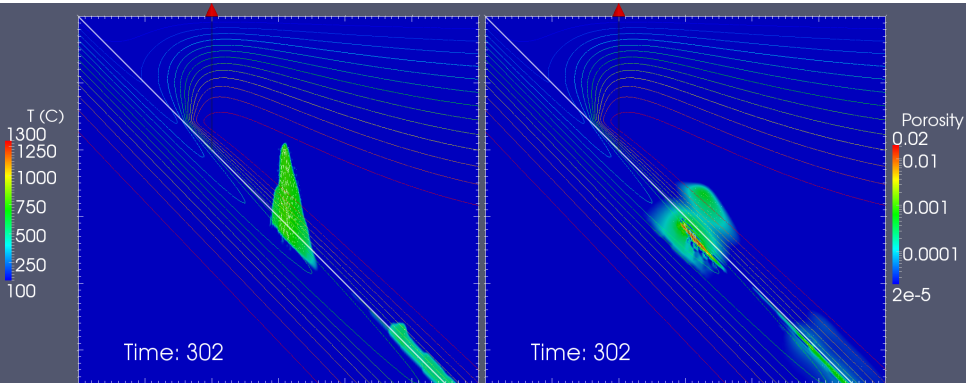
Buoyancy and Pore Pressure



# Fluid Migration

Buoyancy

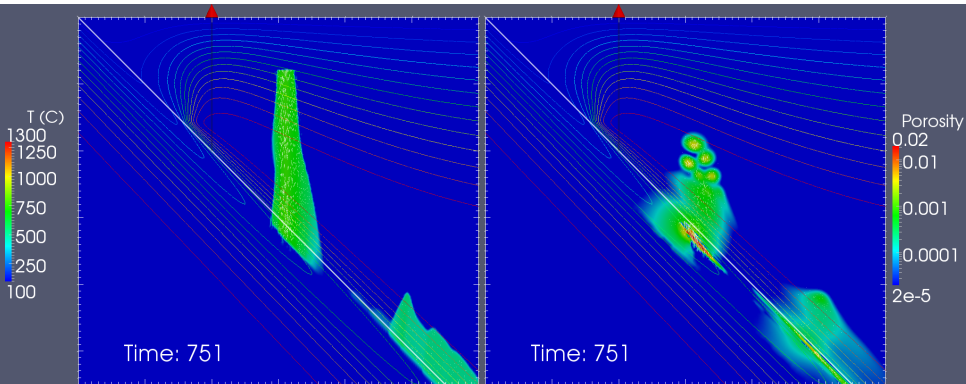
Buoyancy and Pore Pressure



# Fluid Migration

Buoyancy

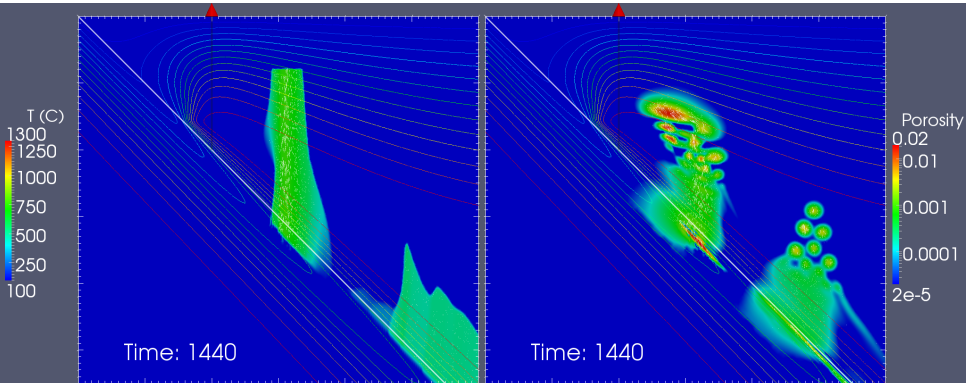
Buoyancy and Pore Pressure



# Fluid Migration

Buoyancy

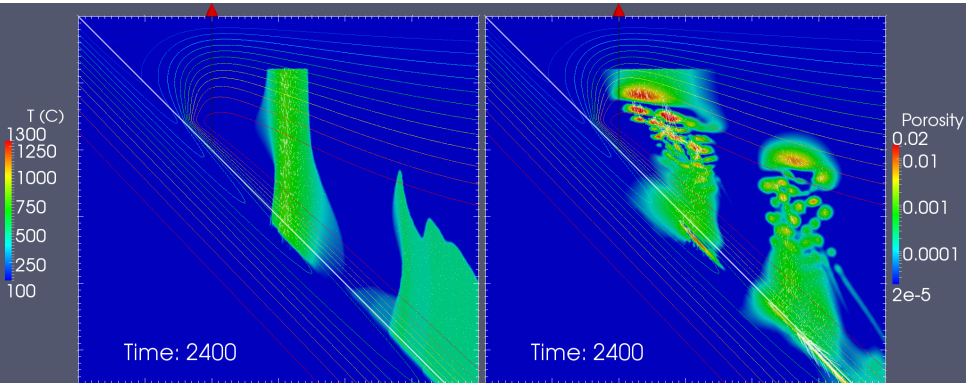
Buoyancy and Pore Pressure



# Fluid Migration

Buoyancy

Buoyancy and Pore Pressure



# Conclusions

- Many important problems in Earth Sciences can be described by non-linear coupled systems of partial differential equations.
- Much of the complexity modelling these systems stems from the nature of multi-physics where small changes in the coupling between fields or constitutive relations can lead to radical changes in behavior and the resulting demands on discretizations and solvers.
- Many established "single-physics" models have locked in solution strategies that are unsuited or difficult to extend to the more challenging non-linear behaviour of a multi-physics system.
- We require a significant increase in flexibility for users for both composing problems (from simple model problems to more 'realistic' coupled simulations) and changing solver strategies.
- Several advanced, open-source computational libraries mean that it is now possible to provide much greater flexibility to the user.
- TerraFERMA is a model building framework that gives access to these libraries while also providing a transparent description of the model.





# Equations (Katz et al., 2007)

Solid flow for solid velocity,  $\mathbf{v}_s$ , and dynamic pressure,  $p^*$ :

$$\nabla \cdot \left( 2 \eta \left( \frac{\nabla \mathbf{v}_s + \nabla \mathbf{v}_s^T}{2} \right) \right) = \nabla p^* + \varphi_0 \varphi \mathbf{k}$$

$$\nabla \cdot \mathbf{v}_s = \varphi_0 \frac{\mathcal{P}}{\zeta}$$

Fluid flow for pore pressure,  $\mathcal{P}$ , and porosity,  $\varphi$ :

$$-\nabla \cdot \frac{K}{\mu} \nabla \mathcal{P} + \frac{\mathcal{P}}{\zeta} = -\nabla \cdot \left( \frac{K}{\mu} \mathbf{k} - \nabla p^* \right) + \Gamma \frac{\Delta \rho}{\rho_f \varphi_0}$$

$$\frac{D\varphi}{Dt} = (1 - \varphi_0 \varphi) \frac{\mathcal{P}}{\zeta} + \frac{\Gamma}{\varphi_0}$$

Constitutive relations:

$$\eta = \eta(T, \dot{\epsilon}, \varphi), \quad \zeta = \eta \varphi^{-m}, \quad K = \varphi^n$$

# Equations (Katz et al., 2007)

Solid flow for solid velocity,  $\mathbf{v}_s$ , and dynamic pressure,  $p^*$ :

$$\nabla \cdot \left( 2 \eta \left( \frac{\nabla \mathbf{v}_s + \nabla \mathbf{v}_s^T}{2} \right) \right) = \nabla p^* + \varphi_0 \varphi \mathbf{k}$$

$$\nabla \cdot \mathbf{v}_s = \varphi_0 \frac{\mathcal{P}}{\zeta}$$

Fluid flow for pore pressure,  $\mathcal{P}$ , and porosity,  $\varphi$ :

$$-\nabla \cdot \frac{K}{\mu} \nabla \mathcal{P} + \frac{\mathcal{P}}{\zeta} = -\nabla \cdot \left( \frac{K}{\mu} \mathbf{k} - \nabla p^* \right) + \Gamma \frac{\Delta \rho}{\rho_f \varphi_0}$$

$$\frac{D\varphi}{Dt} = (1 - \varphi_0 \varphi) \frac{\mathcal{P}}{\zeta} + \frac{\Gamma}{\varphi_0}$$

Constitutive relations:

$$\eta = \eta(T, \dot{\epsilon}, \varphi), \quad \zeta = \eta \varphi^{-m}, \quad K = \varphi^n$$

# Equations (Katz et al., 2007)

Solid flow for solid velocity,  $\mathbf{v}_s$ , and dynamic pressure,  $p^*$ :

$$\nabla \cdot \left( 2\eta \left( \frac{\nabla \mathbf{v}_s + \nabla \mathbf{v}_s^T}{2} \right) \right) = \nabla p^* + \varphi_0 \varphi \mathbf{k}$$
$$\nabla \cdot \mathbf{v}_s = \varphi_0 \frac{\mathcal{P}}{\zeta}$$

Fluid flow for pore pressure,  $\mathcal{P}$ , and porosity,  $\varphi$ :

$$-\nabla \cdot \left( \frac{K}{\mu} \nabla \mathcal{P} + \frac{\mathcal{P}}{\zeta} \right) = -\nabla \cdot \left( \frac{K}{\mu} \mathbf{k} - \nabla p^* \right) + \Gamma \frac{\Delta \rho}{\rho_f \varphi_0}$$
$$\frac{D\varphi}{Dt} = (1 - \varphi_0 \varphi) \frac{\mathcal{P}}{\zeta} + \frac{\Gamma}{\varphi_0}$$

Constitutive relations:

$$\eta = \eta(T, \dot{\epsilon}, \varphi), \quad \zeta = \eta \varphi^{-m}, \quad K = \varphi^n$$

# Equations (Katz et al., 2007)

Solid flow for solid velocity,  $\mathbf{v}_s$ , and dynamic pressure,  $p^*$ :

$$\nabla \cdot \left( 2 \eta \left( \frac{\nabla \mathbf{v}_s + \nabla \mathbf{v}_s^T}{2} \right) \right) = \nabla p^* + \varphi_0 \varphi \mathbf{k}$$

$$\nabla \cdot \mathbf{v}_s = \varphi_0 \frac{\mathcal{P}}{\zeta}$$

Fluid flow for pore pressure,  $\mathcal{P}$ , and porosity,  $\varphi$ :

$$-\nabla \cdot \frac{K}{\mu} \nabla \mathcal{P} + \frac{\mathcal{P}}{\zeta} = -\nabla \cdot \left( \frac{K}{\mu} \mathbf{k} - \nabla p^* \right) + \Gamma \frac{\Delta \rho}{\rho_f \varphi_0}$$

$$\frac{D\varphi}{Dt} = (1 - \varphi_0 \varphi) \frac{\mathcal{P}}{\zeta} + \frac{\Gamma}{\varphi_0}$$

Constitutive relations:

$$\eta = \eta(T, \dot{\epsilon}, \varphi), \quad \zeta = \eta \varphi^{-m}, \quad K = \varphi^n$$

# Equations (Katz et al., 2007)

Solid flow for solid velocity,  $\mathbf{v}_s$ , and dynamic pressure,  $p^*$ :

$$\nabla \cdot \left( 2 \eta \left( \frac{\nabla \mathbf{v}_s + \nabla \mathbf{v}_s^T}{2} \right) \right) = \nabla p^* + \varphi_0 \varphi \mathbf{k}$$

$$\nabla \cdot \mathbf{v}_s = \varphi_0 \frac{\mathcal{P}}{\zeta}$$

Fluid flow for pore pressure,  $\mathcal{P}$ , and porosity,  $\varphi$ :

$$-\nabla \cdot \frac{K}{\mu} \nabla \mathcal{P} + \frac{\mathcal{P}}{\zeta} = -\nabla \cdot \left( \frac{K}{\mu} \mathbf{k} - \nabla p^* \right) + \Gamma \frac{\Delta \rho}{\rho_f \varphi_0}$$

$$\frac{D\varphi}{Dt} = \left( 1 - \varphi_0 \varphi \right) \frac{\mathcal{P}}{\zeta} + \frac{\Gamma}{\varphi_0}$$

Constitutive relations:

$$\eta = \eta(T, \dot{\epsilon}, \varphi), \quad \zeta = \eta \varphi^{-m}, \quad K = \varphi^n$$

# Equations (Katz et al., 2007)

Solid flow for solid velocity,  $\mathbf{v}_s$ , and dynamic pressure,  $p^*$ :

$$\nabla \cdot \left( 2 \eta \left( \frac{\nabla \mathbf{v}_s + \nabla \mathbf{v}_s^T}{2} \right) \right) = \nabla p^*$$

$$\nabla \cdot \mathbf{v}_s = 0$$

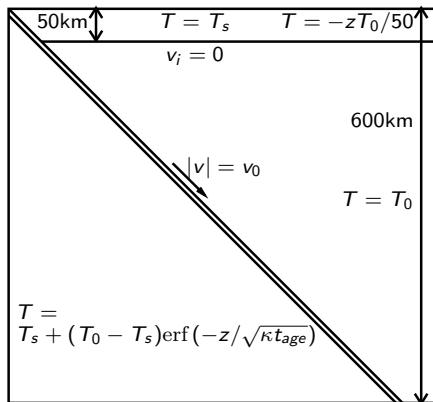
Fluid flow for pore pressure,  $\mathcal{P}$ , and porosity,  $\varphi$ :

$$-\nabla \cdot \frac{K}{\mu} \nabla \mathcal{P} + \frac{\mathcal{P}}{\zeta} = -\nabla \cdot \frac{K}{\mu} \mathbf{k} + \Gamma \frac{\Delta \rho}{\rho_f \varphi_0}$$

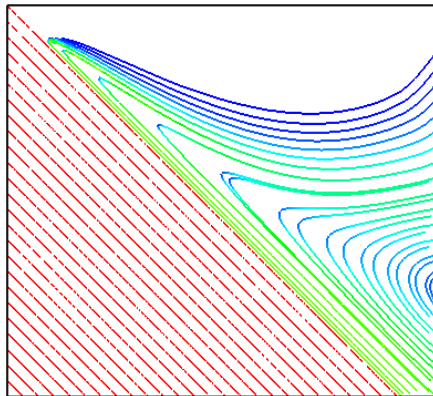
$$\frac{D\varphi}{Dt} = \frac{\mathcal{P}}{\zeta} + \frac{\Gamma}{\varphi_0}$$

Constitutive relations:

$$\eta = \eta(T, \dot{\epsilon}, \varphi), \quad \zeta = \eta \varphi^{-m}, \quad K = \varphi^n$$



Velocity streamlines





Buoyancy and Pore Pressure:

$$-\nabla \cdot \frac{K}{\mu} \nabla \mathcal{P} + \frac{\mathcal{P}}{\zeta} = -\nabla \cdot \frac{K}{\mu} \mathbf{k} + \Gamma \frac{\Delta \rho}{\rho_f \varphi_0}$$

$$\frac{D\varphi}{Dt} = \frac{\mathcal{P}}{\zeta} + \frac{\Gamma}{\varphi_0}$$

Constitutive relations:

$$\eta = \eta(T, \dot{\epsilon}, \varphi), \quad \zeta = \eta \varphi^{-m}, \quad K = \varphi^n$$

Buoyancy (zero compaction length approximation):

$$\frac{D\varphi}{Dt} = -\nabla \cdot \frac{K}{\mu} \mathbf{k} + \frac{\Gamma}{\varphi_0} \left( \frac{\Delta\rho}{\rho_f} + 1 \right)$$

Buoyancy and Pore Pressure:

$$-\nabla \cdot \frac{K}{\mu} \nabla \mathcal{P} + \frac{\mathcal{P}}{\zeta} = -\nabla \cdot \frac{K}{\mu} \mathbf{k} + \Gamma \frac{\Delta\rho}{\rho_f \varphi_0}$$

$$\frac{D\varphi}{Dt} = \frac{\mathcal{P}}{\zeta} + \frac{\Gamma}{\varphi_0}$$

Constitutive relations:

$$\eta = \eta(T, \dot{\epsilon}, \varphi), \quad \zeta = \eta \varphi^{-m}, \quad K = \varphi^n$$