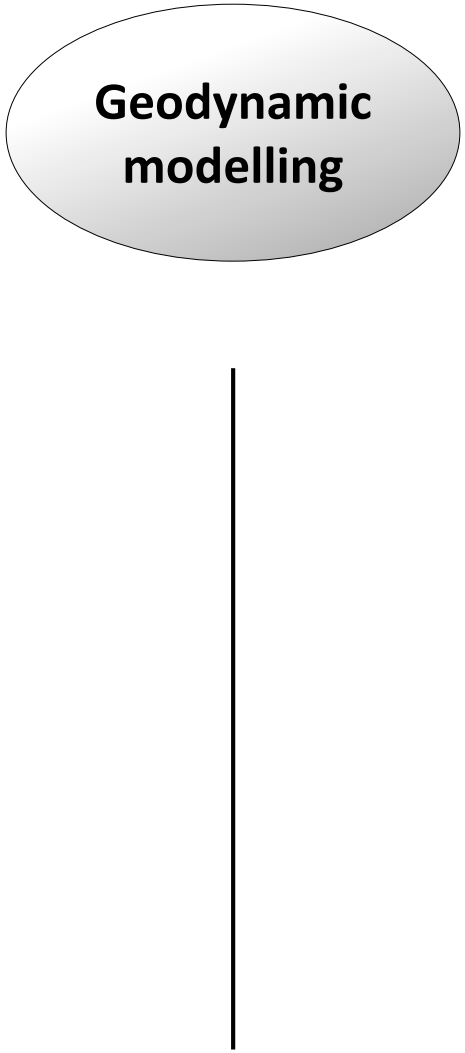
JOHANNES GUTENBERG UNIVERSITÄT MAINZ
MAX-PLANCK-GESELLSCHAFT
INSTITUTE FOR GEOSCIENCES

Adjoint based inversion in geodynamic modelling: A guide

Georg Reuber,
Boris Kaus, Anton Popov

DFG Deutsche Forschungsgemeinschaft
German Research Foundation

Motivation



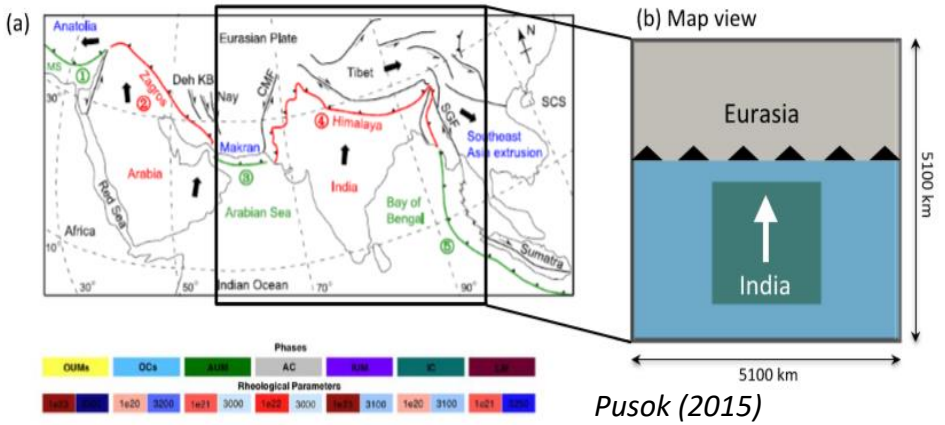
Motivation

Geodynamic modelling

Idealized test

- Parameterize real scenario

→ Easy to test/quantify effects



Pusok (2015)

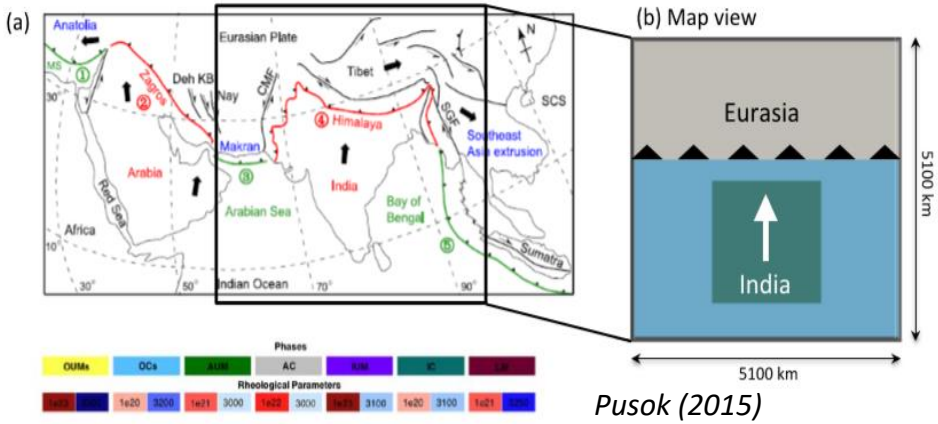
Motivation

Geodynamic modelling

Idealized test

- Parameterize real scenario

→ Easy to test/quantify effects

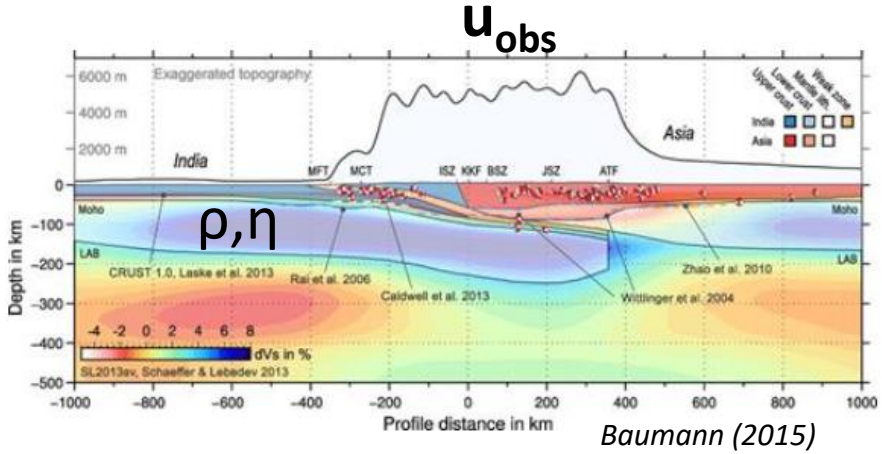


Pusok (2015)

Inversion

- Fit data

→ Ultimately represents nature



Baumann (2015)

Motivation

Inversion

Measurement device

Physical description

$$\nabla \cdot \sigma = (0, \rho g)^T$$

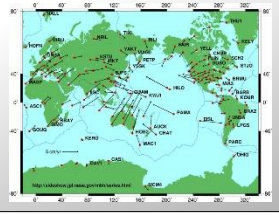
$$\nabla \cdot \mathbf{u} = 0$$

Motivation

Inversion

Measurement device

(Field) measurement: $\mathbf{u}_{obs}$

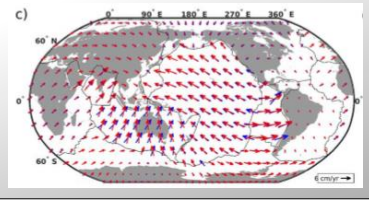


Physical description

$$\nabla \cdot \sigma = (0, \rho g)^T$$

$$\nabla \cdot \mathbf{u} = 0$$

Computer simulation: **u**

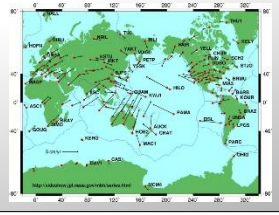


Motivation

Inversion

Measurement device

(Field) measurement: $\mathbf{u}_{obs}$

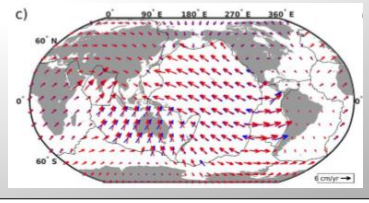


Physical description

$$\nabla \cdot \sigma = (0, \rho g)^T$$

$$\nabla \cdot \mathbf{u} = 0$$

Computer simulation: **u**



Cost function:

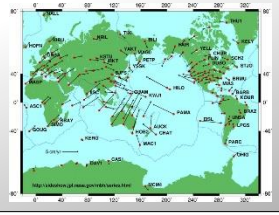
$F = 0.5(u-u_{obs})^T(u-u_{obs})$

Motivation

Inversion

Measurement device

(Field) measurement: $\mathbf{u}_{obs}$

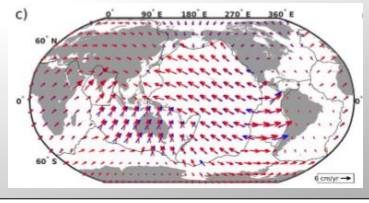


Physical description

$$\nabla \cdot \sigma = (0, \rho g)^T$$

$$\nabla \cdot \mathbf{u} = 0$$

Computer simulation: **u**



Cost function:

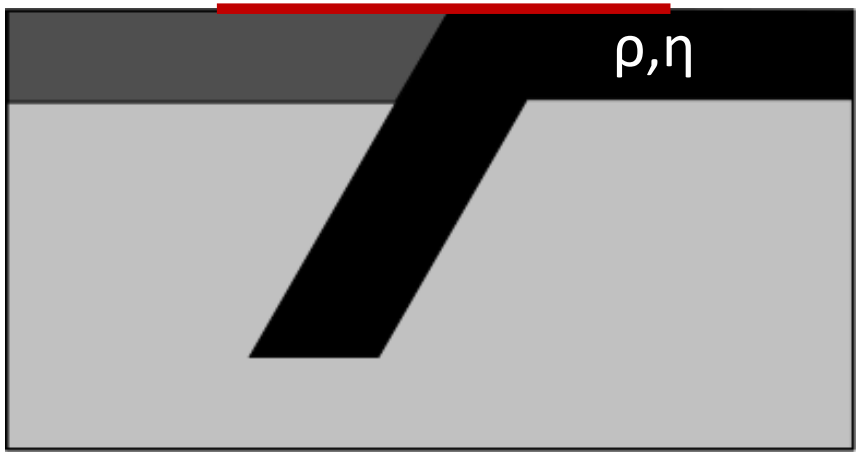
$F = 0.5(u-u_{obs})^T(u-u_{obs})$

Update computer Simulation until F = 0

Motivation

Inversion

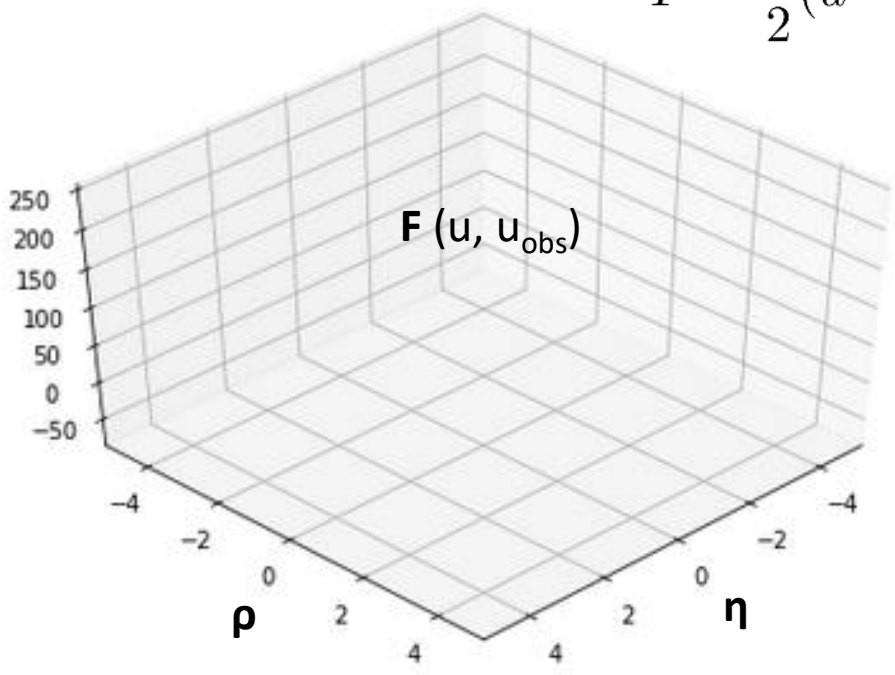
E.g: u_{obs}

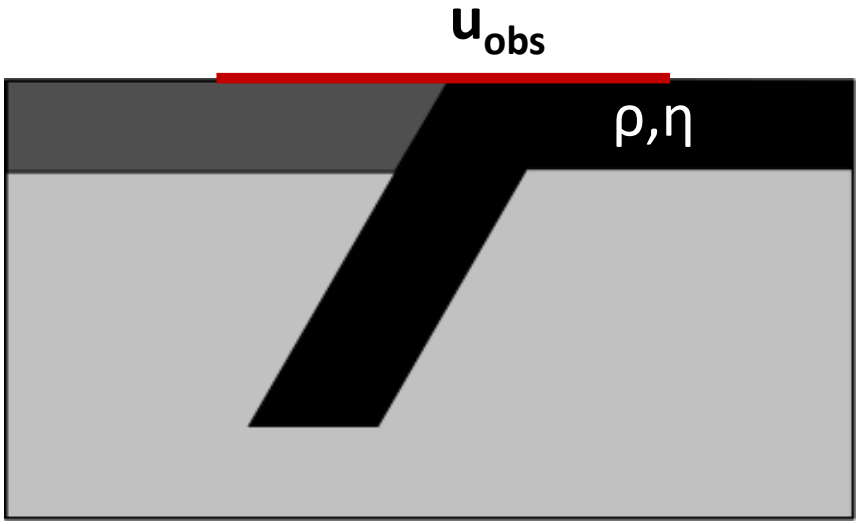


Motivation

Inversion

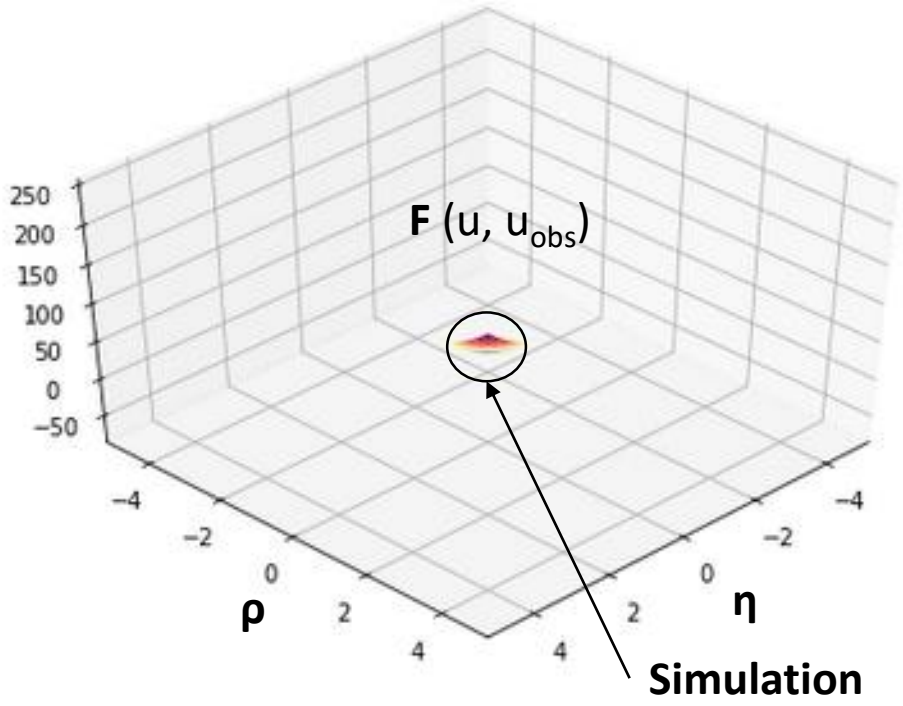
$$F = \frac{1}{2}(u - u_{obs})^T (u - u_{obs})$$

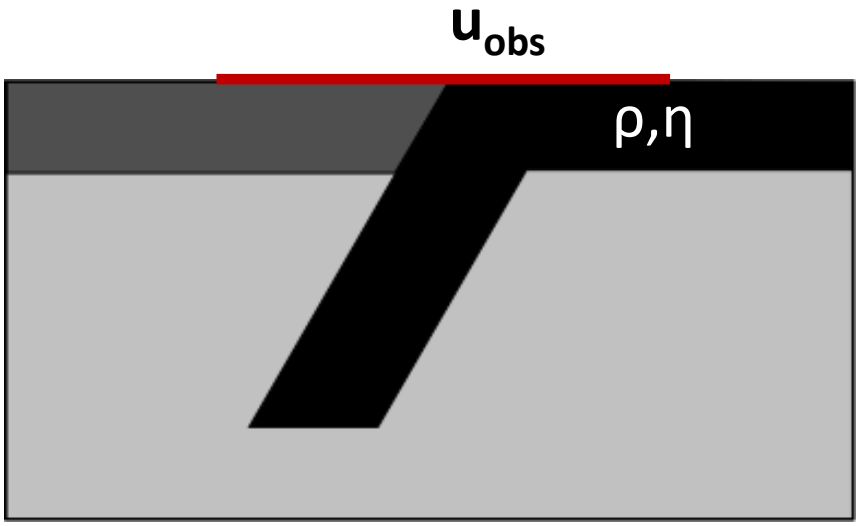




Motivation

Inversion

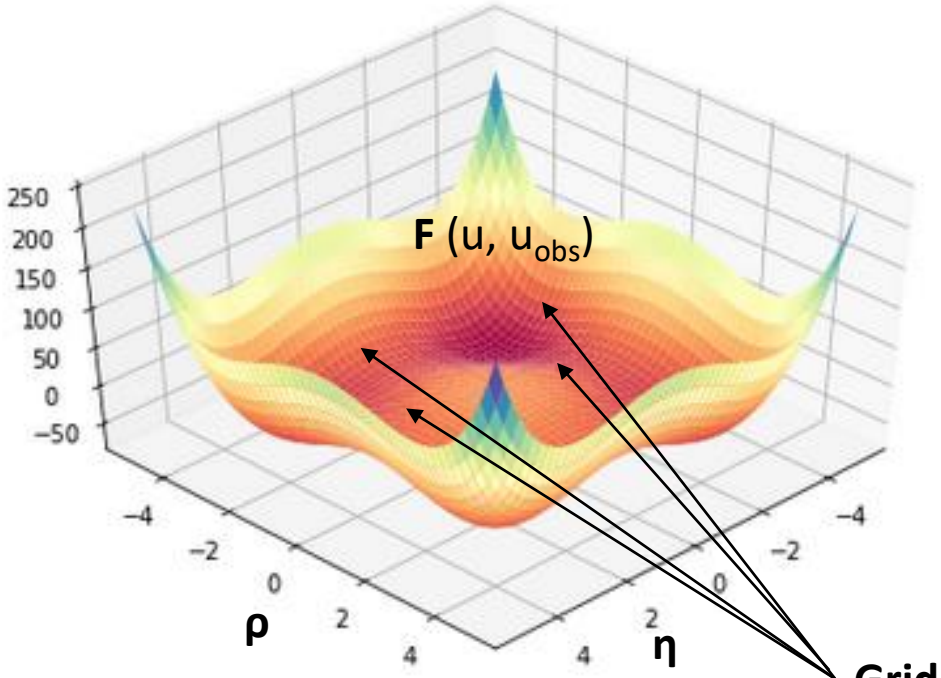


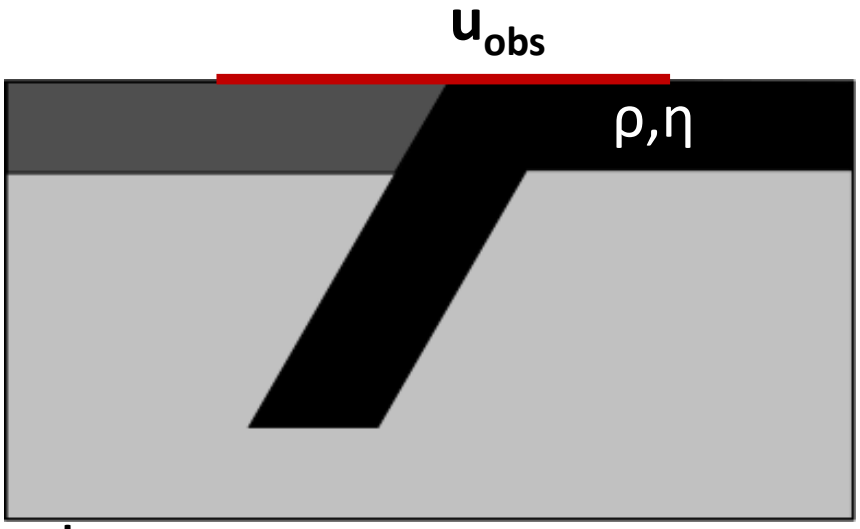


→ **Find minimum of F**

Motivation

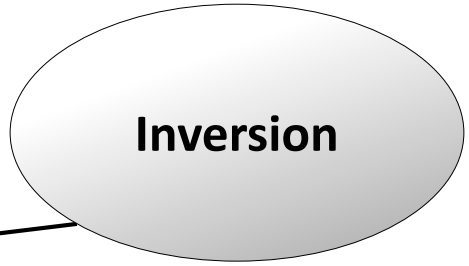
Inversion





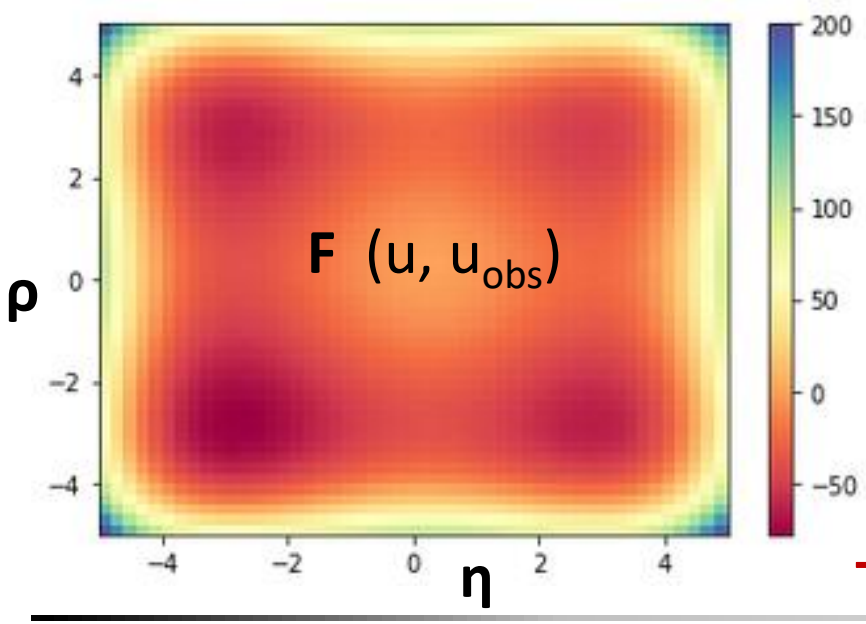
→ **Find minimum of F**

Motivation



Statistical

- (Randomized) statistical search



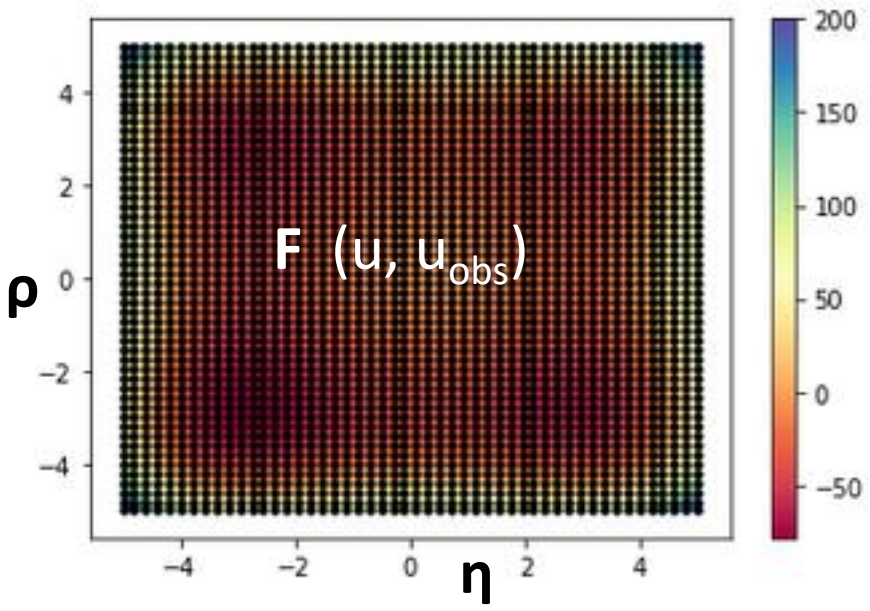
→ **Find minimum of F**

Motivation

Inversion

Statistical

- *Grid Search Sampling*



Motivation

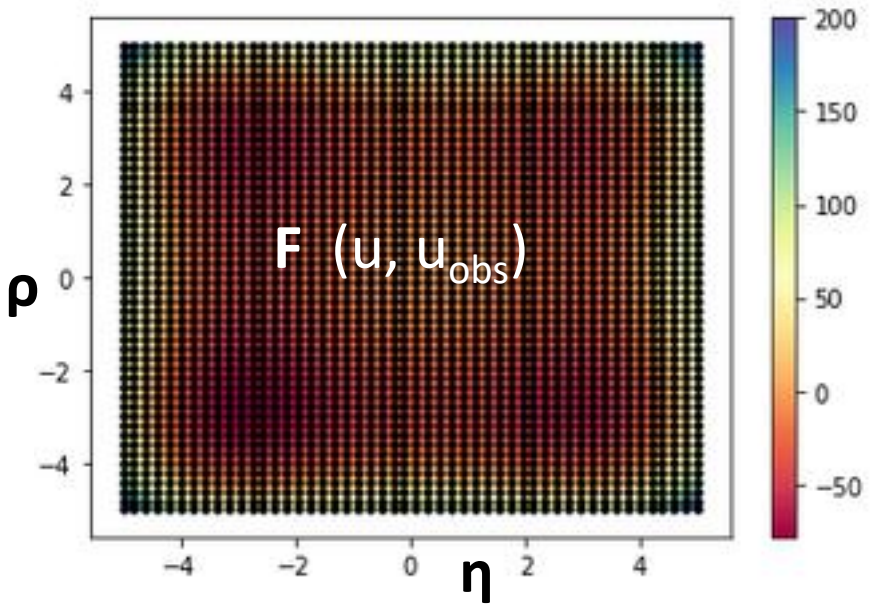
Inversion

Statistical

- *Grid Search Sampling*

→ Many simulations

→ Broad knowledge of cost function

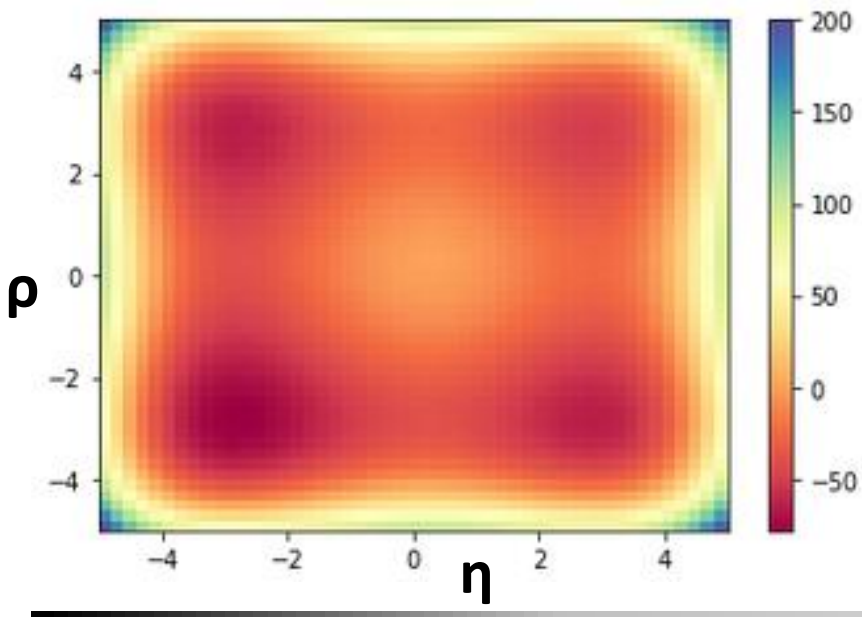


Motivation

Inversion

Statistical

- *Monte-Carlo Sampling*

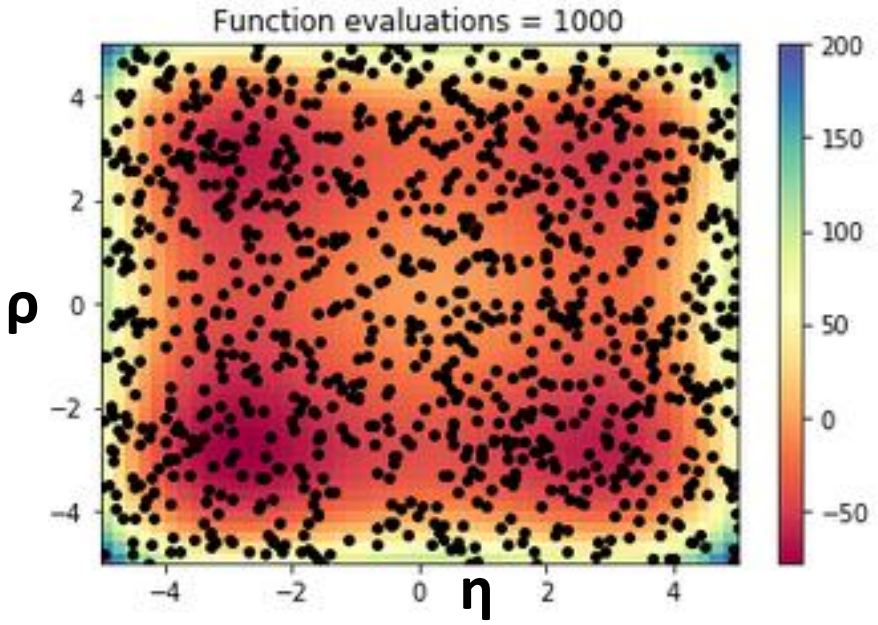


Motivation

Inversion

Statistical

- *Monte-Carlo Sampling*



Motivation

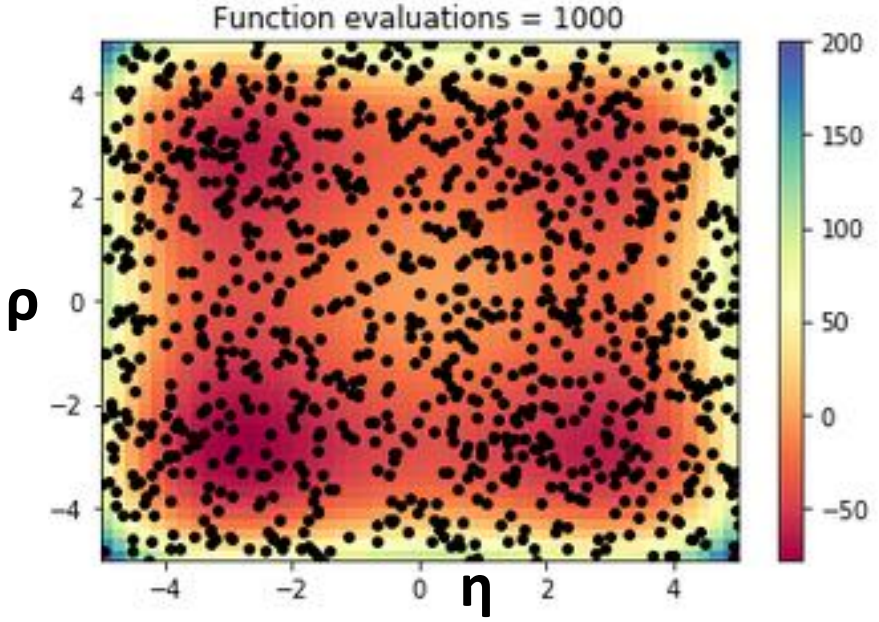
Inversion

Statistical

- *Monte-Carlo Sampling*

→ Many simulations

→ Broad knowledge of cost function

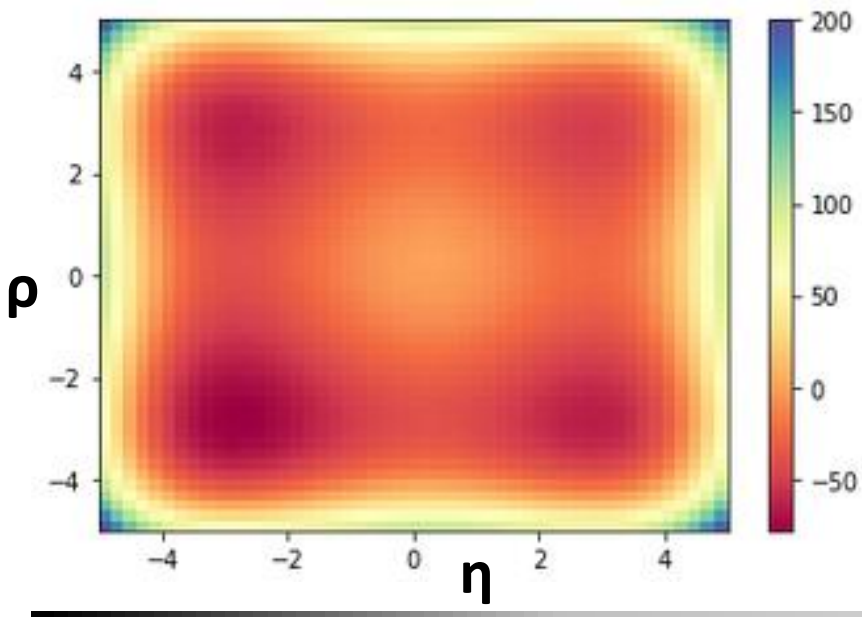


Motivation

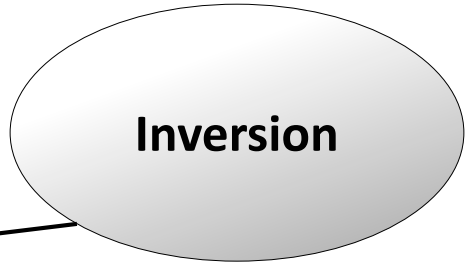
Inversion

Statistical

- *MCMC Sampling*

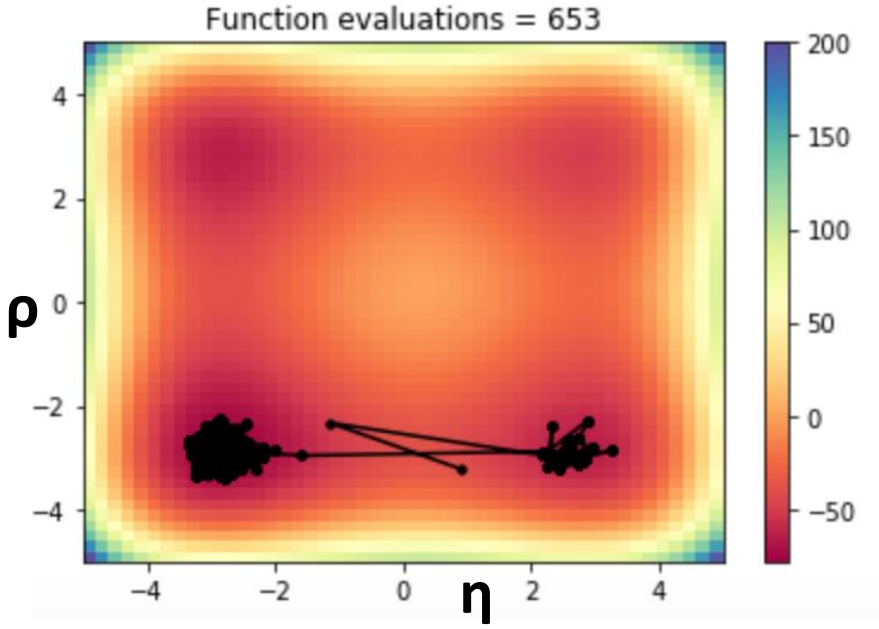


Motivation

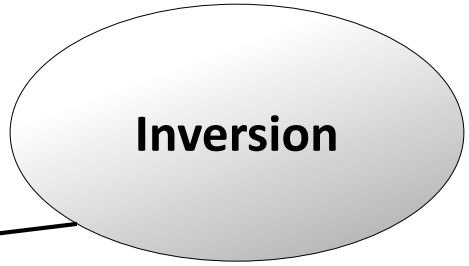


Statistical

- *MCMC Sampling*



Motivation

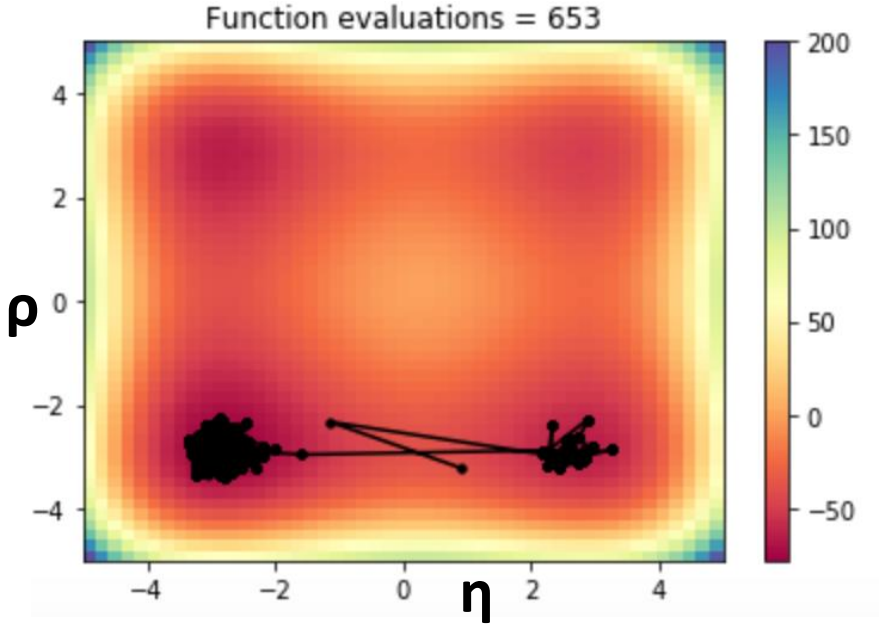


Statistical

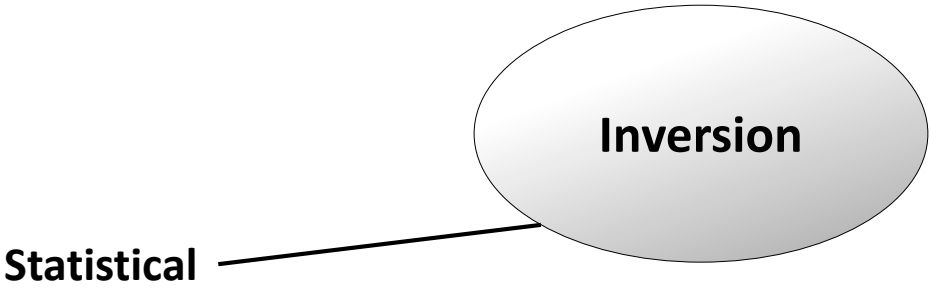
- *MCMC Sampling*

→ Few simulations

→ Requires fine-tuning

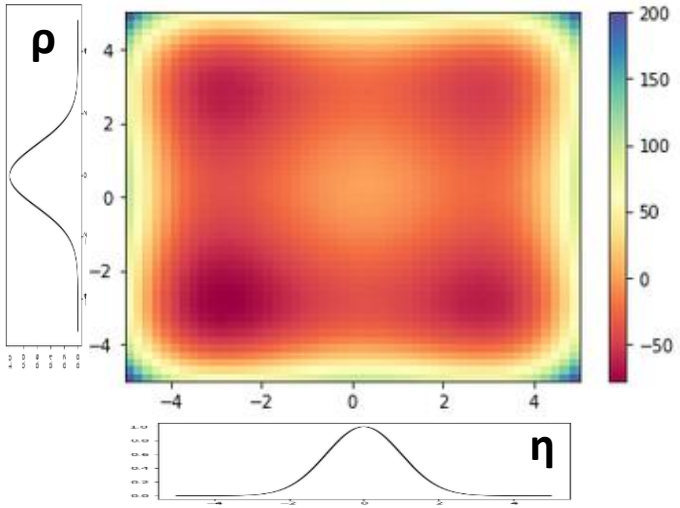


Motivation



- *Bayes Theorem*

→ Prior knowledge goes in sampling



Motivation

Inversion

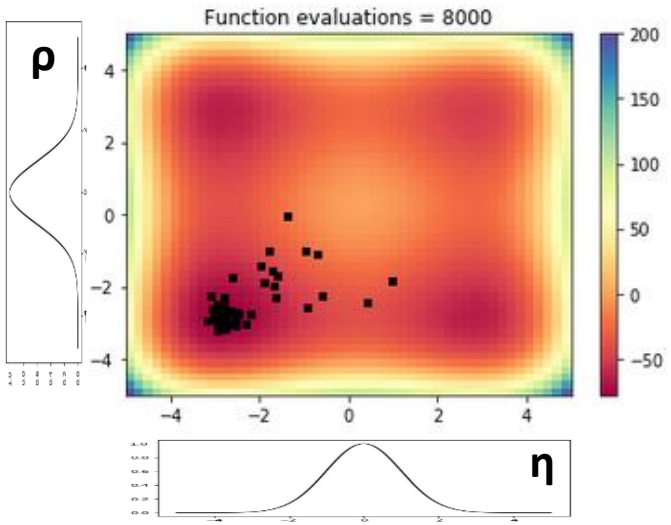
Statistical

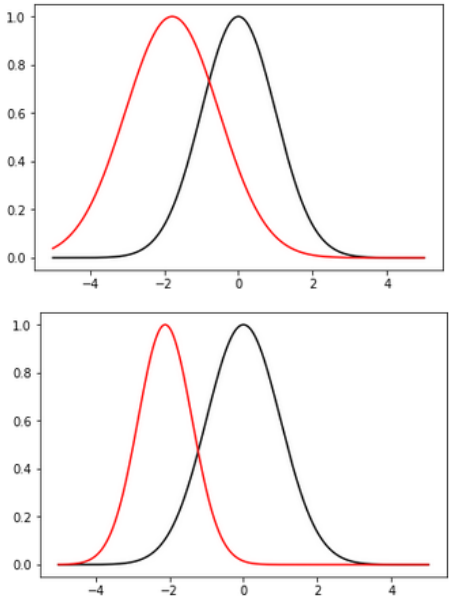
- *Bayes Theorem*

→ Prior knowledge goes in sampling

→ Compute posterior

→ Uncertainty in parameters





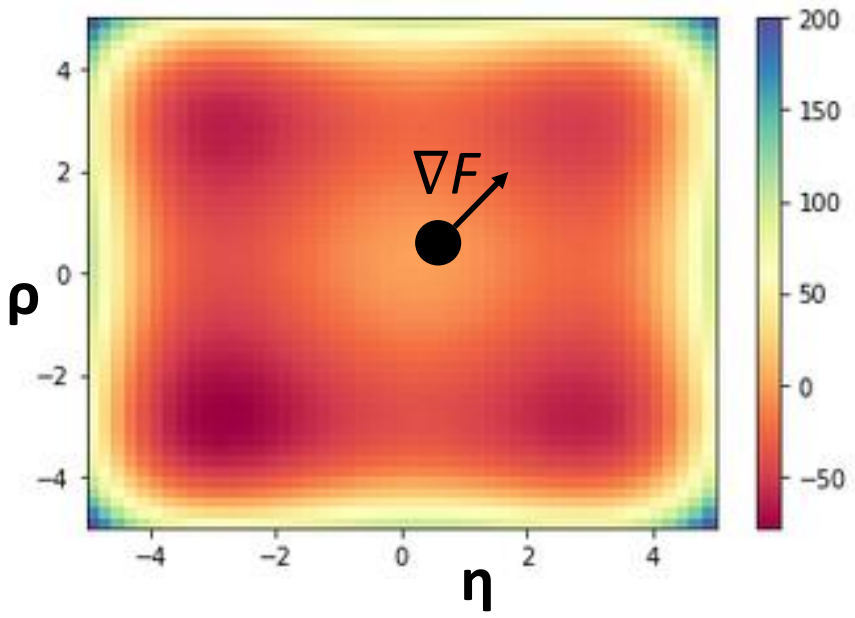
… many more methods

Motivation

Inversion

Deterministic

Gradient based:



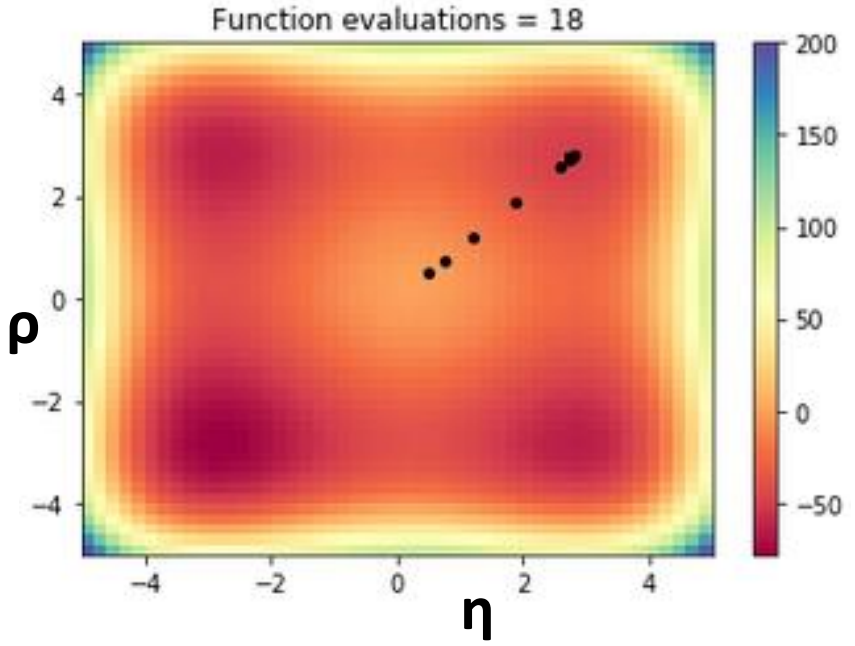
Motivation

Inversion

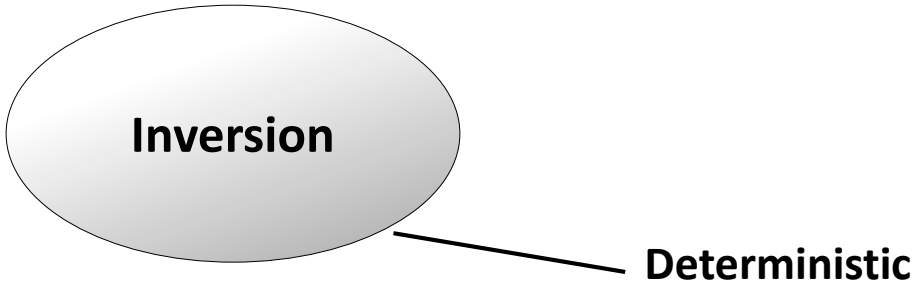
Deterministic

Gradient based:

- Few ‚simulations‘
- Sensitive to local minima



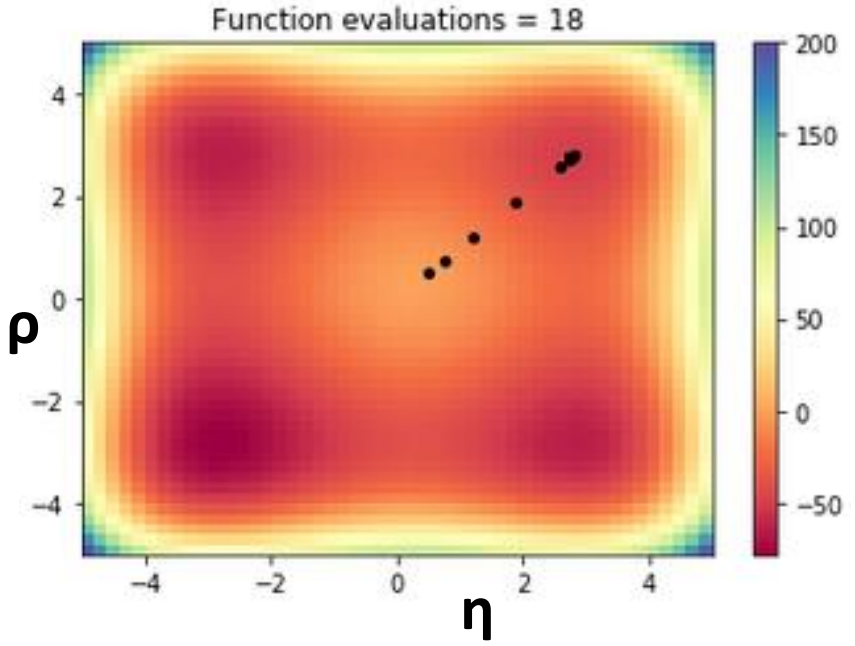
Motivation



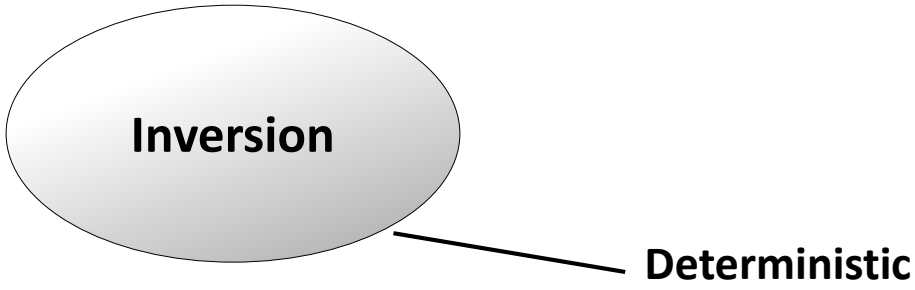
Gradient based [gradient descent]:

$$F = \frac{1}{2}(u - u_{obs})^T (u - u_{obs})$$

$$\mathbf{p}^{n+1} = \mathbf{p}^n - \alpha \frac{dF}{d\mathbf{p}}$$



Motivation



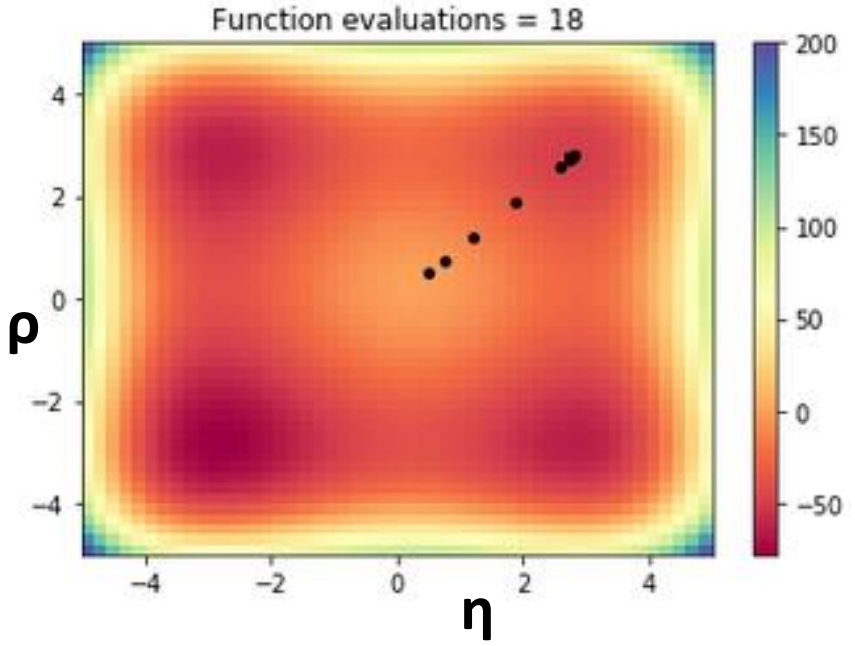
Gradient based [(Quasi)-Newton]:

$$F = \frac{1}{2}(u - u_{obs})^T (u - u_{obs})$$

$$\mathbf{p}^{n+1} = \mathbf{p}^n - \alpha \left[\frac{d}{d\mathbf{p}} \left(\frac{dF}{d\mathbf{p}} \right) \right]^{-1} \frac{dF}{d\mathbf{p}}$$

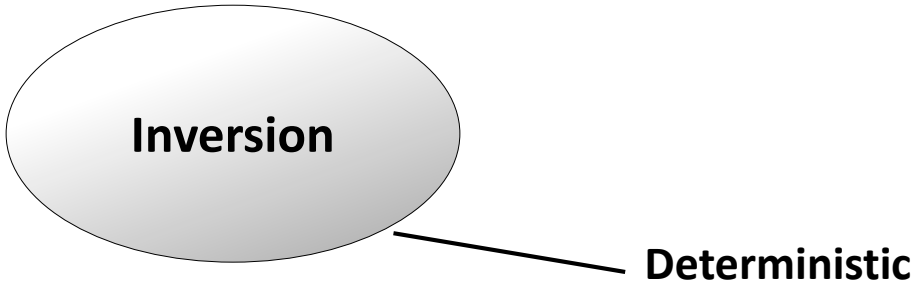
Hessian matrix

→ *Relates to the covariance matrix in Bayesian context*



… more methods

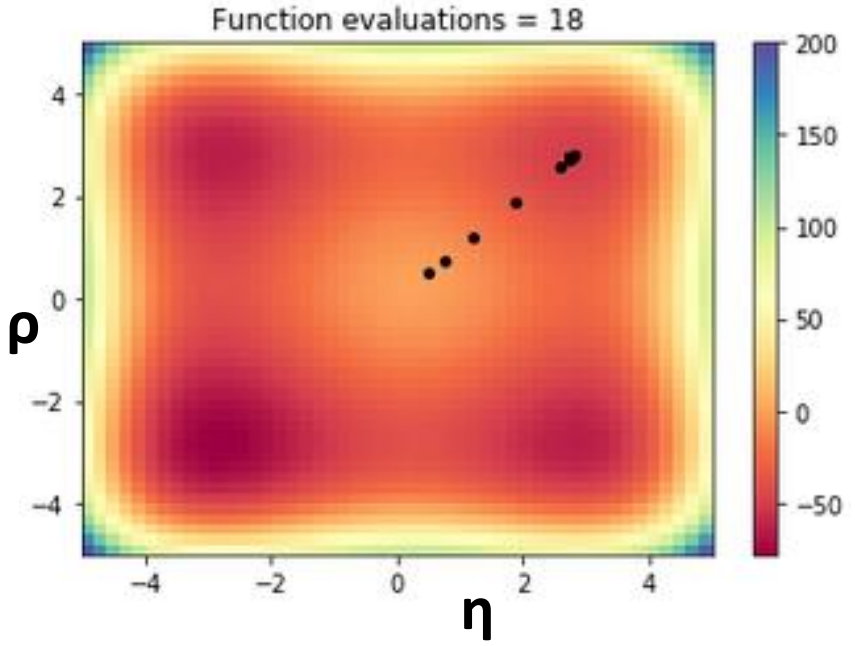
Motivation



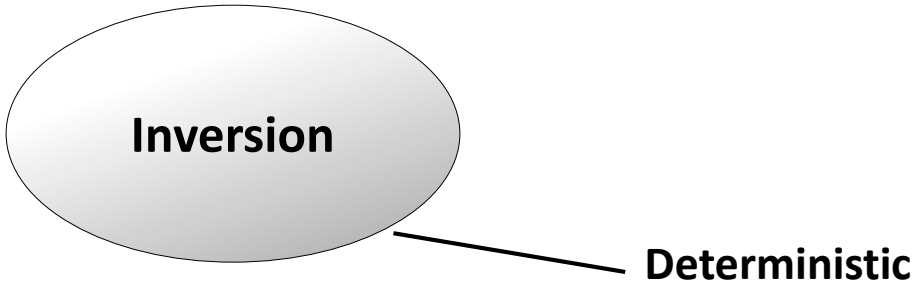
Gradient based:

- Few ‚simulations‘
- Sensitive to local minima

- **How to evaluate gradient?**



Motivation



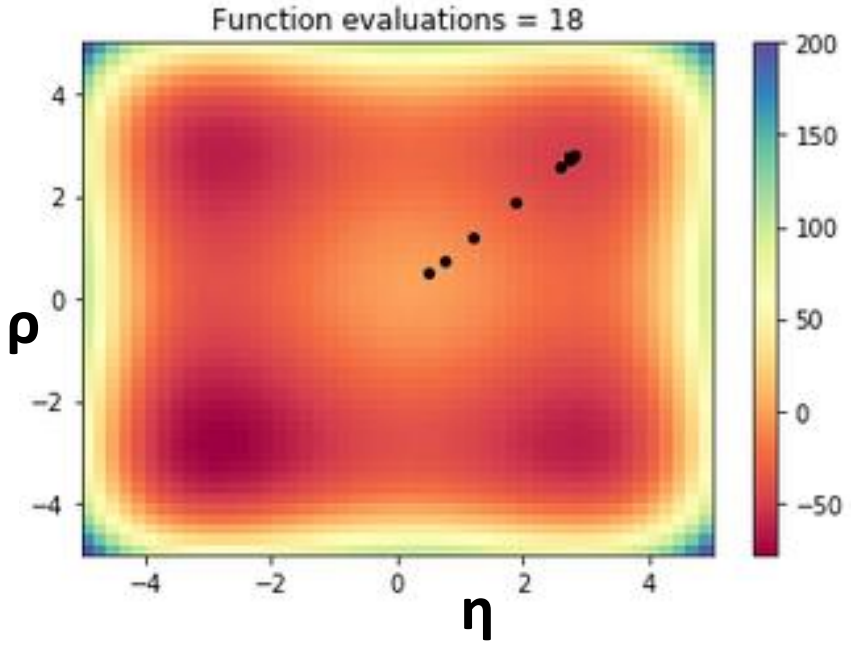
Gradient based:

- Few ‚simulations‘
- Sensitive to local minima

- **How to evaluate gradient?**

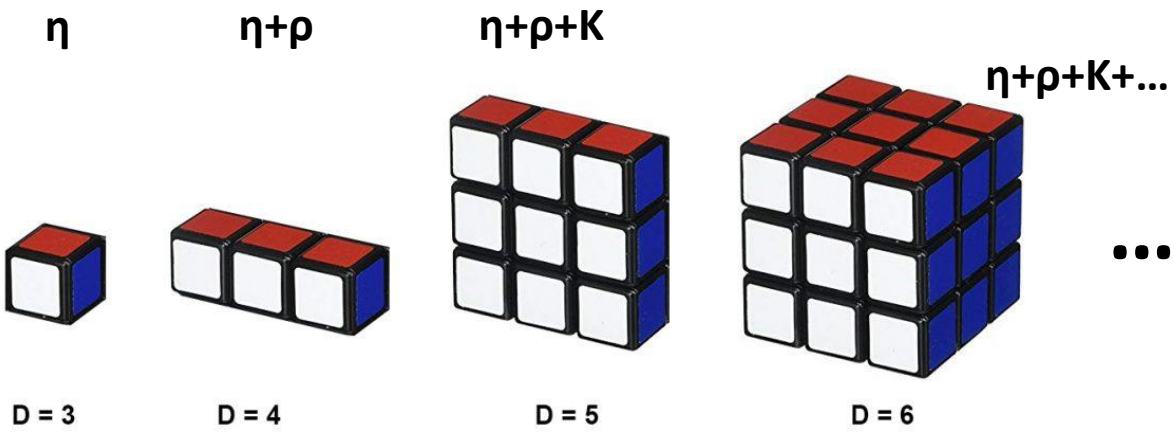
→ Here simple: analytical

→ Or FD: $\frac{dF}{d\mathbf{p}} = \frac{F(\mathbf{p}+h) - F(\mathbf{p})}{h}$



Motivation

nD-Inversion



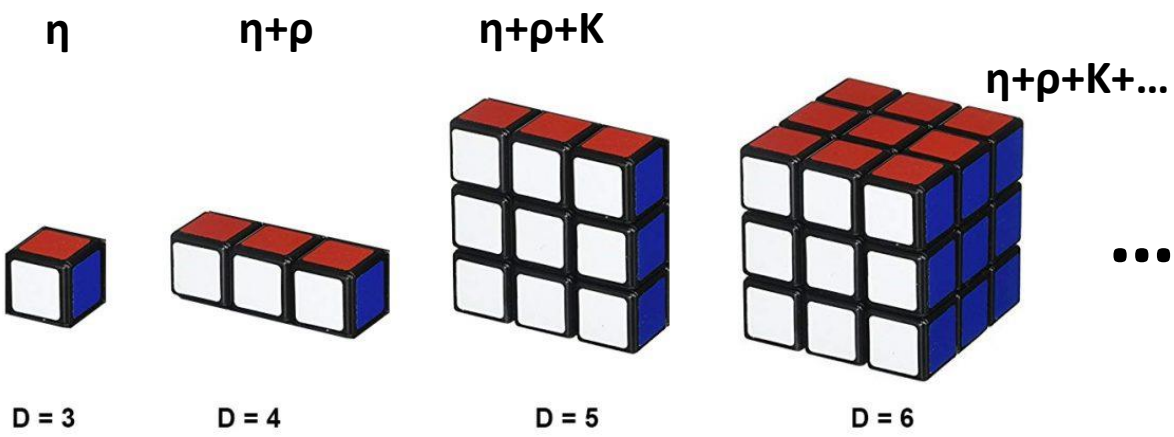
… n dimensions

- 5 rock phases in thermo-elasto-visco-plastic model = **40 parameters** (=dimensions)
- Seismology: wavespeed at every node is a free parameter = **# nodes parameters** (billions)

→ Every viscosity free parameter = # nodes parameters

Motivation

nD-Inversion

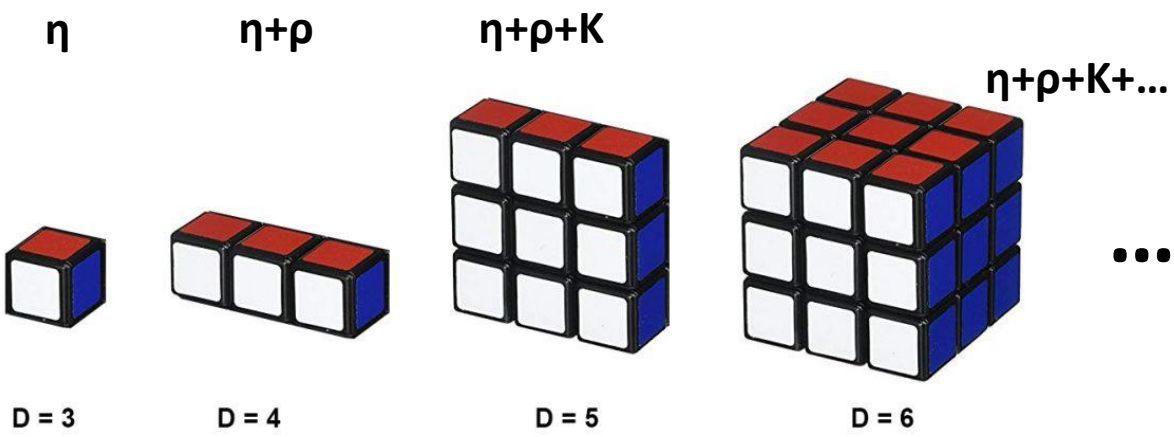


… n dimensions

- **Statistical:** 40 parameters + 10 samples per parameter = 10^40 simulations
- **Deterministic:** Two solves per parameter with FD = 80 simulations (per descent iteration)

Motivation

nD-Inversion



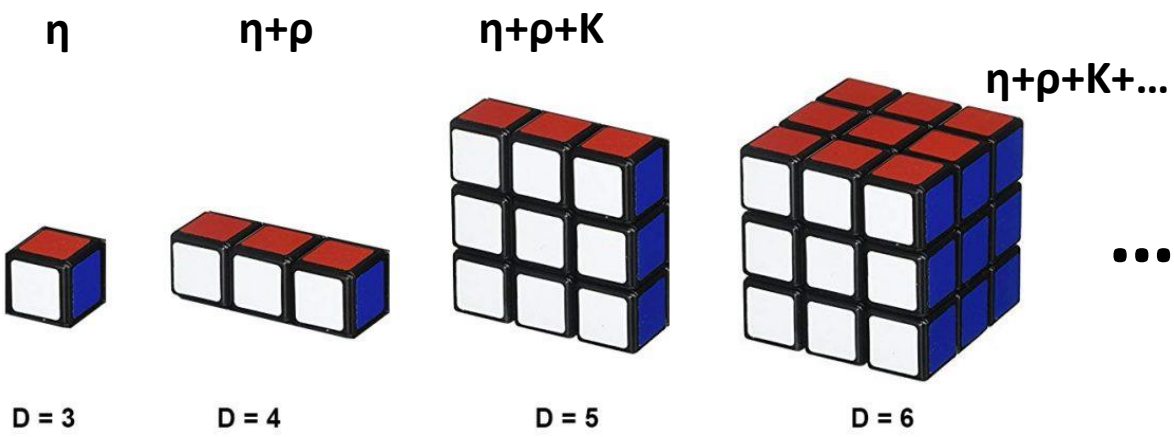
… n dimensions

- **Statistical:** 40 parameters + 10 samples per parameter = 10^40 simulations
- **Deterministic:** Two solves per parameter with FD = 80 simulations (per descent iteration)

→ **More efficient gradient computation available?**

Motivation

nD-Inversion

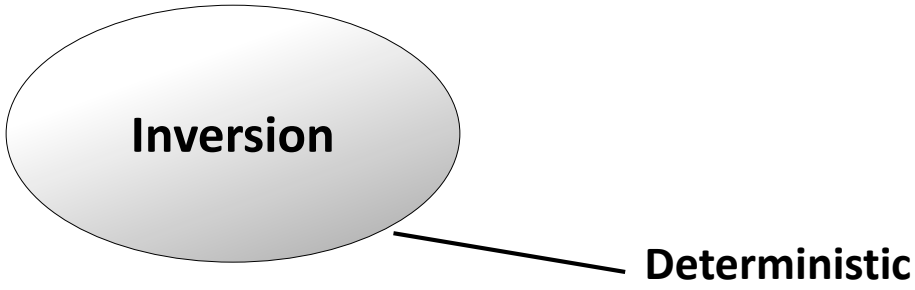


… n dimensions

- **Adjoint method:**
  - Independent of # parameters (very efficient)
  - Requires forward (nonlinear) + adjoint solve (linear) and gradient computation

  → Requires formulation of adjoint equation

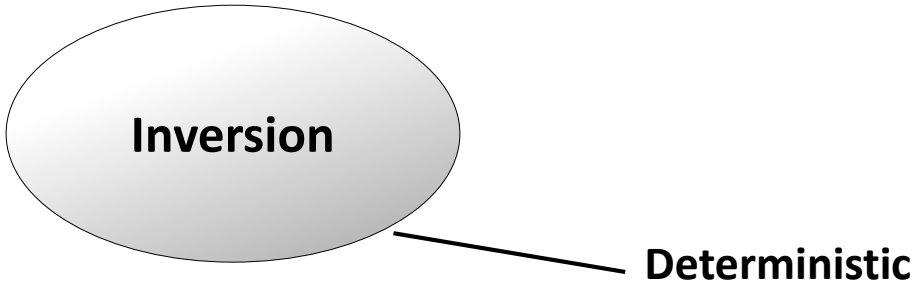
Method



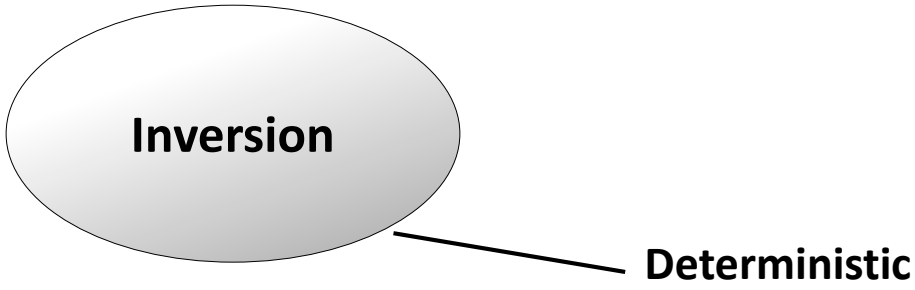
Constrained optimization:

- Appears a lot:
  - E.g. Economics: Maximize profit in terms of labor hours and raw material with limited budget
  - Logistics
  - Thermodynamics
  - …

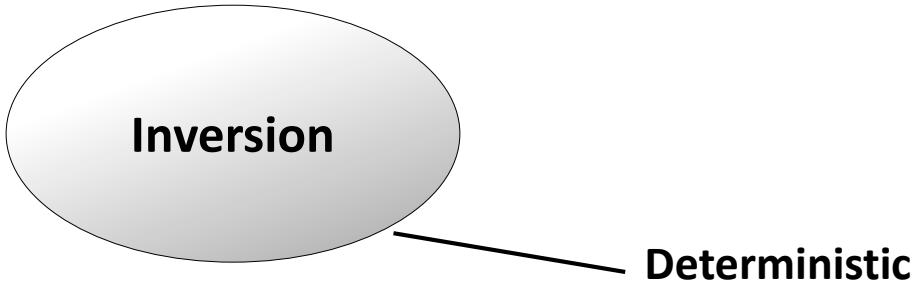
Method



Method

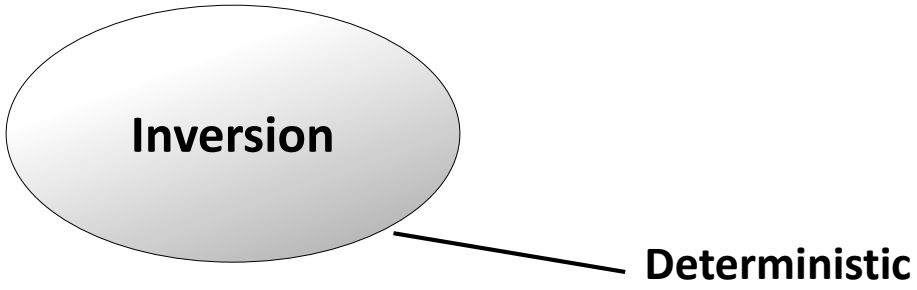


Method



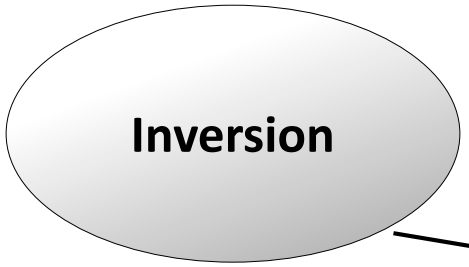
→ **Find lowest point of Hike**

Method

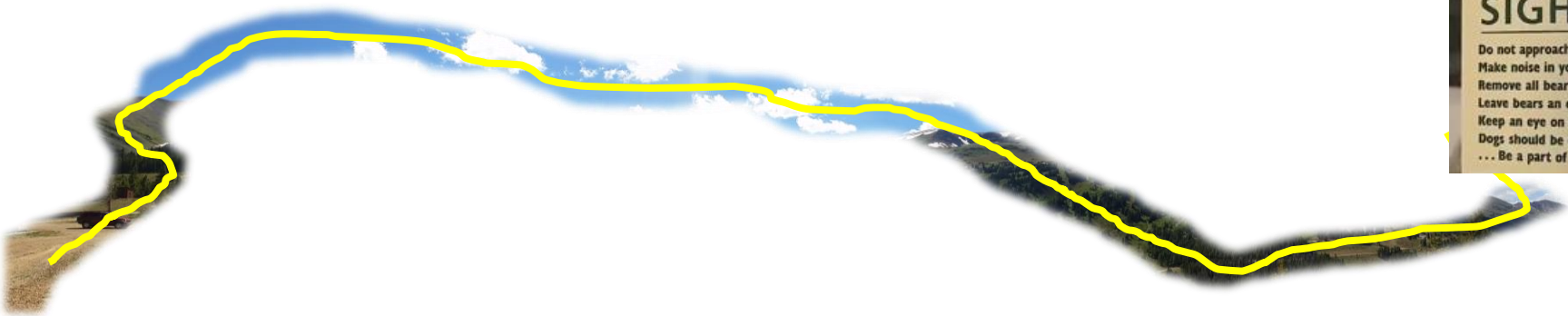


I) National Park (with Bears)

Method

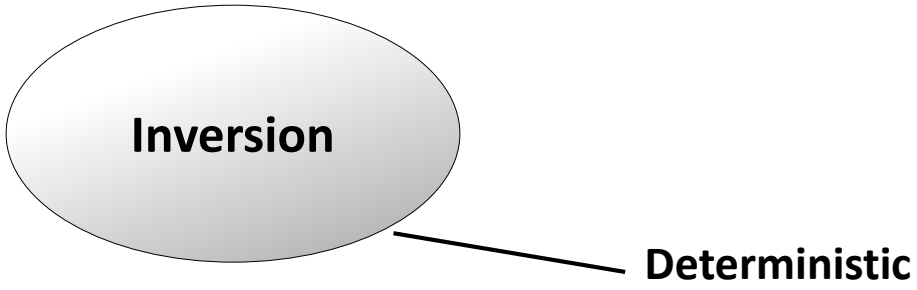


Deterministic



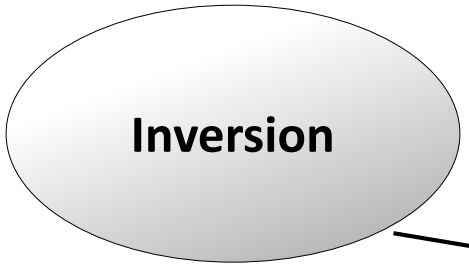
I) National Park (with Bears)
II) Foggy day

Method



I) National Park
II) Foggy day

Method

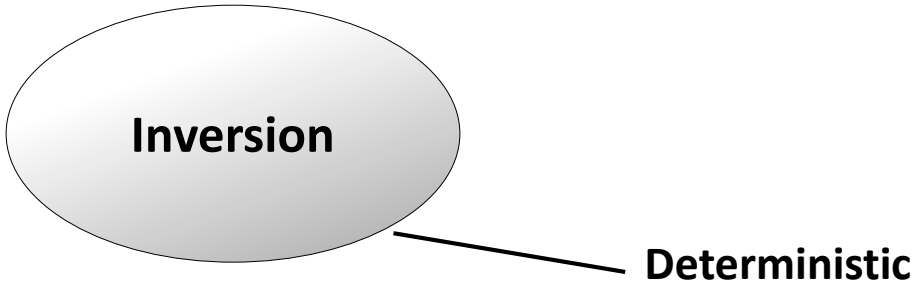


Deterministic

I) National Park
II) Foggy day

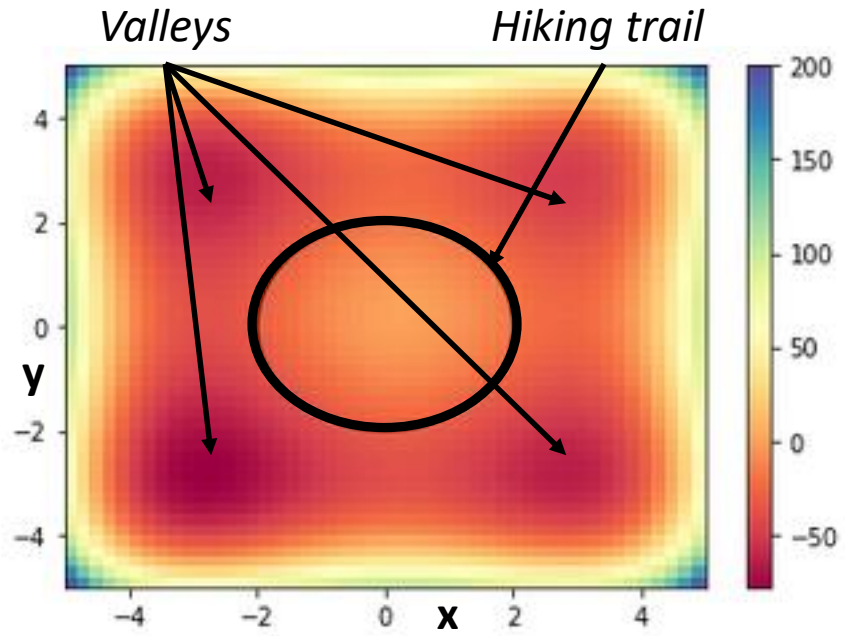
→ Find lowest point of Hike

Method

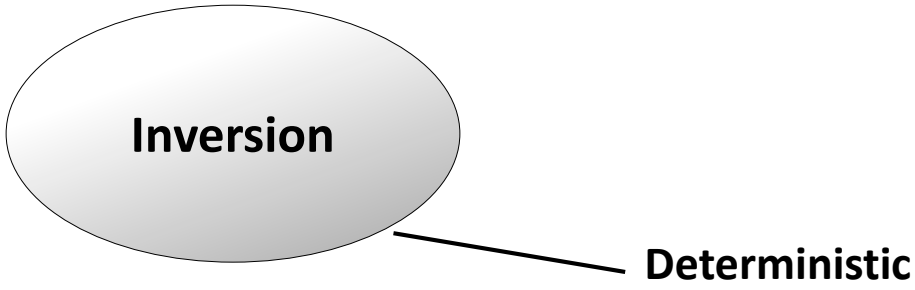


Constrained optimization:

- In our example:
  1. Function represents mountain belt
  2. Find lowest point of hike

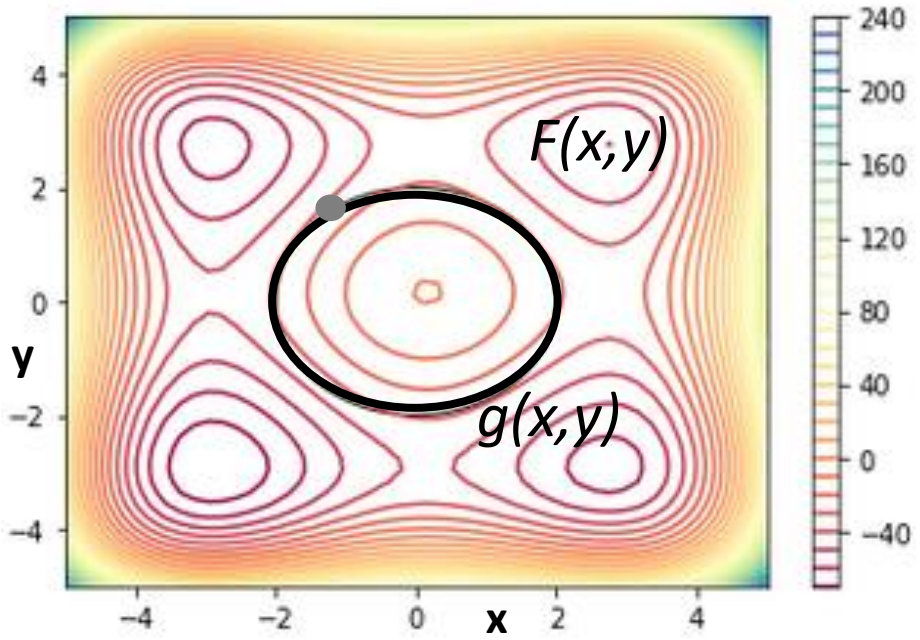


Method

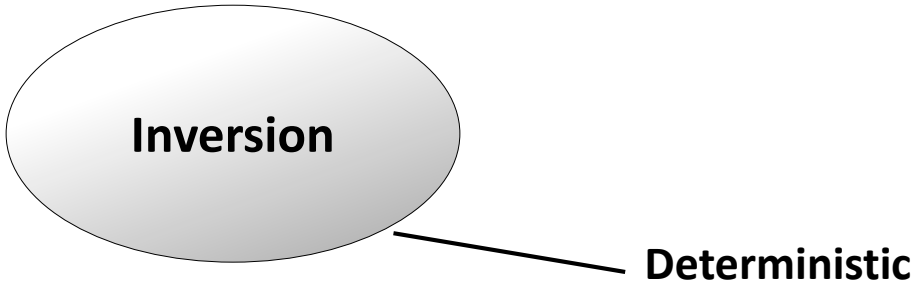


Constrained optimization:

- Function F(x,y) - topography
- Constraint g(x,y) – hike

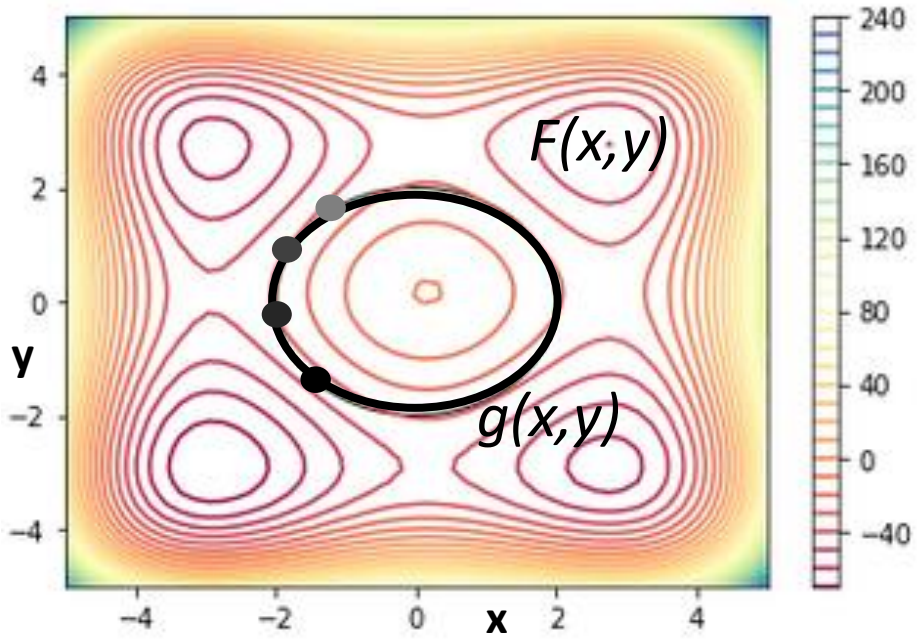


Method



Constrained optimization:

- Function F(x,y) - topography
- Constraint g(x,y) – hike



Method

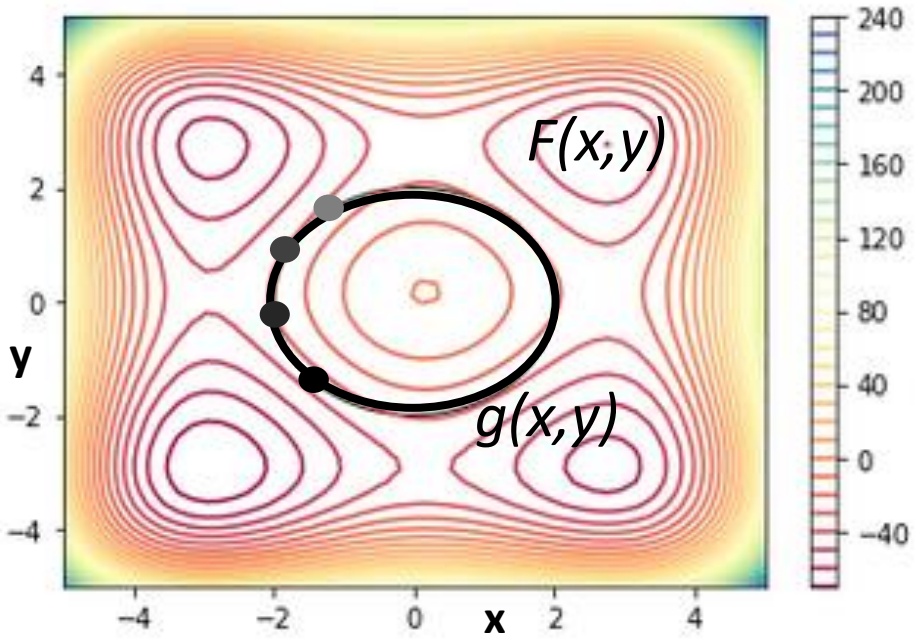
Inversion

Deterministic

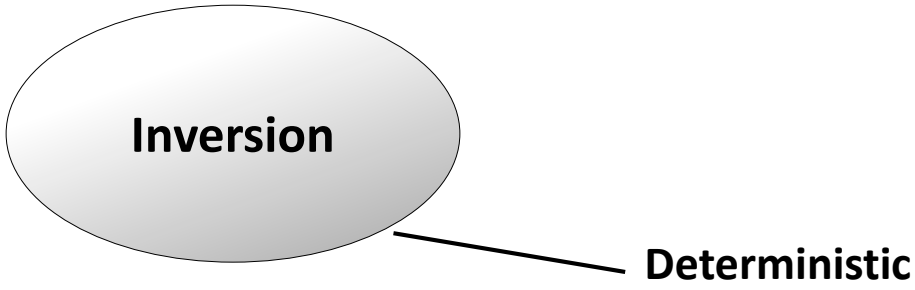
ion:

ography

ike

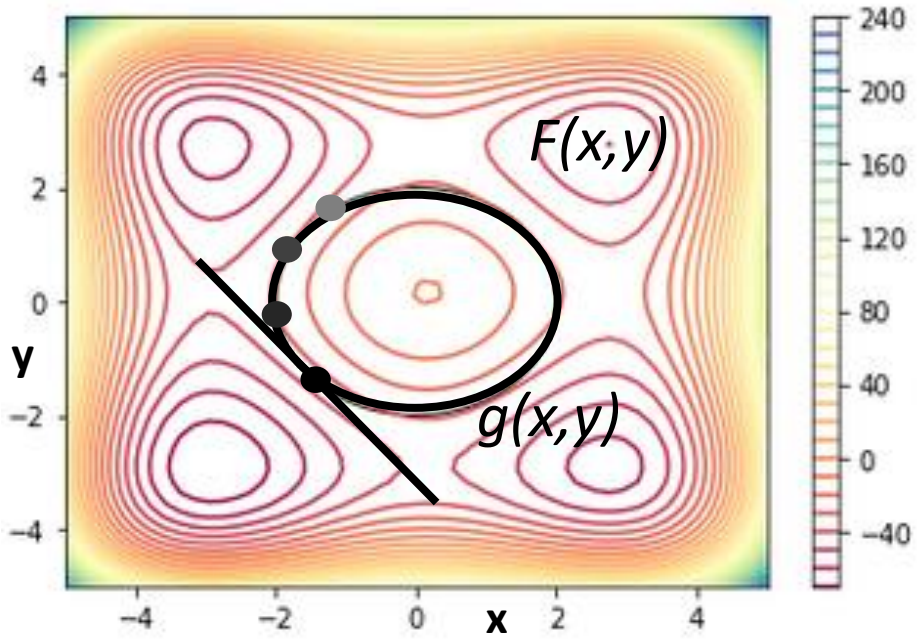


Method

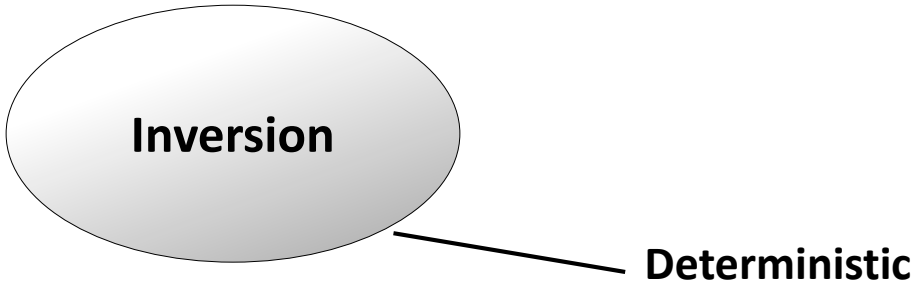


Constrained optimization:

- Function F(x,y) - topography
- Constraint g(x,y) – hike

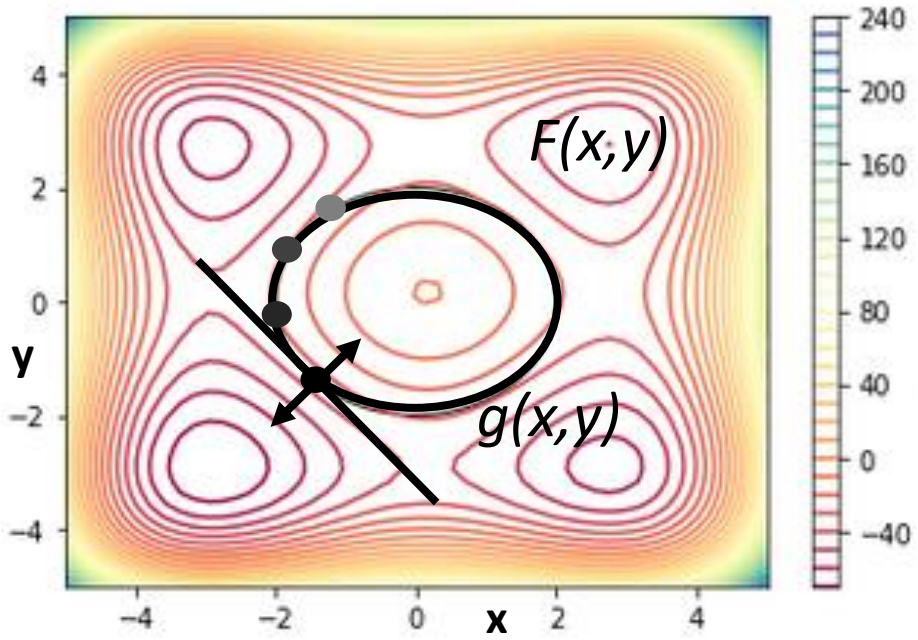


Method

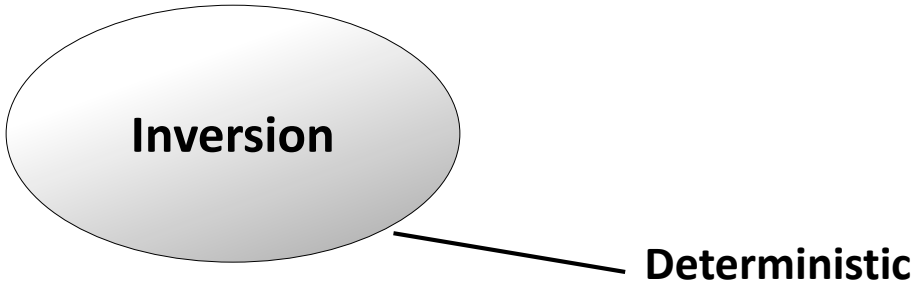


Constrained optimization:

- Function F(x,y) - topography
- Constraint g(x,y) – hike



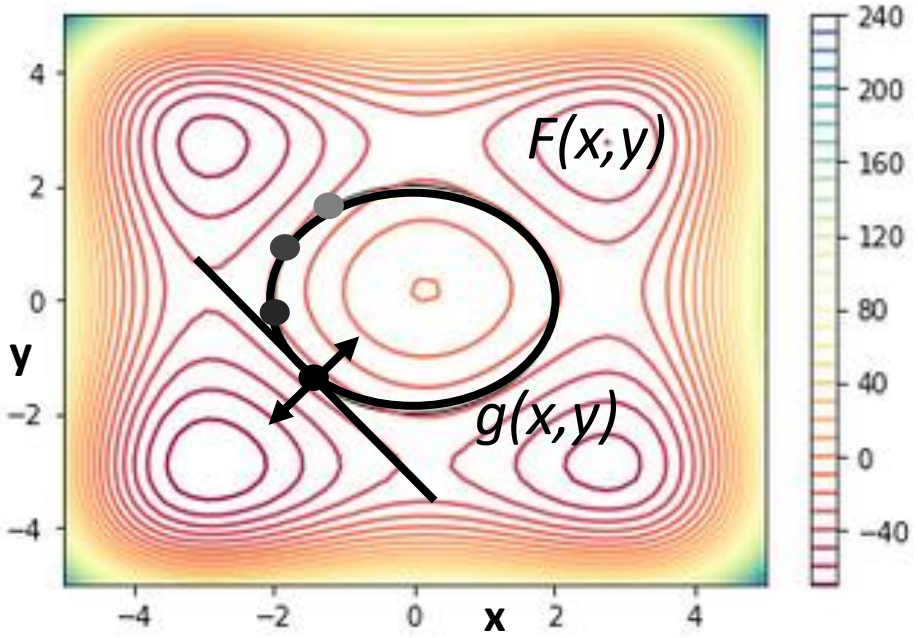
Method



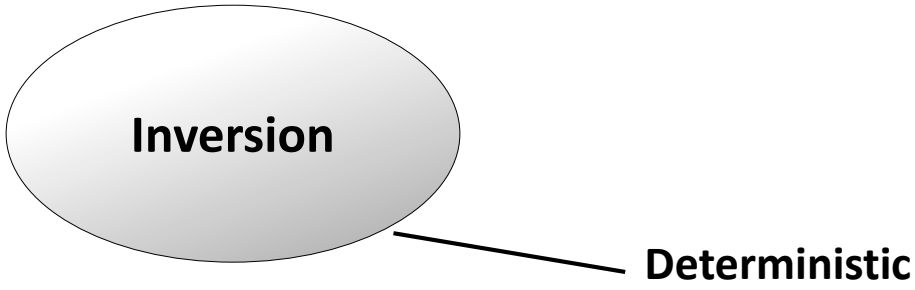
Constrained optimization:

- Function F(x,y) - topography
- Constraint g(x,y) – hike

→ **Find common gradient along constraint**



Method



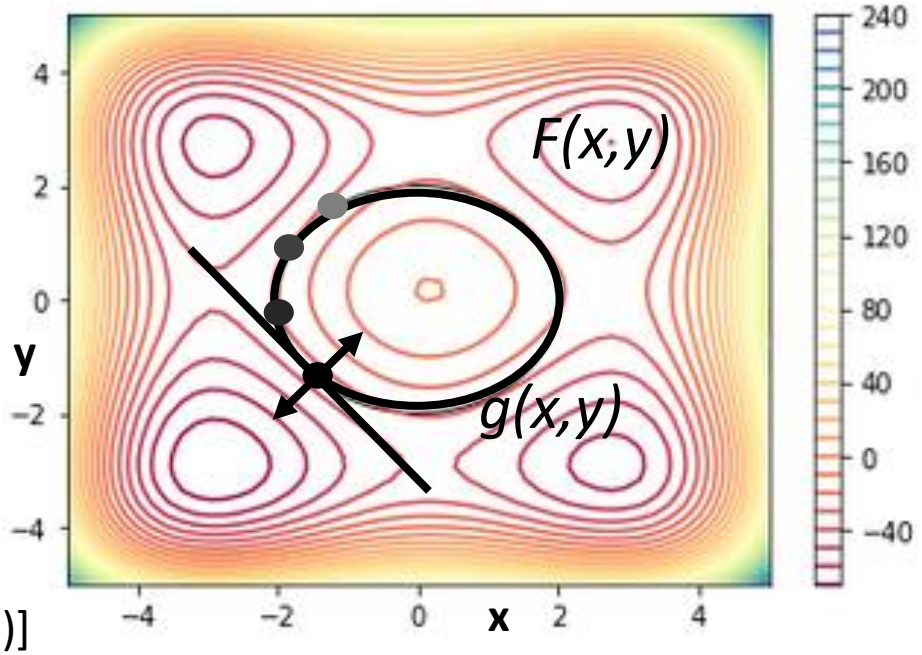
Constrained optimization:

$$\mathcal{L}(\mathbf{x}, \mathbf{y}, \lambda) = F(\mathbf{x}, \mathbf{y}) + \lambda g(\mathbf{x}, \mathbf{y})$$

$$\nabla \mathcal{L}(\mathbf{x}, \mathbf{y}, \lambda) \stackrel{!}{=} 0$$

$$\nabla F(\mathbf{x}, \mathbf{y}) = \lambda \nabla g(\mathbf{x}, \mathbf{y})$$

$$g(\mathbf{x}, \mathbf{y}) = \mathbf{x}^2 + \mathbf{y}^2 - 2$$



→ Solve for x,y & λ
→ x & y = coordinates of minimum
→ [P(x,y) = lowest point → λ = d(P)/d(Radius)]

Adjoint method
(PDE – constrained optimization)

I) Field inversion

II) Vector (p) inversion

Field

Objective function (no regularization):

$$F(\mathbf{x}, \mathbf{x}(\mathbf{p})) \qquad \mathbf{x} = (\mathbf{u}, p)^T \tag{1}$$

$$F = \frac{1}{2} (\mathbf{u} - \mathbf{u}_{obs})^T (\mathbf{u} - \mathbf{u}_{obs}) \tag{2}$$

(3)

(4)

(5)

(6)

Field

Objective function (no regularization):

$$F(\mathbf{x}, \mathbf{x}(\mathbf{p})) \qquad \mathbf{x} = (\mathbf{u}, p)^T \tag{1}$$

$$F = \frac{1}{2} (\mathbf{u} - \mathbf{u}_{obs})^T (\mathbf{u} - \mathbf{u}_{obs}) \tag{2}$$

Stokes (inversion constraint, linear, no advection):

$$\nabla \cdot \sigma = (0, \rho g)^T \qquad \nabla \cdot \mathbf{u} = 0 \tag{3}$$

$$\sigma = 2\eta\dot{\epsilon}(\mathbf{u}) - pI \qquad \dot{\epsilon}(u) = \frac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^T) \tag{4}$$

(5)

(6)

Field

Objective function (no regularization):

$$F(\mathbf{x}, \mathbf{x}(\mathbf{p})) \qquad \mathbf{x} = (\mathbf{u}, p)^T \tag{1}$$

$$F = \frac{1}{2} (\mathbf{u} - \mathbf{u}_{obs})^T (\mathbf{u} - \mathbf{u}_{obs}) \tag{2}$$

Stokes (inversion constraint, linear, no advection):

$$\nabla \cdot \sigma = (0, \rho g)^T \qquad \nabla \cdot \mathbf{u} = 0 \tag{3}$$

$$\sigma = 2\eta\dot{\epsilon}(\mathbf{u}) - pI \qquad \dot{\epsilon}(u) = \frac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^T) \tag{4}$$

Lagrangian (constrained optimization):

$$\mathcal{L}(\mathbf{u}, \mathbf{v}, p, q, \mathbf{p}) = F + (v, q) \cdot FP \tag{5}$$

$$\nabla\mathcal{L}(\mathbf{u}, \mathbf{v}, p, q, \mathbf{p}) \stackrel{!}{=} \mathbf{0} \tag{6}$$

→ **Find critical points of this function**

Field

Objective function (no regularization):

$F($

$F =$

Stoke

$\nabla \cdot$

Lagra

$\mathcal{L}$

$\nabla \mathcal{L}($

I. FP multiplied by some Lagrange multipliers (shape functions) is the weak form (FEM) of the FP

II. FP linear – (often) self- adjoint

III. If FP nonlinear – additional terms appear (e.g. viscosity derivative) – like deriving Jacobian

IV. Gradient of parameter at every node – ‚field' inversion

V. Independent of discretization

VI. Can be derived for any variable that occurs in the equation, e.g. principal stress directions:

$$\phi_{\boldsymbol{\sigma}} = \frac{1}{2} tan^{-1} \left(\frac{2\dot{\epsilon}_{xy}}{\dot{\epsilon}_{xx} - \dot{\epsilon}_{yy}} \right)$$

Reuber et al. (in prep)

Vector

Approach implemented in LaMEM (equal result):

1) **Objective function** (no regularization):

$$F(\mathbf{x}, \mathbf{x}(\mathbf{p})) \qquad \mathbf{x} = (\mathbf{u}, p)^T \tag{1}$$

$$F = \frac{1}{2} (\mathbf{u} - \mathbf{u}_{obs})^T (\mathbf{u} - \mathbf{u}_{obs}) \tag{2}$$

(3)

(4)

Vector

Approach implemented in LaMEM (equal result):

1) **Objective function** (no regularization):

$$F(\mathbf{x}, \mathbf{x}(\mathbf{p})) \qquad \mathbf{x} = (\mathbf{u}, p)^T \tag{1}$$

$$F = \frac{1}{2} (\mathbf{u} - \mathbf{u}_{obs})^T (\mathbf{u} - \mathbf{u}_{obs}) \tag{2}$$

2) **Objective function derivative** (final gradient sought):

$$G = \frac{dF}{dp} = \frac{\partial F}{\partial p} + \frac{\partial F}{\partial x} \frac{dx}{dp}$$

First term = 0 (!= 0 if regularization) (3)

(4)

Vector

Approach implemented in LaMEM (equal result):

1) **Objective function** (no regularization):

$$F(\mathbf{x}, \mathbf{x}(\mathbf{p})) \qquad \mathbf{x} = (\mathbf{u}, p)^T \tag{1}$$

$$F = \frac{1}{2} (\mathbf{u} - \mathbf{u}_{obs})^T (\mathbf{u} - \mathbf{u}_{obs}) \tag{2}$$

2) **Objective function derivative** (final gradient sought):

$$G = \frac{dF}{dp} = \frac{\partial F}{\partial p} + \frac{\partial F}{\partial x}\frac{dx}{dp}$$

First term = 0 (!= 0 if regularization) (3)

2) **Residual derivative** (R = 0):

$$\frac{dR}{dp} = \frac{\partial R}{\partial p} + \frac{\partial R}{\partial x}\frac{dx}{dp} \tag{4}$$

Jacobian

Vector

3) **Substitute**:

$$\frac{dx}{dp} = -J^{-1}\frac{\partial R}{\partial p} \tag{5}$$

$$G = \frac{\partial F}{\partial x}\frac{dx}{dp} = -\frac{\partial F}{\partial x}J^{-1}\frac{\partial R}{\partial p} \tag{6}$$

(7)

(8)

Vector

3) **Substitute**:

$$\frac{dx}{dp} = -J^{-1}\frac{\partial R}{\partial p} \tag{5}$$

$$G = \frac{\partial F}{\partial x}\frac{dx}{dp} = -\frac{\partial F}{\partial x}J^{-1}\frac{\partial R}{\partial p} \tag{6}$$

4) **Final**:

$$\psi = J^{-T}\left(\frac{\partial F}{\partial x}\right)^{T} \tag{7}$$

$$G = -\psi^{T}\frac{\partial R}{\partial p} \tag{8}$$

Vector

3) **Substitute**:

$$\frac{dx}{dp} = -J^{-1}\frac{\partial R}{\partial p} \tag{5}$$

$$G = \frac{\partial F}{\partial x}\frac{dx}{dp} = -\frac{\partial F}{\partial x}J^{-1}\frac{\partial R}{\partial p} \tag{6}$$

4) **Final**:

$$\psi = J^{-T}\left(\frac{\partial F}{\partial x}\right)^{T} \tag{7}$$

$$G = -\psi^{T}\frac{\partial R}{\partial p} \tag{8}$$

Note: *Two options to compute gradients with the adjoint method:*

I) Converge forward problem – compute missing partial derivatives – reuse FP Jacobian - ***DtO***

II) Write two codes – one for FP and one for AP – combine solution to compute Gradient - ***OtD***

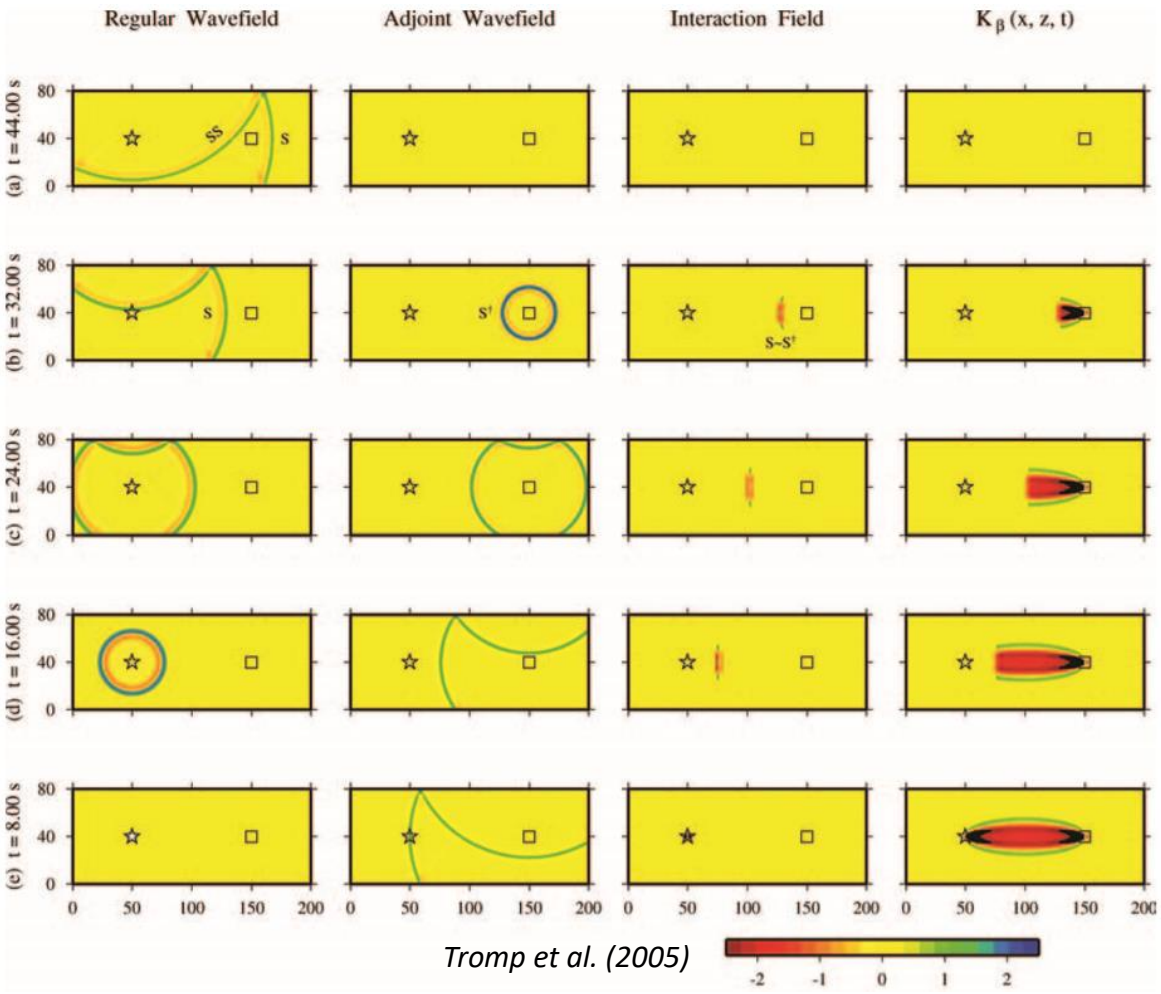
What are these gradients good for?

I) *Obviously for gradient based inversion*

II) **Sensitivity kernels in geodynamics – resolution proxy**

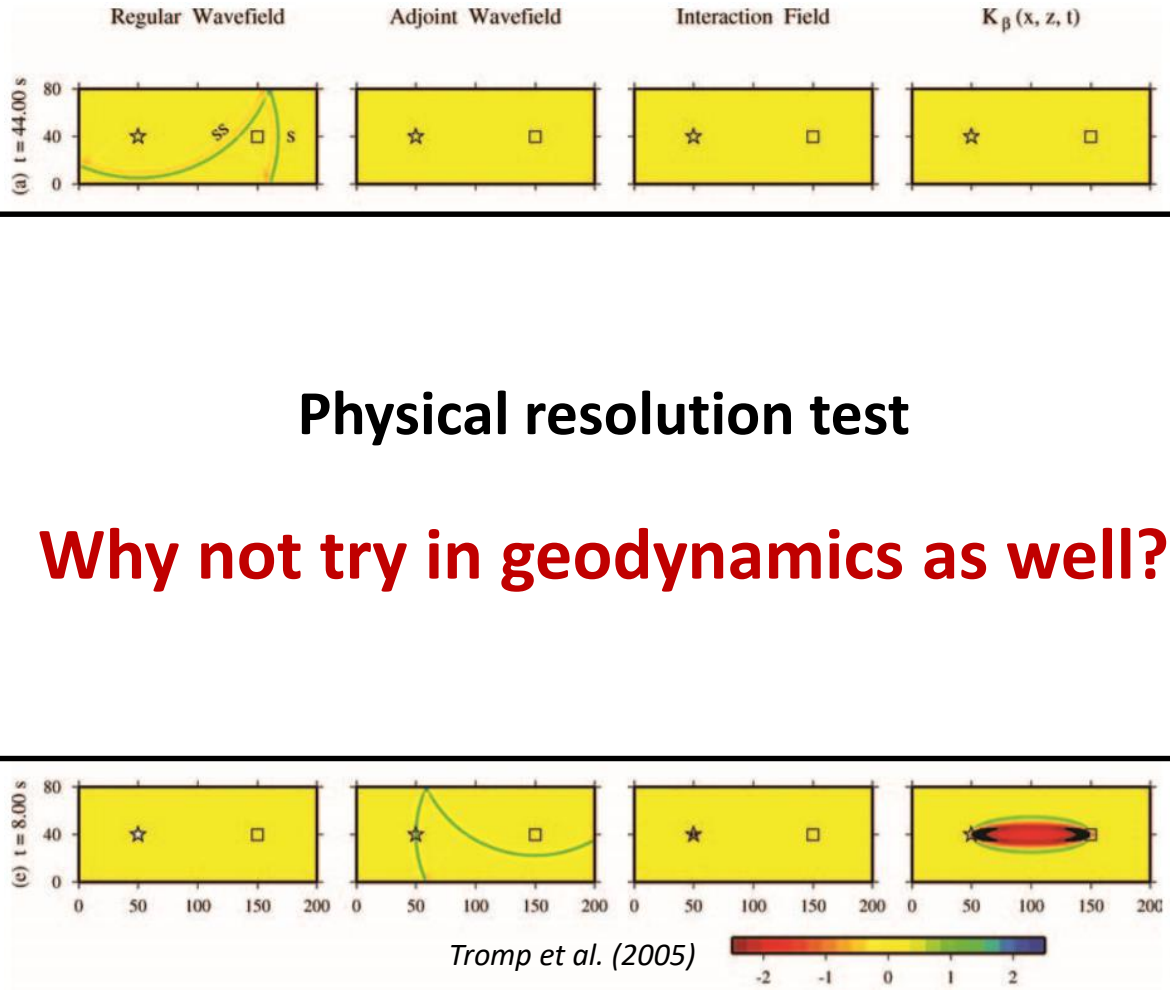
III) **Automatic derivation of scaling laws**

Sensitivity



Tromp et al. (2005)

Sensitivity



Physical resolution test

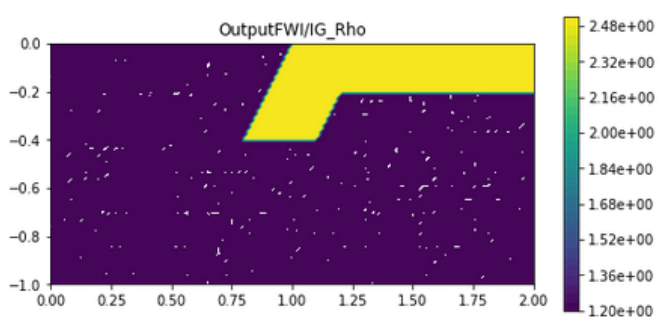
Why not try in geodynamics as well?

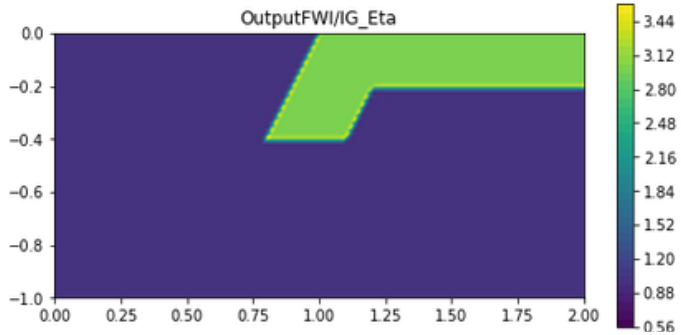
Tromp et al. (2005)

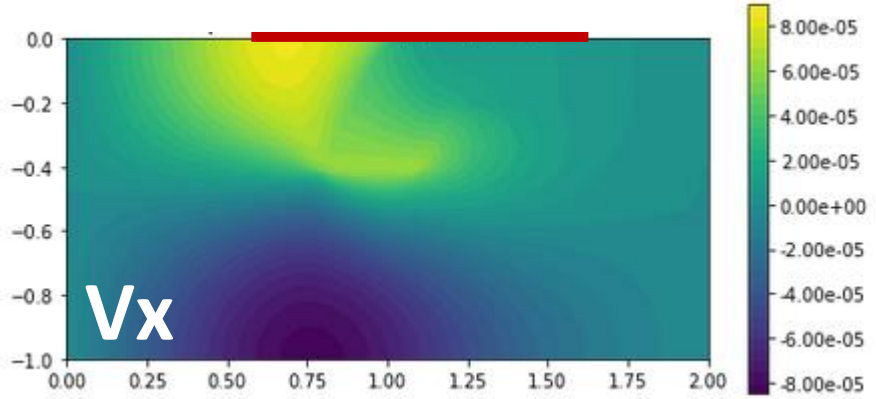
Geodynamic sensitivity kernels

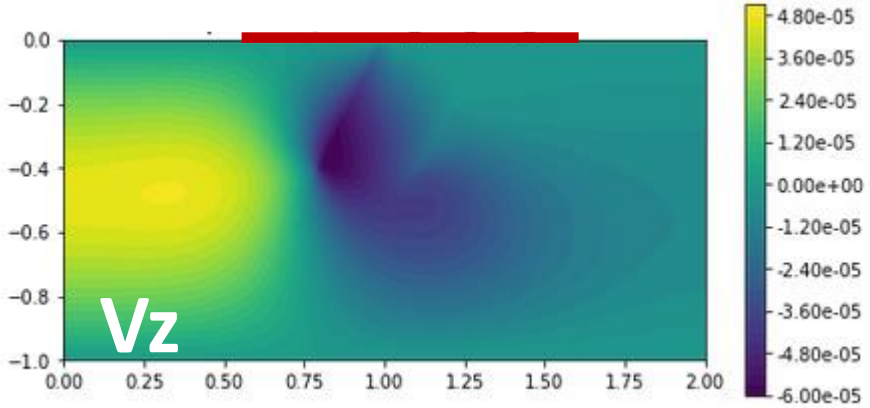
→ **What are the velocities at the surface most sensitive to?**

- → Compute gradient as usual but use: $F = \mathbf{u}$
- → Sensitivity of solution to ρ + η at every node
- → E.g. resolution test in seismology





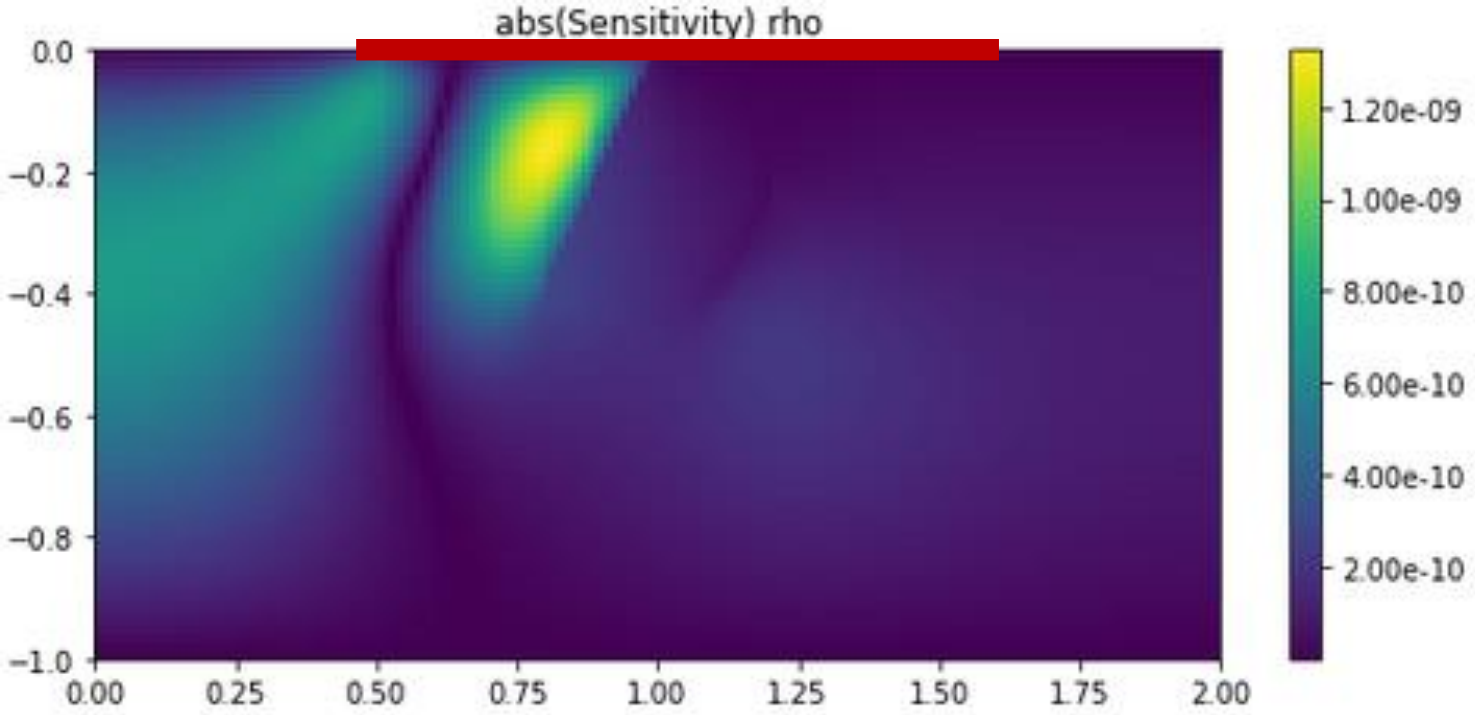




Sensitivity

→ **What are the velocities at the surface most sensitive to?**

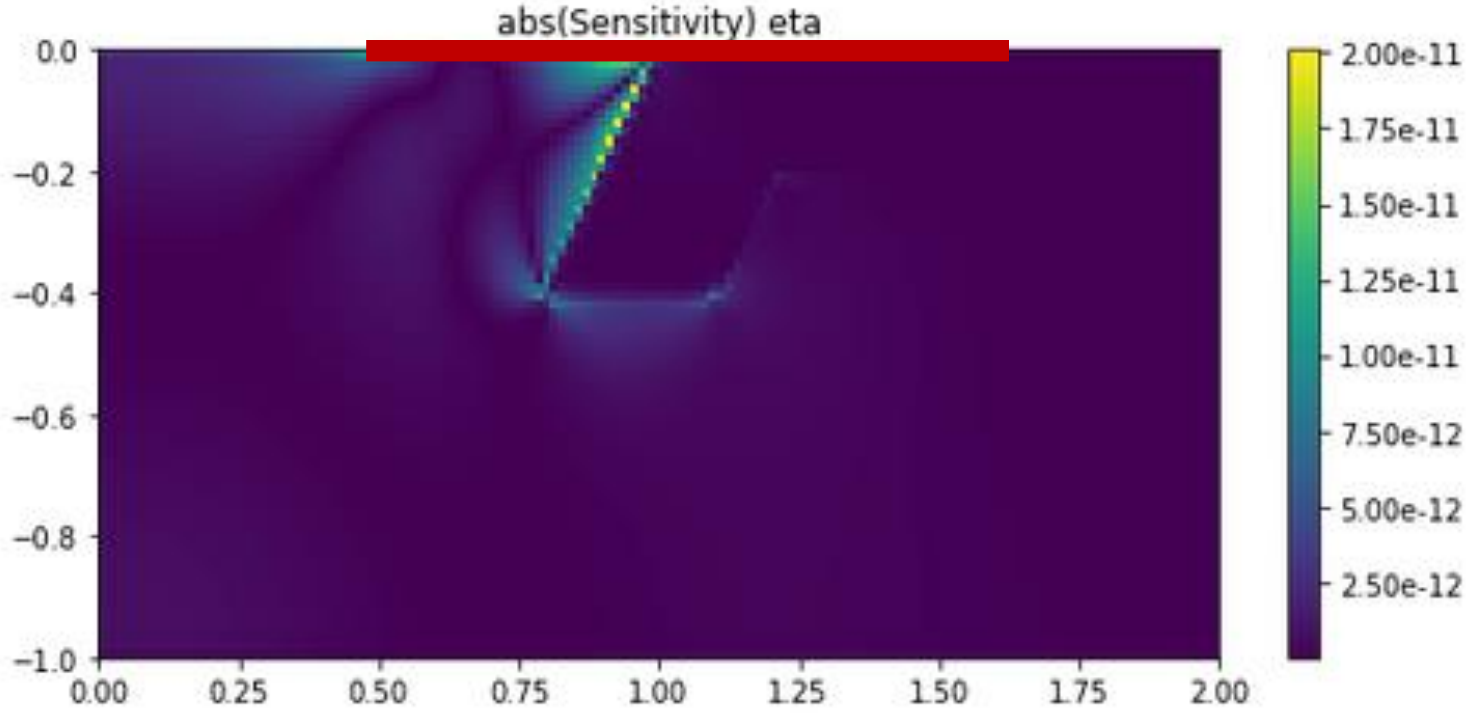
- → Compute gradient as usual but use: $F = \mathbf{u}$
- → Sensitivity of solution to $\rho + \eta$ at every node
- → E.g. resolution test in seismology



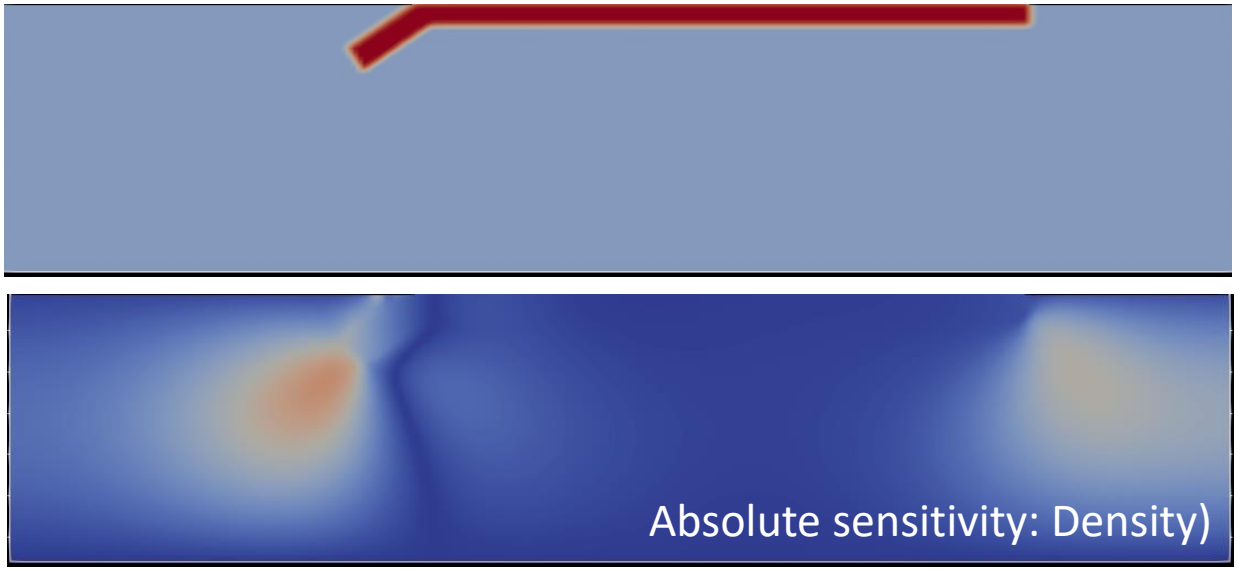
Sensitivity

→ **What are the velocities at the surface most sensitive to?**

- → Compute gradient as usual but use: $F = \mathbf{u}$
- → Sensitivity of solution to ρ + η at every node
- → E.g. resolution test in seismology

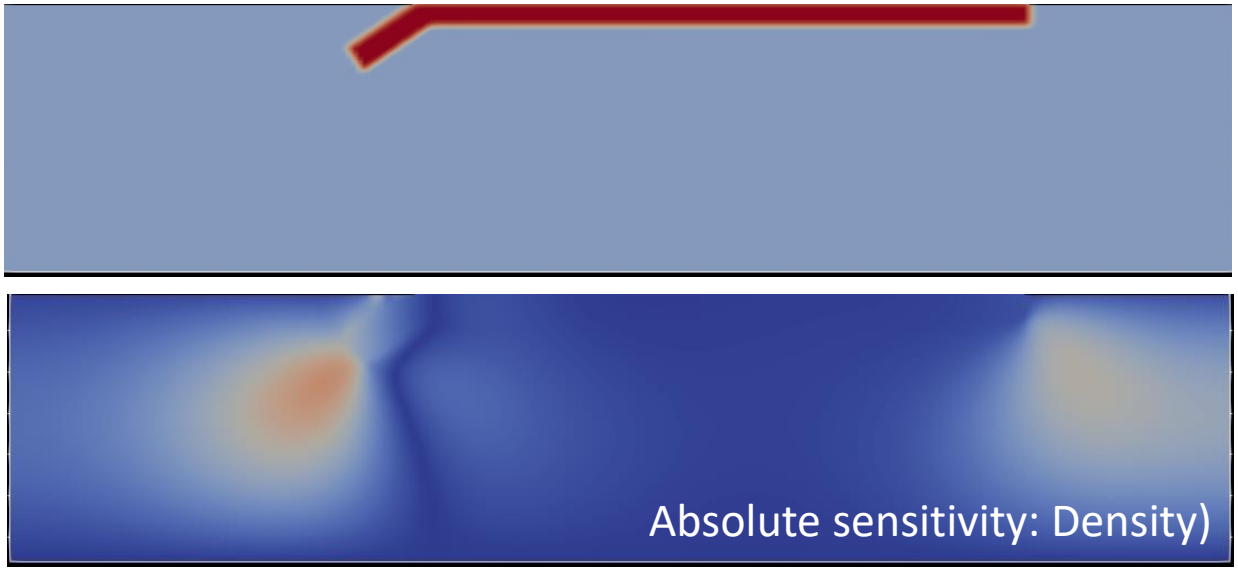


Sensitivity

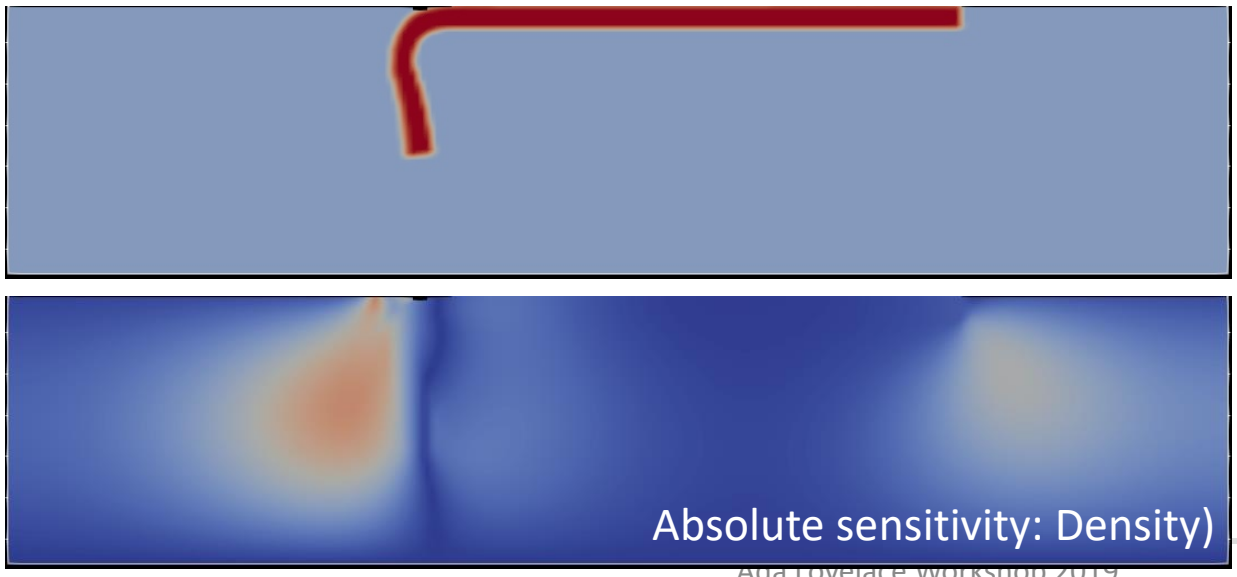


Timestep 1

Sensitivity



Timestep 1

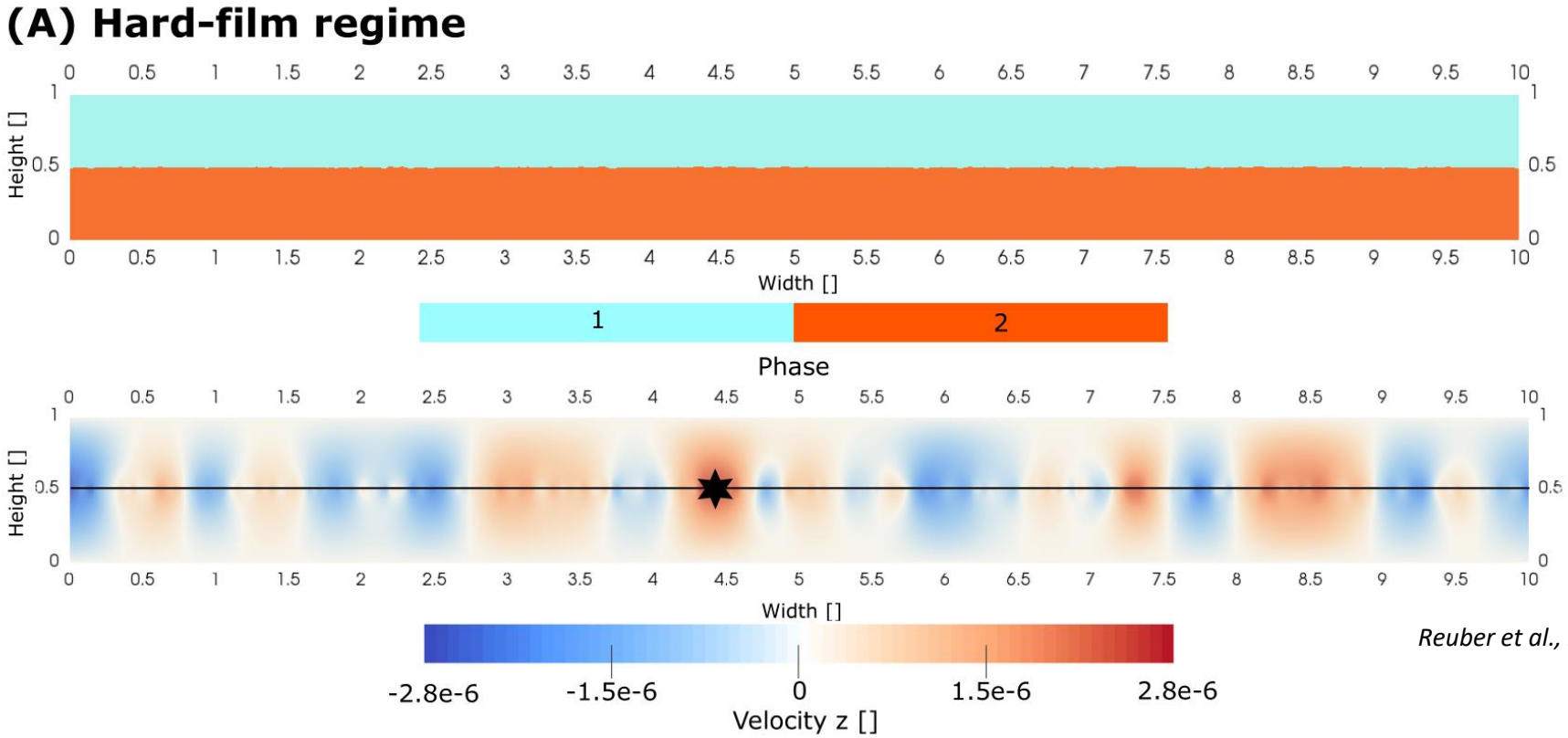


Timestep 50

Cheap!
Can be done every timestep

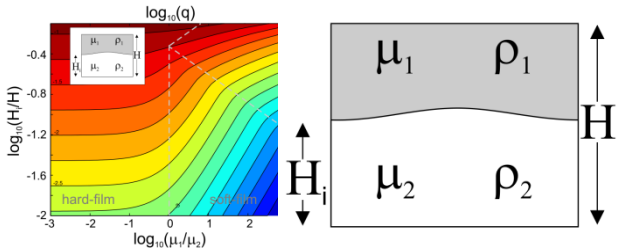
Automatic derivation of scaling laws

Scaling law

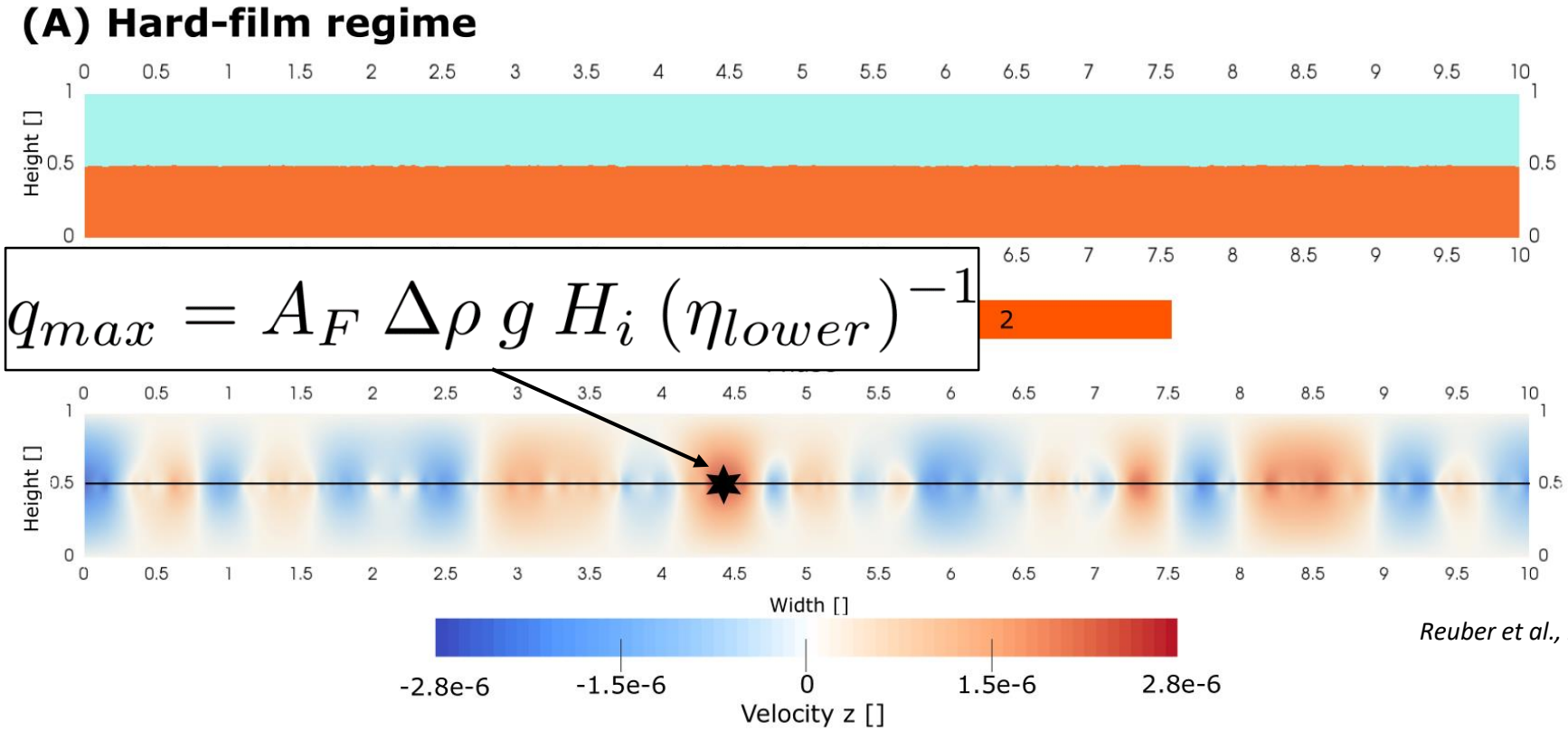


Reuber et al., 2017

- Nondimensional 2 layer Rayleigh-Taylor instability
- Random perturbation at interface

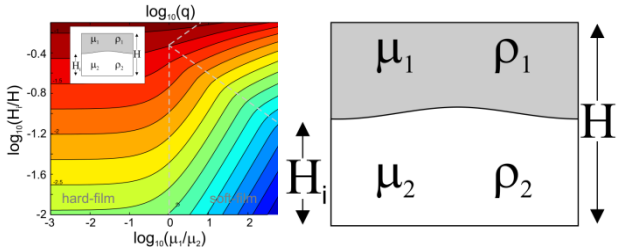


Scaling law

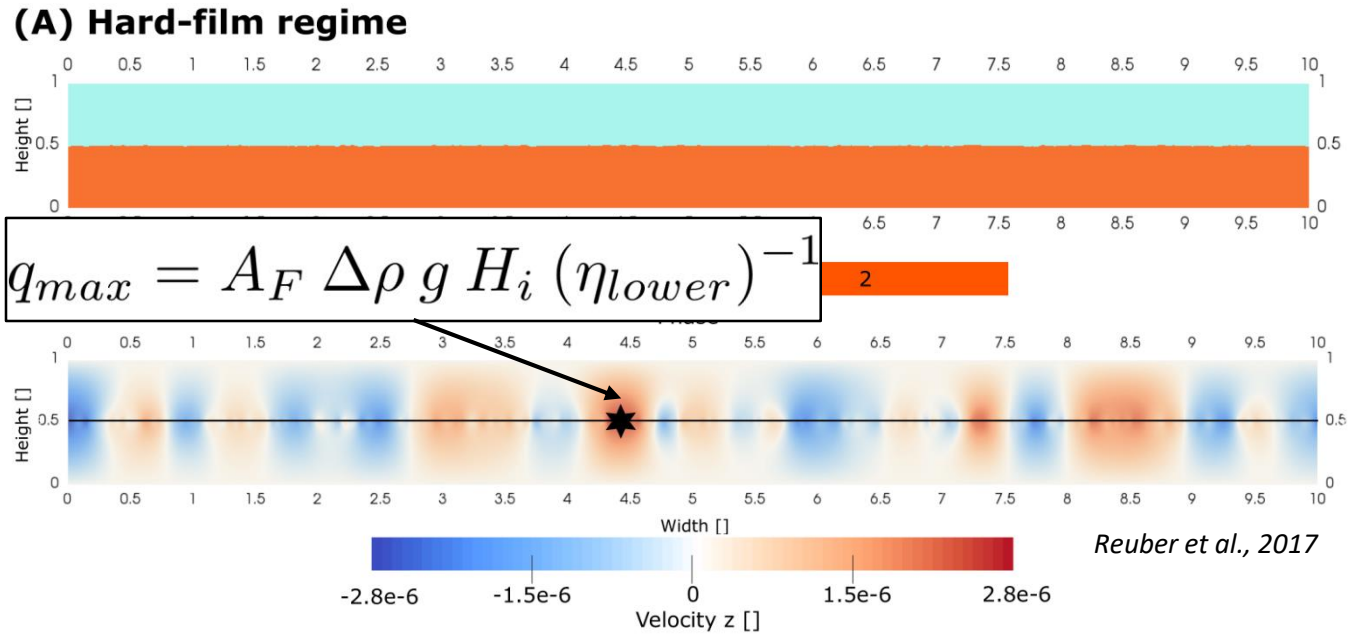


Reuber et al., 2017

- Nondimensional 2 layer Rayleigh-Taylor instability
- Random perturbation at interface



Scaling law

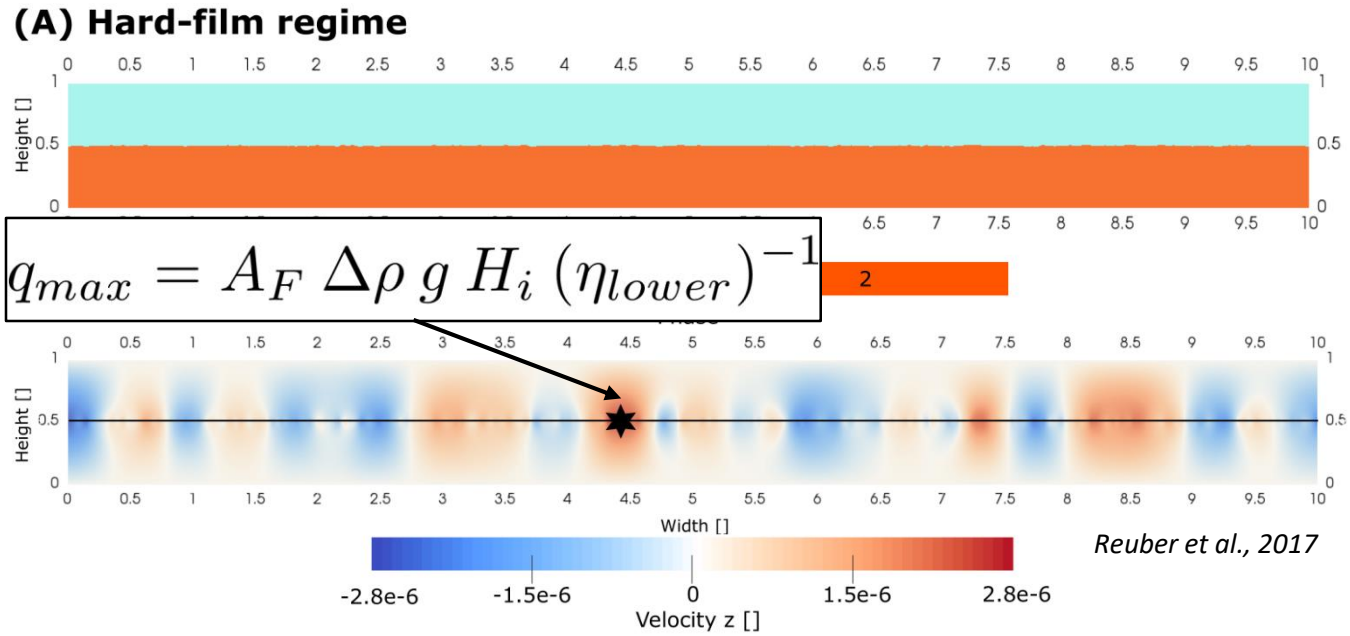


Reuber et al., 2017

$F = x$

No cost function (as before)

Scaling law



Reuber et al., 2017

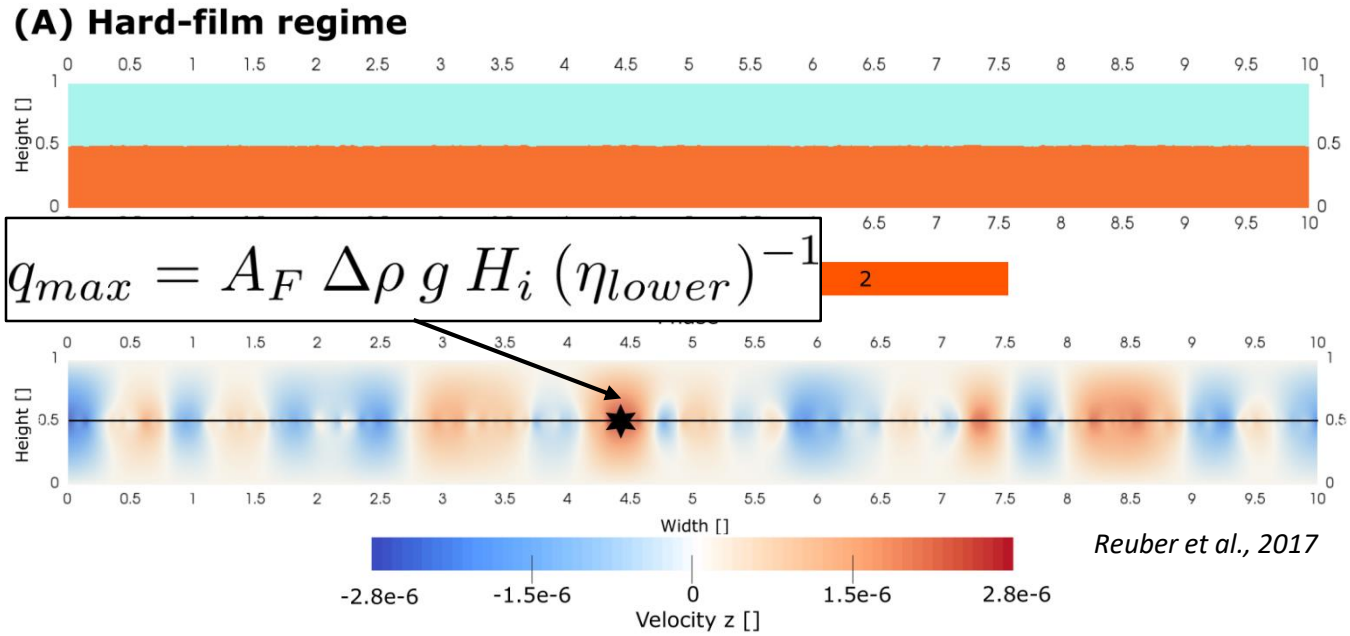
$$F = x$$

$$F \approx F^* = A_F \, p_1^{b_1} \, p_2^{b_2} \ldots p_n^{b_n}$$

No cost function!

General (multiplicative) scaling law

Scaling law



Reuber et al., 2017

$$F = x$$

No cost function!

$$F \approx F^* = A_F \, p_1^{b_1} \, p_2^{b_2} \, \dots \, p_n^{b_n}$$

General (multiplicative) scaling law

$$b_i = \frac{\partial F^*}{\partial p_i} \, \frac{p_i}{F^*}$$

Exponent

$$A_F = \frac{F}{p_1^{b_1} \, p_2^{b_2} \, \dots \, p_n^{b_n}}$$

Prefactor

Scaling law

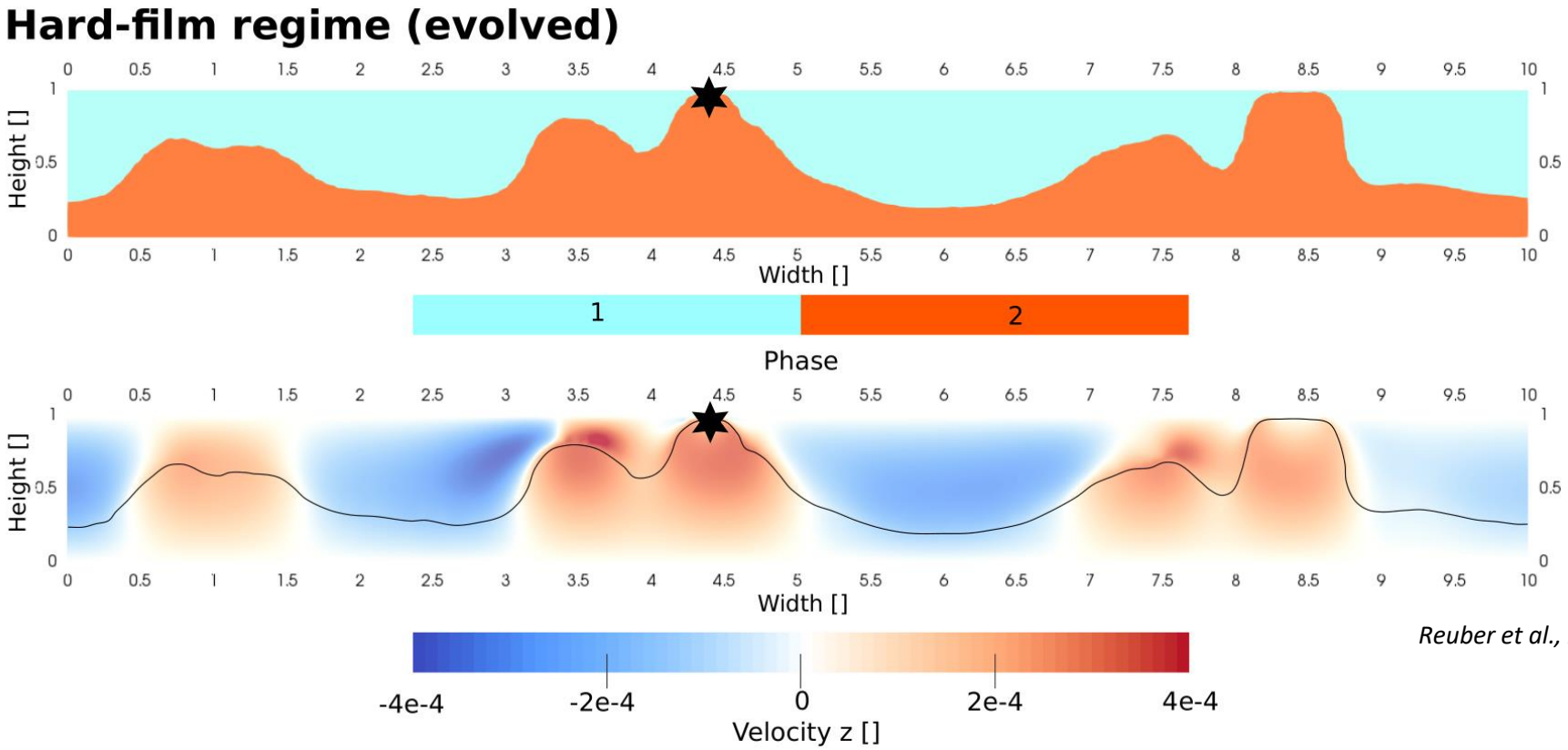
$$q_{max} = A_F \, \Delta\rho \, g \, H_i \, (\eta_{lower})^{-1}$$

Maximum growth rate value $q_{max} = 9.5 \times 10^{-4}$.

Parameter	Index	Value	Scaling exponent adjoint / analytical	dF/dp_i	$\partial q_{max}/\partial p_i$
$\Delta\rho$	1	1	1.02 / 1	9.53×10^{-4}	8×10^{-4}
η_{upper}	2	1	-3.38×10^{-2} / 0	-3.2×10^{-5}	0
η_{lower}	3	100	−0.97 / -1	-9.21×10^{-6}	-8×10^{-6}
H_i	4	0.5	1 / 1	1.9×10^{-3}	1.6×10^{-3}

→ Exact reproduction of analytical solution

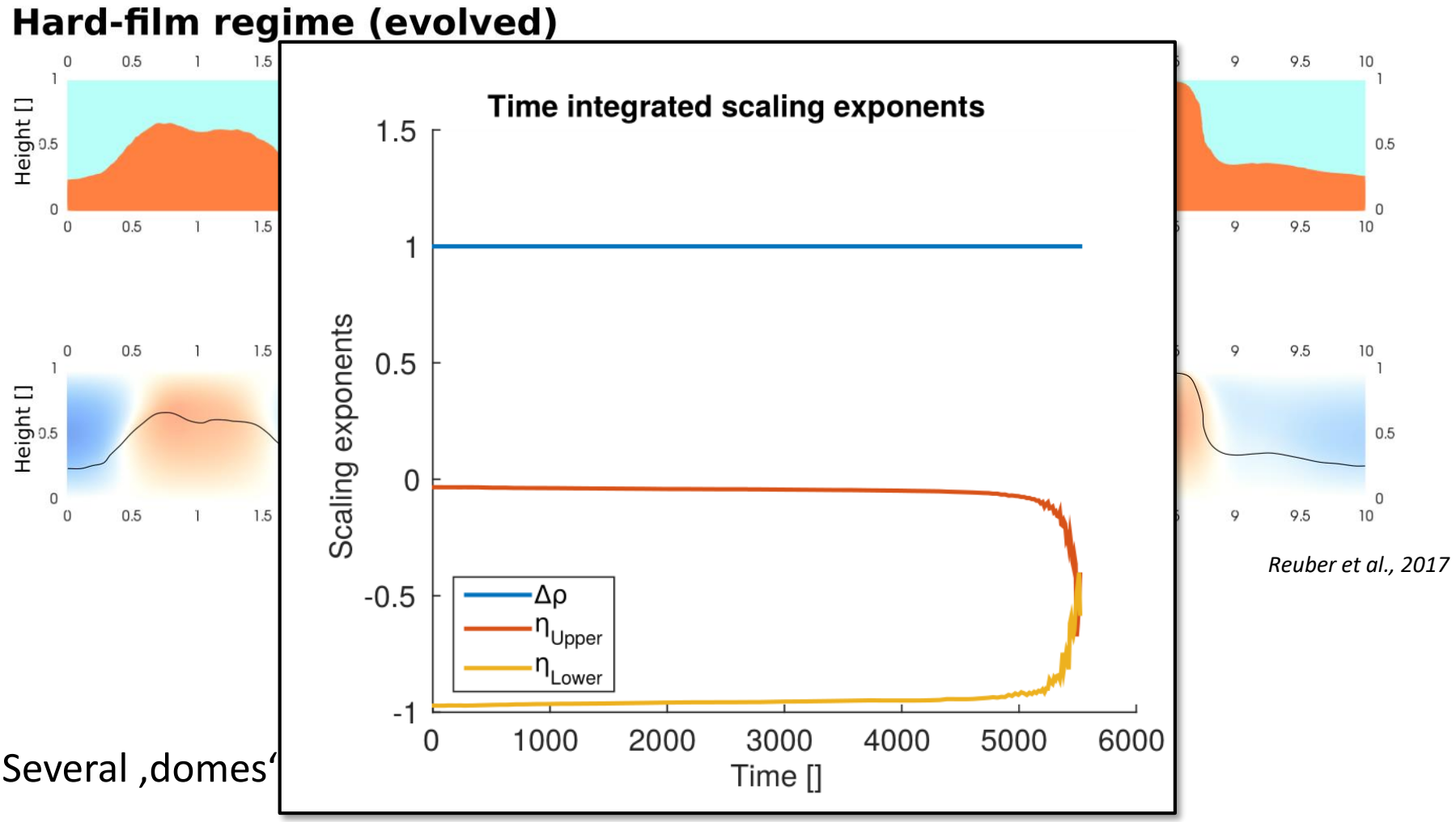
Scaling law



Reuber et al., 2017

Several ‚domes' evolve

Scaling law



Several ‚domes'

Reuber et al., 2017

Conclusions

- **Adjoint method very efficient:**
  - independent of number of parameters (‚field inversion‘)
  - Can be derived for any parameter for any PDE

Conclusions

- **Adjoint method very efficient:**
  - independent of number of parameters (‚field inversion‘)
  - Can be derived for any parameter for any PDE

- **Statistic methods:**
  - Full knowledge (topology) of cost function
  - (Almost) No additional implementation required
  - Numerical cost scales with number of parameters

Conclusions

- **Adjoint method very efficient:**
  - independent of number of parameters (‚field inversion‘)
  - Can be derived for any parameter for any PDE

- **Statistic methods:**
  - Full knowledge (topology) of cost function
  - (Almost) No additional implementation required
  - Numerical cost scales with number of parameters

- **Deterministic methods:**
  - Gradient information
    → Reveals sensitivity of e.g. velocity on density – resolution test
    → Scaling law computable
    → Time integration
    → Hessian – link to statistic methods