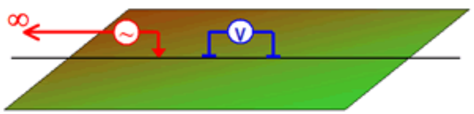
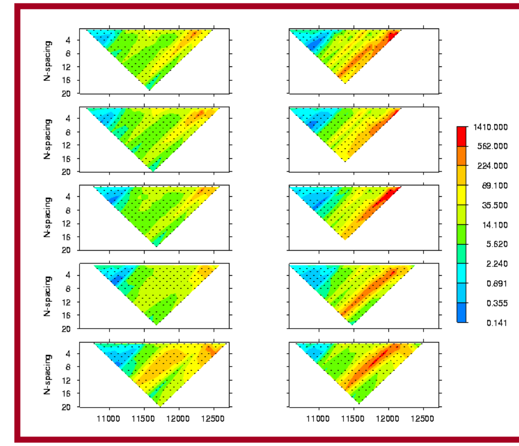
Inverse Theory

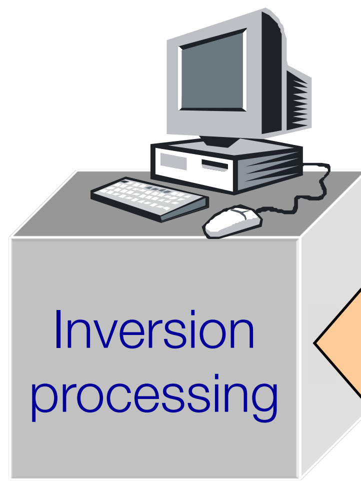
Inversion

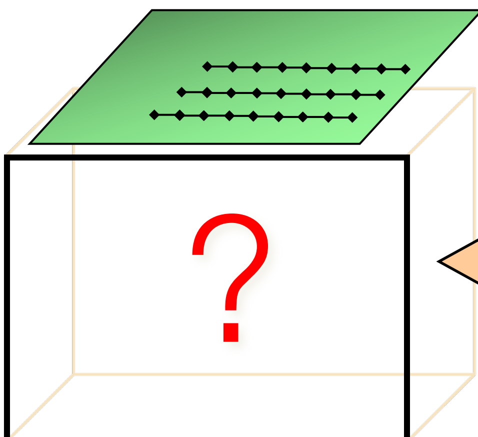


Measurements over the Earth are data.

Data



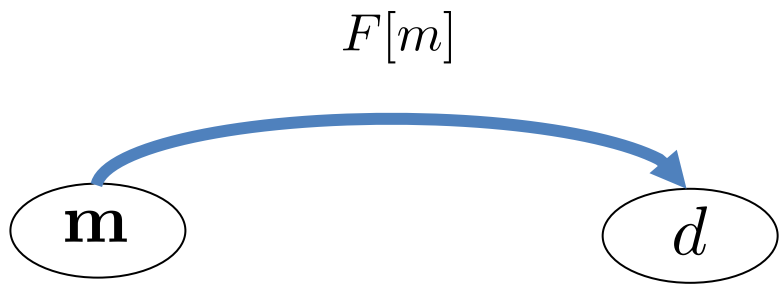




Model

Inversion estimates Earth models based upon data and prior knowledge.

Forward problem



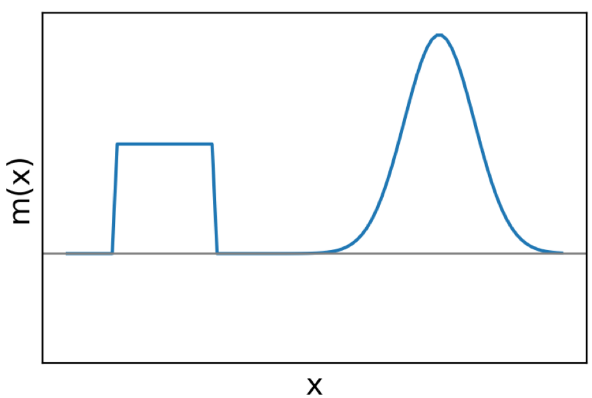
- Symbolically: $F[m] = d$
  - $F[m]$: forward modelling operator
  - m : physical property
  - d : simulated data

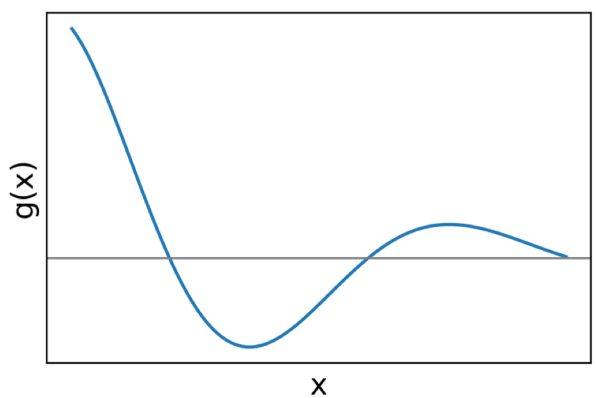
- Two cases for mapping:
  - Linear: $F[c_1 m + c_2 m] = c_1 F[m] + c_2 F[m]$
  - Nonlinear: equality does not hold

Linear problem

$$d_j = \int_v g_j(x) m(x) dx$$

- d_j : j-th datum
- g_j : kernel function for j-th datum
- m : model



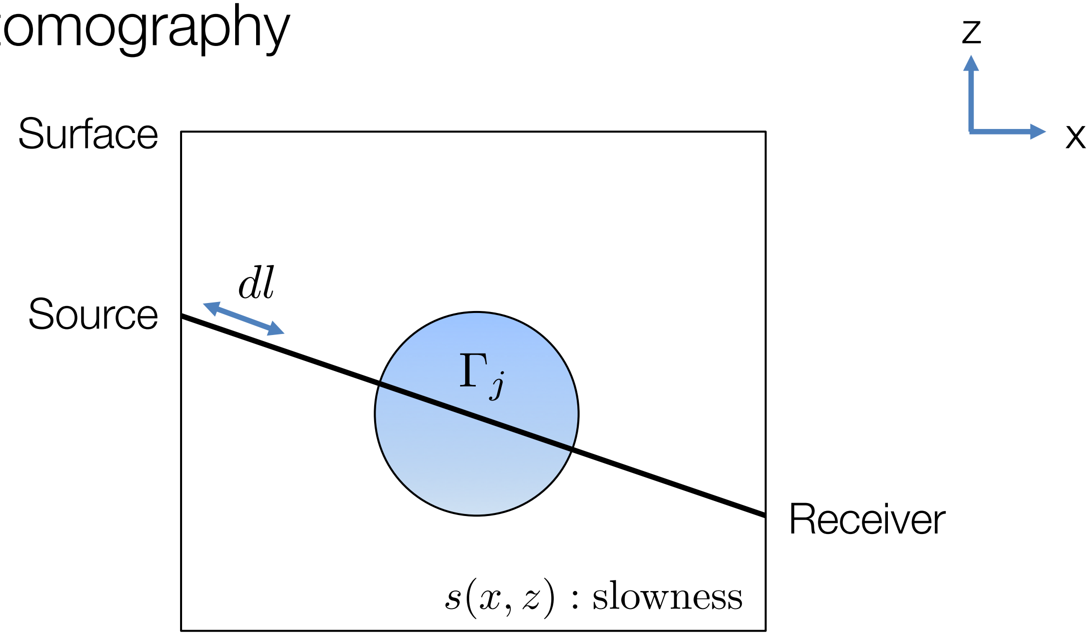


Evaluate product:

$d_j = \mathbf{g} \cdot \mathbf{m} = 4.89$

The linear problem can be in higher dimensions

- Cross well tomography



Travel time: $t_j = \int_{\Gamma_j} s(x,z)dl$

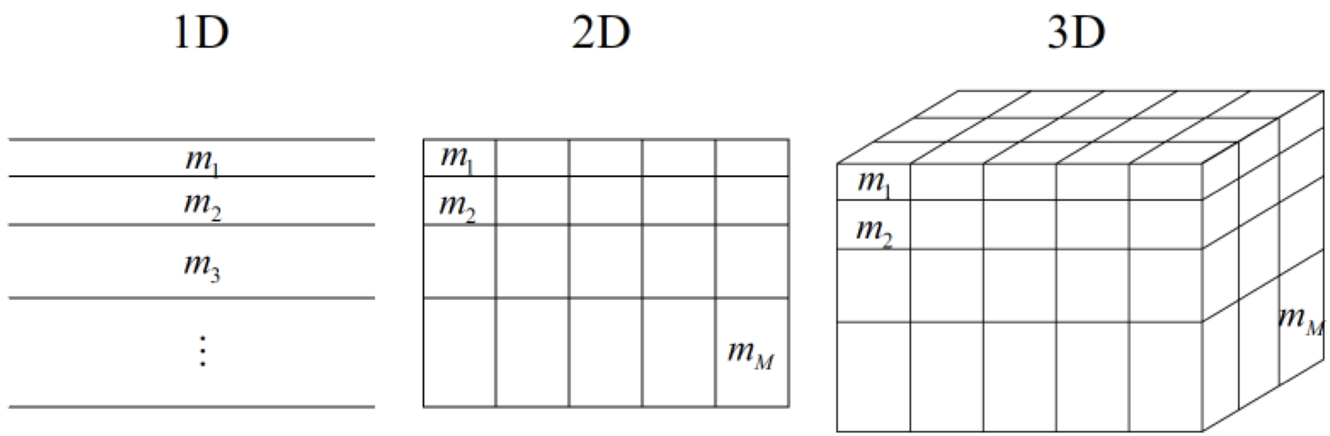
- Or magnetics

$$\mathbf{B}(\mathbf{r}) = \frac{\mu_0}{4\pi}\int_V \nabla\nabla \frac{1}{|\mathbf{r}-\mathbf{r}_o|}\cdot \kappa \mathbf{H}_0 dv, \qquad \kappa(x,y,z) \text{ is 3D susceptibility}$$

Solving the forward problem: linear

$$d_j = \int_v g_j(x) m(x) dx$$

- Discretize the earth



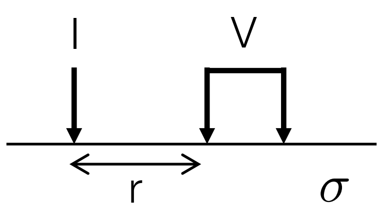
- Evaluate: $\mathbf{d} = \mathbf{G}\mathbf{m}$

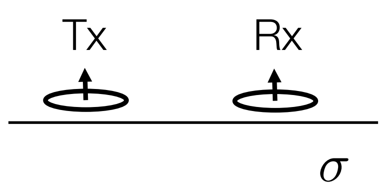
Nonlinear problem

$$F[m] = d$$

- $F[\cdot]$: Maxwell's operator
  - DC
  - Time domain
  - Frequency domain
  - 1D, 2D, 3D

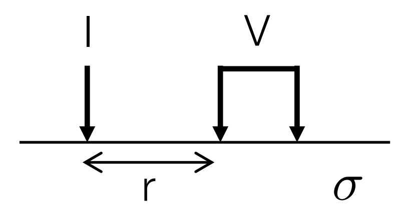
- Examples
  - DC: $\nabla \cdot \sigma \nabla V = I_0 \delta(\mathbf{r} - \mathbf{r}_s)$
  - or EM: $\nabla \times \mu^{-1} \times \mathbf{e} + \sigma \frac{\partial \mathbf{e}}{\partial t} = -\frac{\partial \mathbf{s}}{\partial t}$



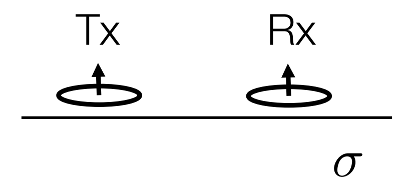


Nonlinear problem

- Examples
  - DC: $$\nabla \cdot \sigma \nabla V = I_0 \delta(\mathbf{r} - \mathbf{r}_s)$$

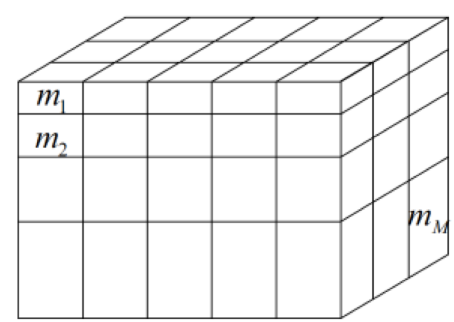


  - or EM: $$\nabla \times \mu^{-1} \times \mathbf{e} + \sigma \frac{\partial \mathbf{e}}{\partial t} = -\frac{\partial \mathbf{s}}{\partial t}$$



- Solve
  - Discretize Maxwell's equations onto a mesh
  - Solve system to find fields (e.g. $V, \ \mathbf{e}$)
  - Evaluate Datum: $d = f[V]$ or $d = f[\mathbf{e}]$

- Lot of details
  - size of cells
  - size of mesh
  - …



Linear inversion app (demo)

- It will help us
  - Develop a model
  - Consider kernels
  - Generate data

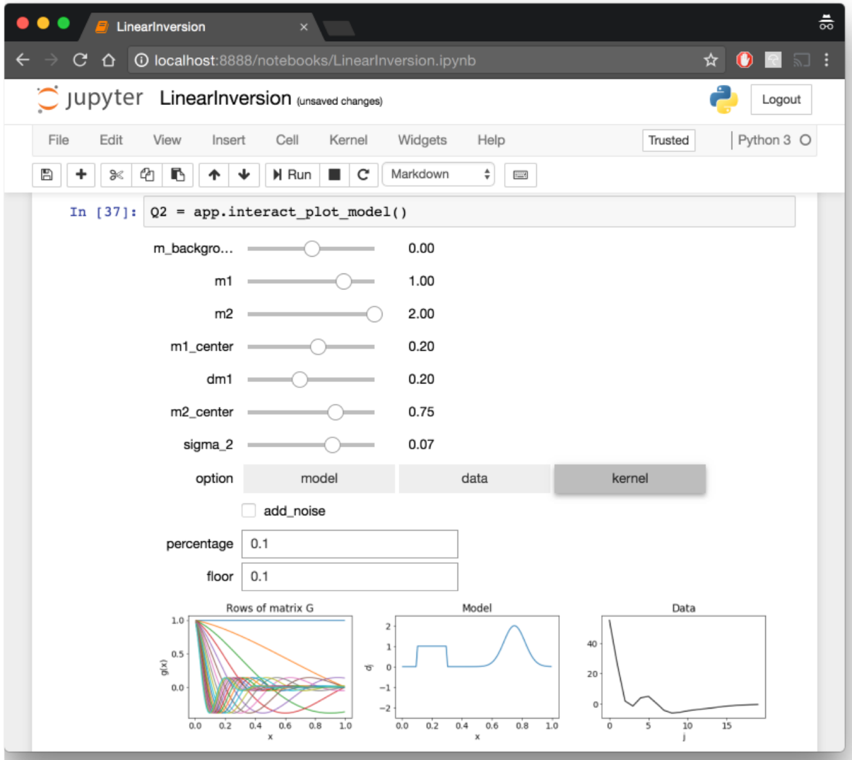
- Model: $m(x)$

- Kernels (physics):

$$g_j(x) = e^{jpx} cos(2\pi jqx)$$

- Data:

$$d_j = \int_v g_j(x) m(x) dx$$

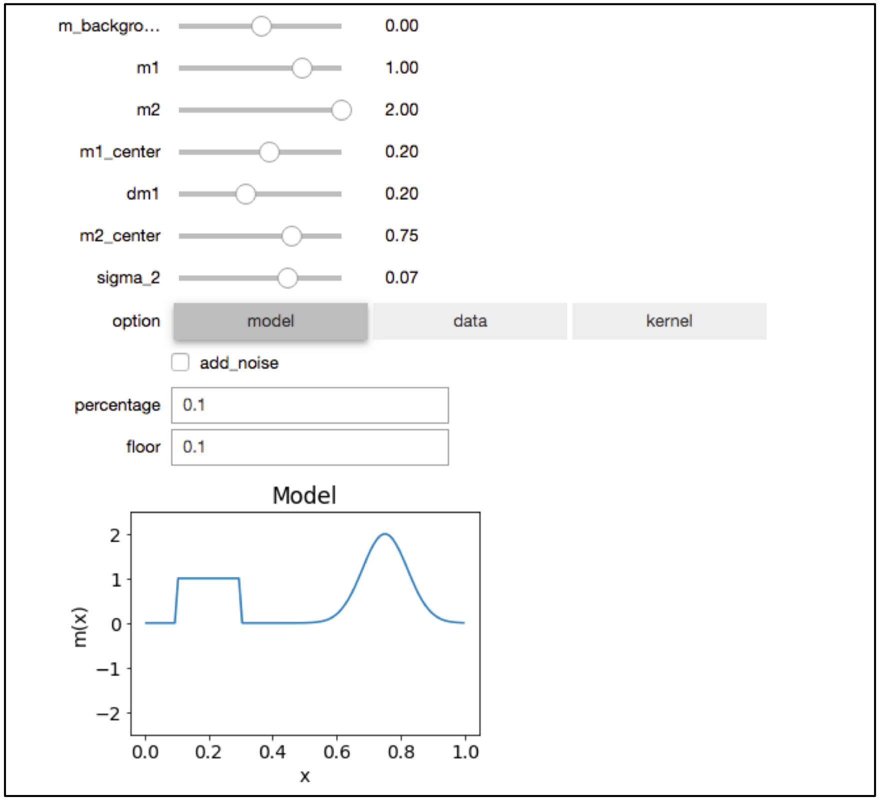


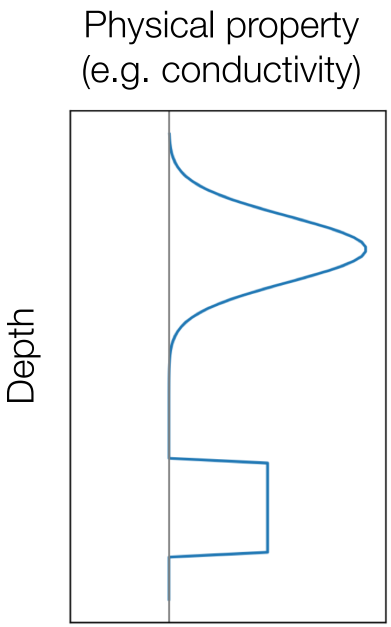
http://em.geosci.xyz/apps.html

Models with the app (demo)

- Start with 1D model
  - Conductivity changes with depth

- In the app this is here:



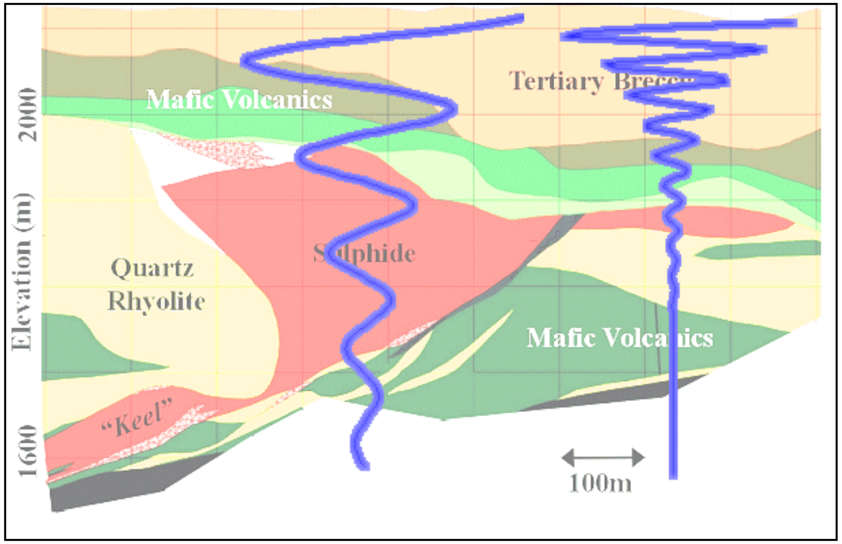


Analogy with 1D frequency domain EM

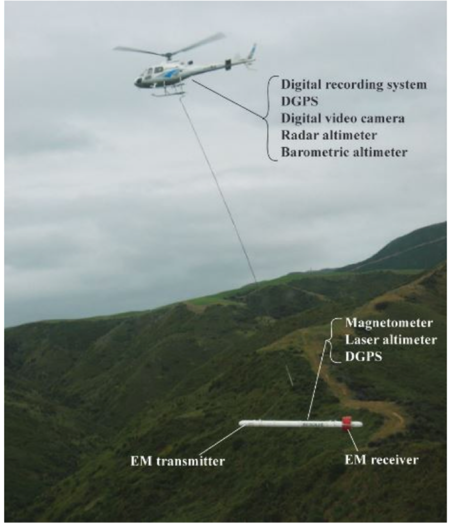
- FDEM system (Resolve)
- Signals: sinusoids at 5 frequencies
- Penetration depth depends upon frequency
- Measurements are fields from buried conductors

$$\delta = 503\sqrt{\frac{\rho}{f}}$$

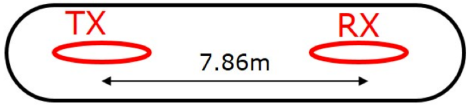
Low frequency   High frequency



Resolve system (2008)



Horizontal Co-planar



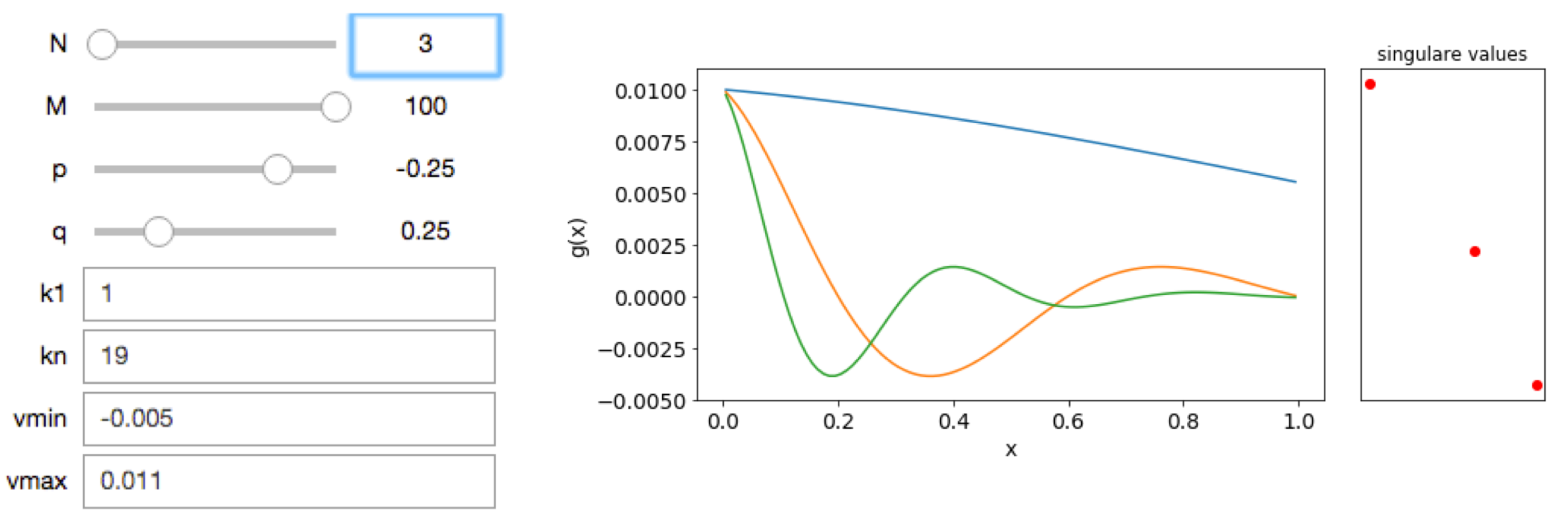
HCP frequencies:
382, 1822, 7970, 35920 and 130100 Hz

Kernels with the app (demo)

- Kernels for FEM are decaying sinusoids

$$g_j(x) = e^{jpx} cos(2\pi jqx)$$

- How many kernels for FEM?
  - MAXMIN: 8 or 10 frequencies, in-phase & quadrature
  - Resolve: In-phase & quad, 5 frequencies



Discretize the app's data and kernels

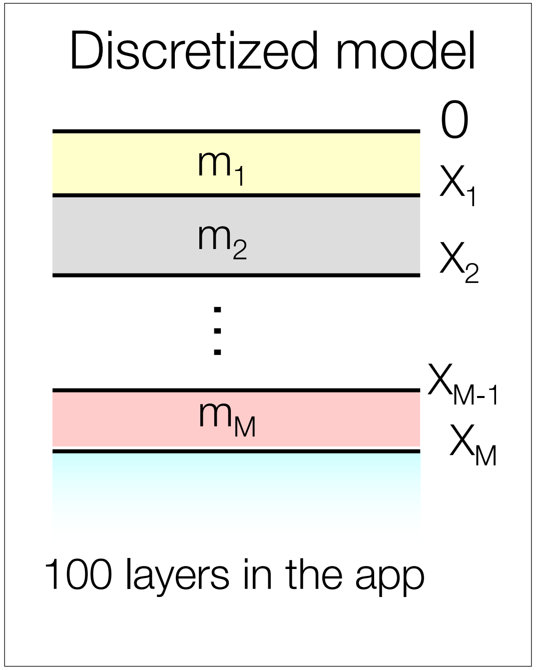
- Datum is defined as: $d_i = \int_0^1 g_i(x)m(x)dx$

- For discretized model:

$$\mathbf{m} = (m_1, m_2, ..., m_M)$$

$$d_i = \int_0^{X_1} g_i(x)m_1dx + \int_{X_1}^{X_2} g_i(x)m_2dx + ...$$

$$= \sum_{j=1}^{M} \Big(\int g_i(x)dx \Big) m_j$$

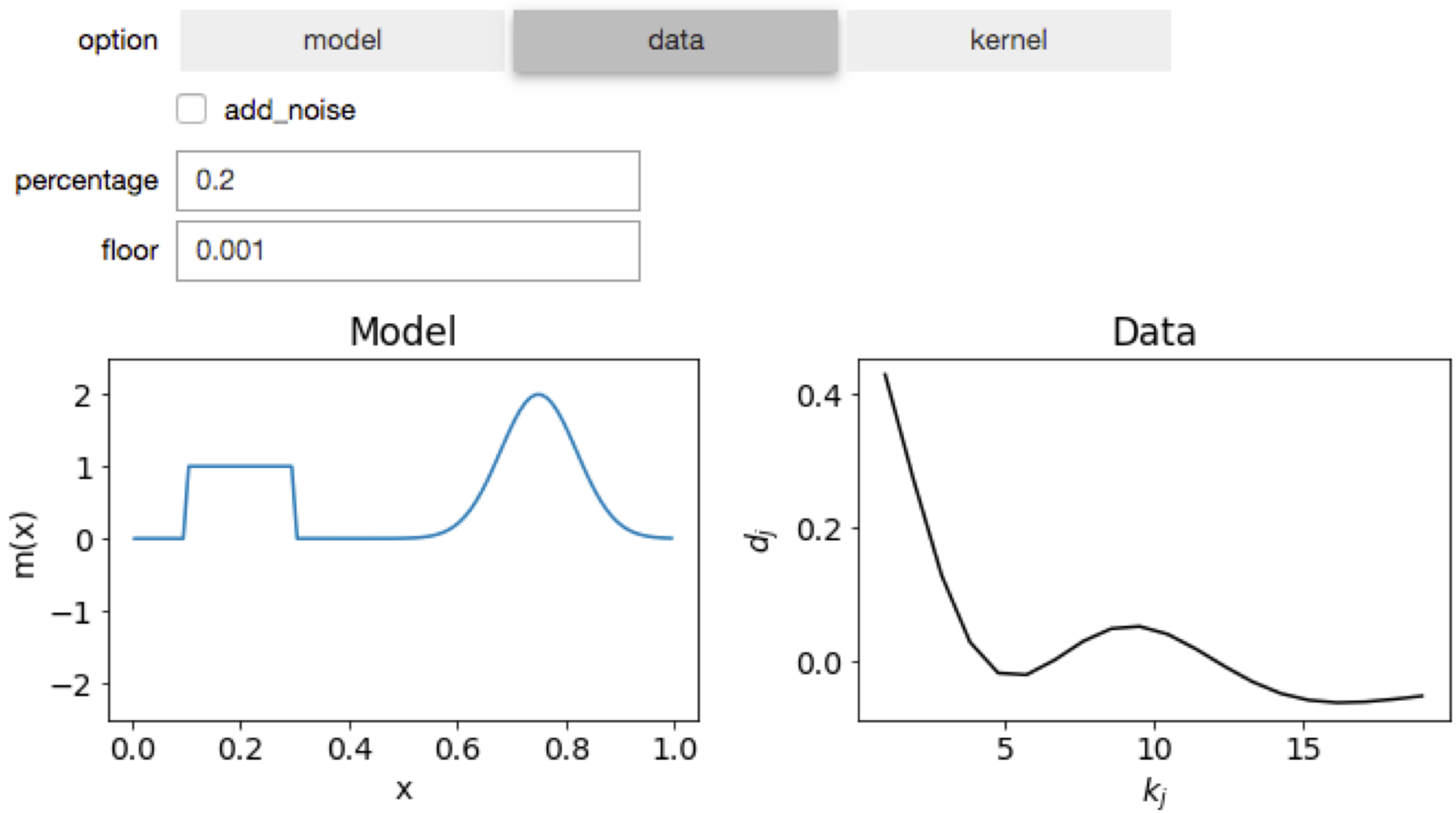


- In matrix form: $\mathbf{d} = \mathbf{G}\mathbf{m}$

$\mathbf{G}$: $(N \times M)$ matrix
$\mathbf{d}$: $(N \times 1)$ vector
$\mathbf{m}$: $(M \times 1)$ vector

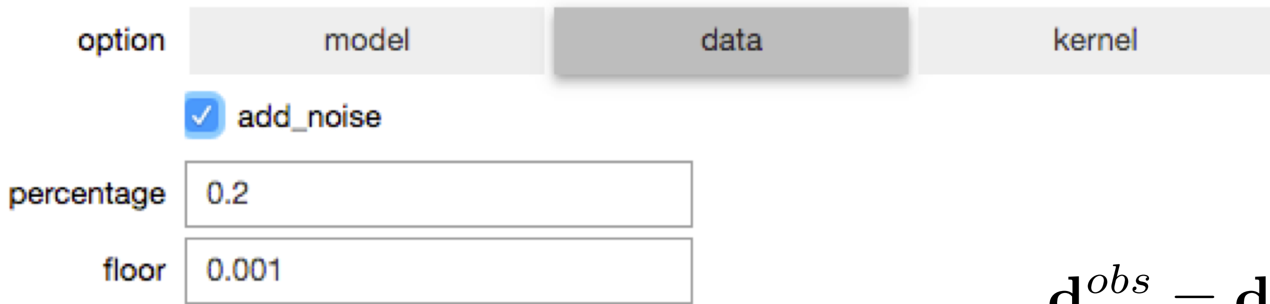
Real observation includes noise

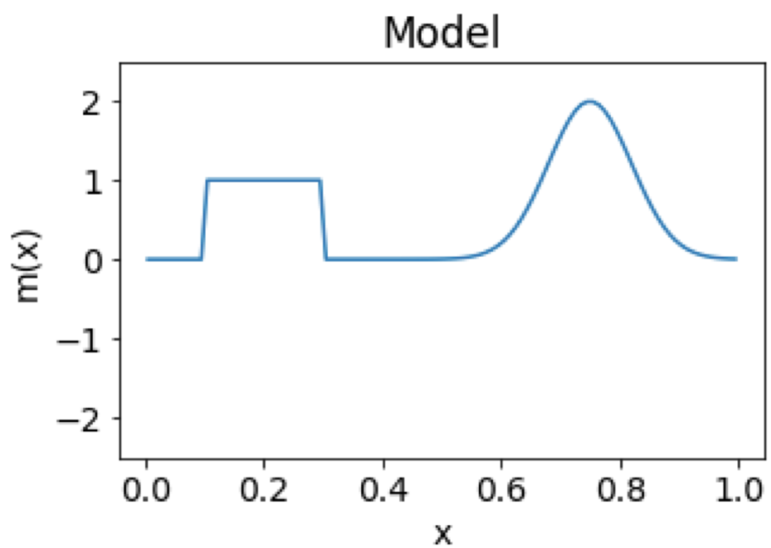
- Data in the app: $\mathbf{d} = \mathbf{G}\mathbf{m}$



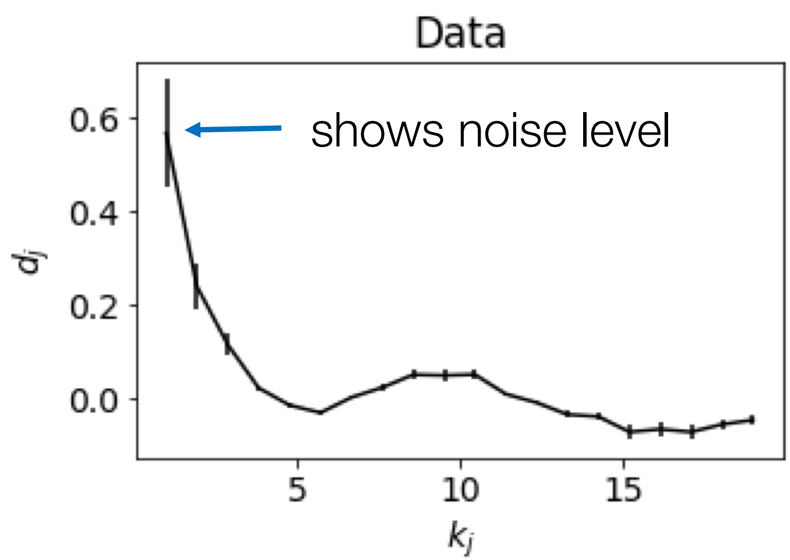
Real observation includes noise

- Data in the app: $\mathbf{d} = \mathbf{G}\mathbf{m}$

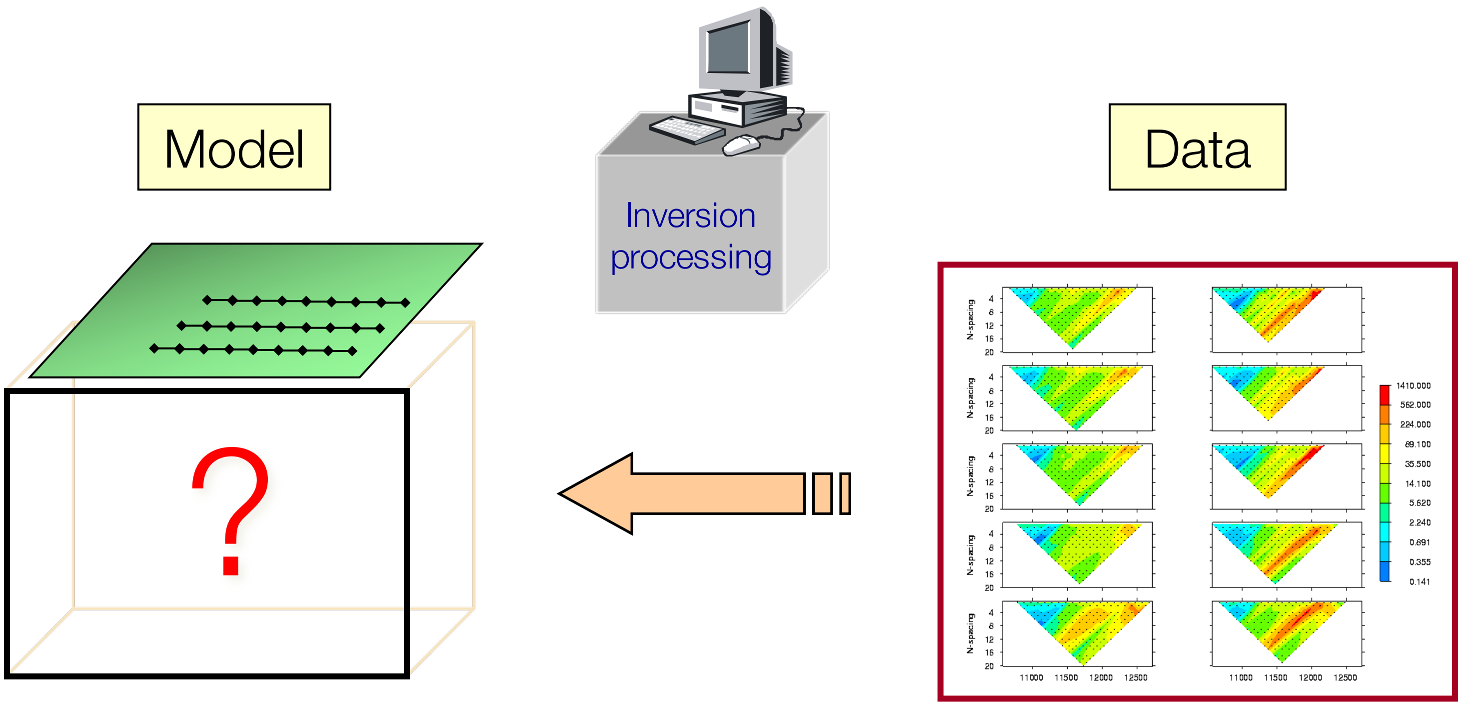




$$\mathbf{d}^{obs} = \mathbf{d} + \mathbf{noise}$$

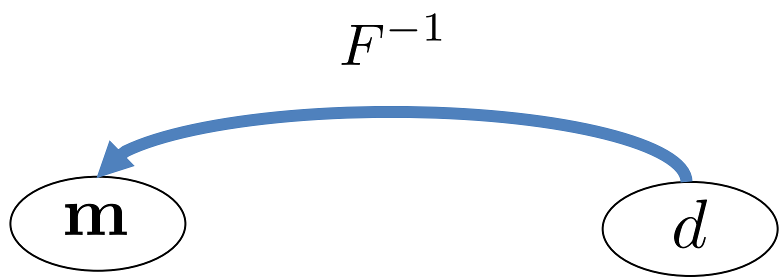


Inversion


Model
Inversion processing
Data
?
N-spacing
4
8
12
16
20
11000
11500
12000
12500
1410.000
562.000
224.000
89.100
35.500
14.100
5.620
2.240
0.891
0.355
0.141

Inverse problem

- Observed data: $d_i^{obs},\ j = 1, 2, ..., N$
- Uncertainty: ϵ_j
- Ability to simulate data: $F[m] = d$
- Find the model which fits the observation



- For linear problem:

$$\mathbf{Gm} = \mathbf{d}$$

$\mathbf{m} \in \mathcal{R}^M$
$\mathbf{d} \in \mathcal{R}^N$
$\mathbf{G} \in \mathcal{R}^{N \times M}$

$M > N$

more unknowns than data
(underdetermined system)

Inversion using Misfit criterion

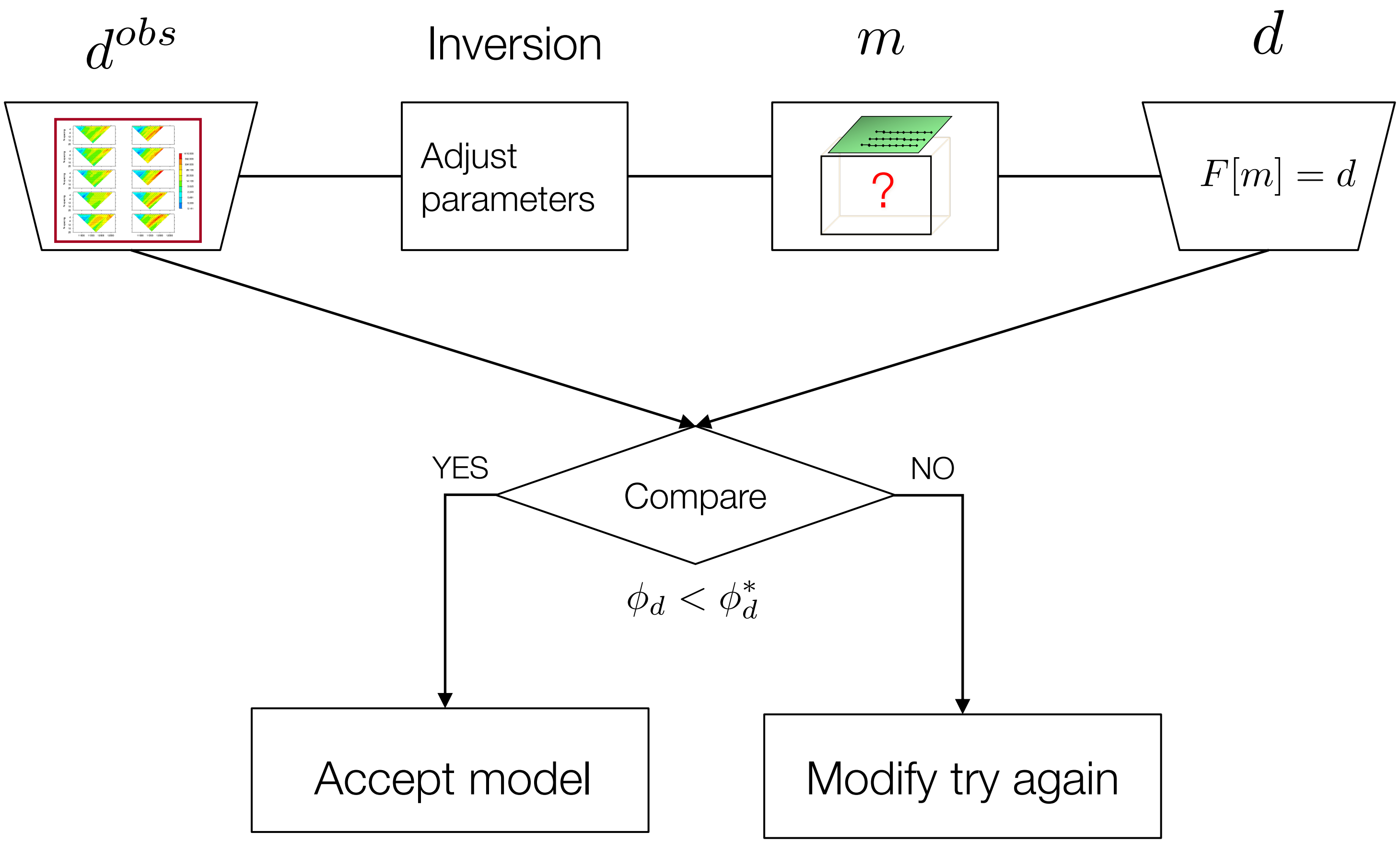
Forward modelling	$\mathbf{Gm} = \mathbf{d}$
Data	$\mathbf{d}^{obs} = \mathbf{d} + \boldsymbol{\epsilon}$
Noise	$\boldsymbol{\epsilon}$

- Gaussian errors with standard deviation, ϵ_j
- Misfit measure: $$\phi_d = \sum_{j=1}^{N} \Big(\frac{d_j - d_j^{obs}}{\epsilon_j}\Big)^2$$
- Expected value of ϕ_d is $E[\phi_d] = N$
- Data are fit when $\phi_d \simeq \phi_d^*$

ϕ_d^*: target misfit

$\phi_d^* = N$

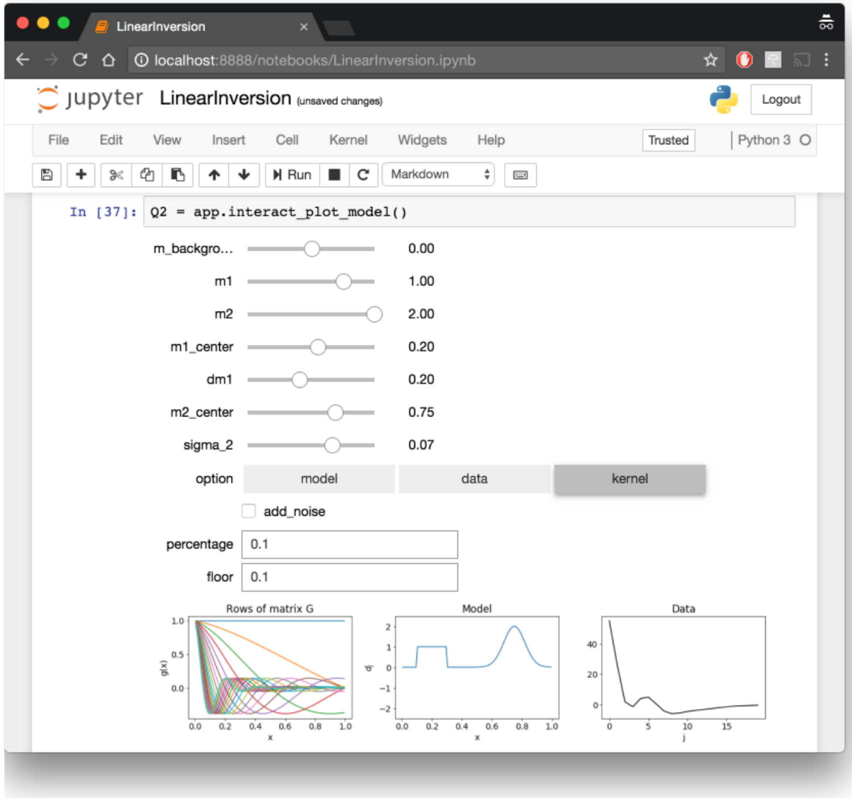
Inversion with misfit only



$$\phi_d = \sum_{j=1}^{N} \left(\frac{d_j - d_j^{obs}}{\epsilon_j} \right)^2$$

Inversion app (demo)

- Use accurate data and show the effects of reducing the data misfit (set α_s=2e-12)
- Add a bit of noise and repeat the process.

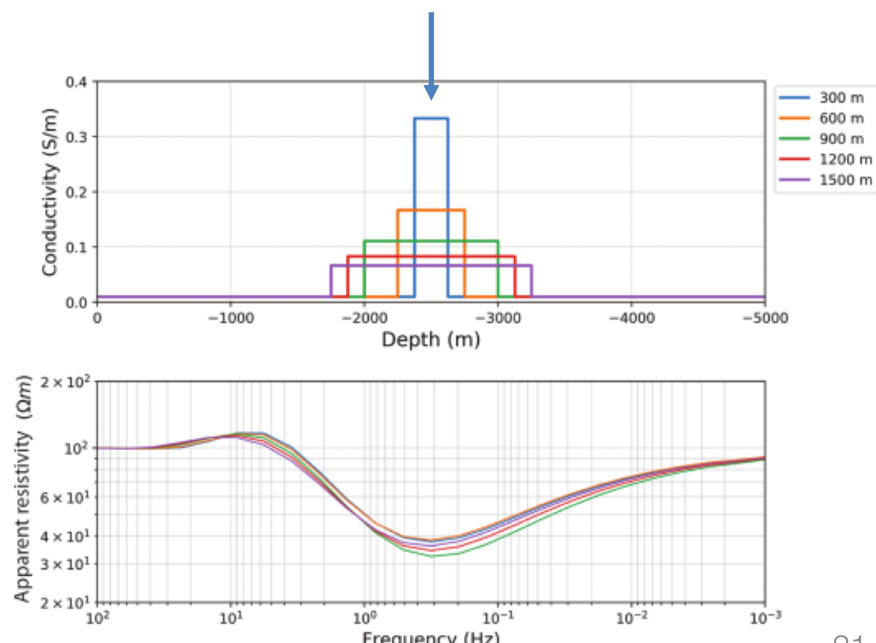


Acceptable models and non-uniqueness

- There are infinitely many models that could generate the data

- Why?
  - # of model parameters (M) > # of data (N)
  - Physics based non-uniqueness
    - Conductance (magnetotelluric)
    - Resistivity-thickness product (DC)
    - Equivalent layer (magnetics)
    - …

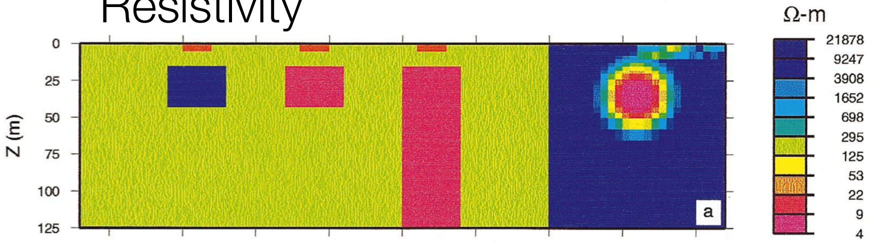
Same conductance (σt)



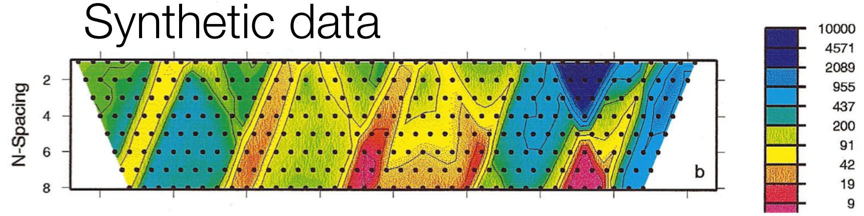
Example non-uniqueness: DC resistivity

- Oldenburg and Li (1999)

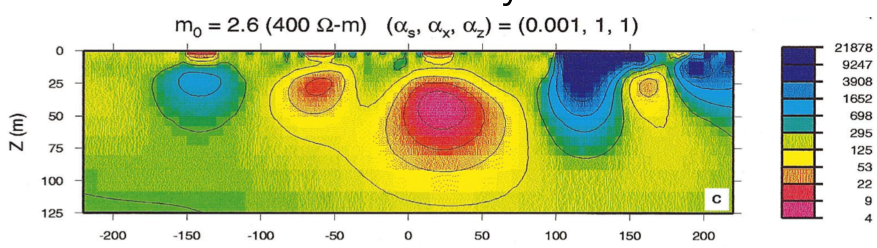
Resistivity



Synthetic data



Recovered resistivity

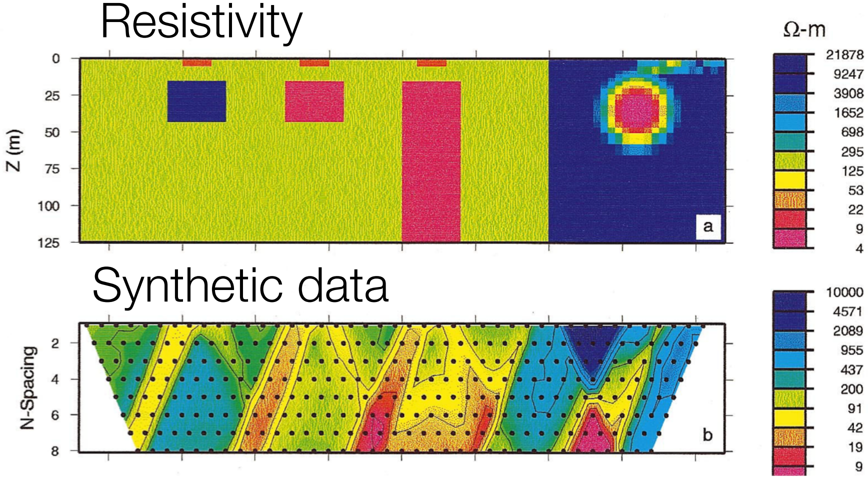


Example non-uniqueness: DC resistivity

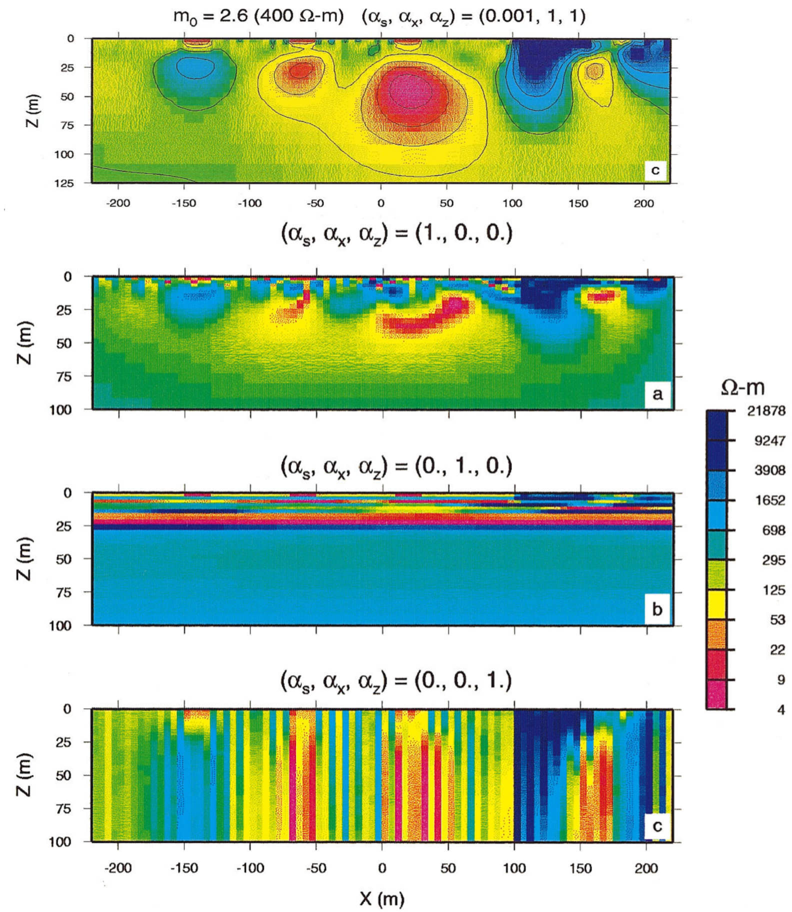
- Oldenburg and Li (1999)

Resistivity

Synthetic data



Recovered resistivities



*All models fit the data well!

The basic problem of non-uniqueness

- Each datum is a *volumetric* response
- Data are

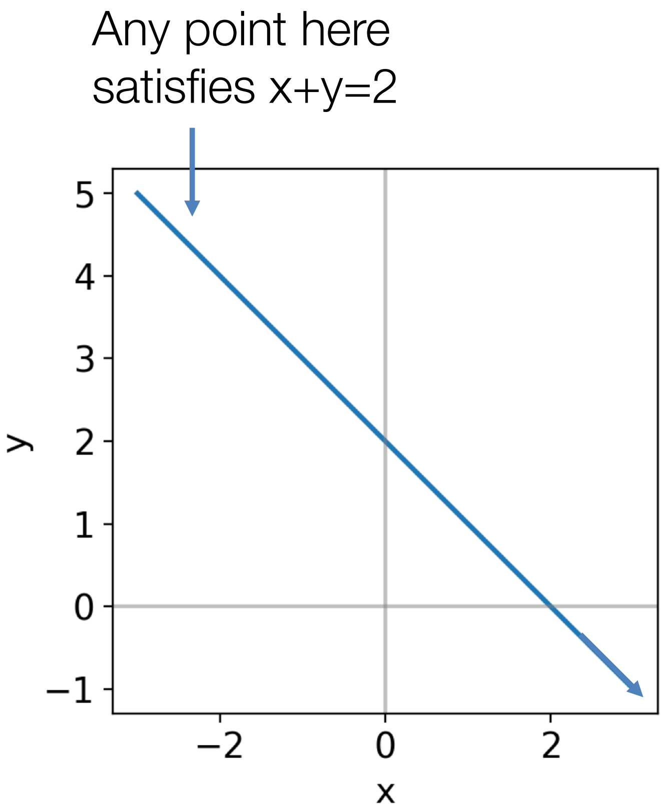
$$d_i = \sum_{j=1}^{M} G_{ij} m_j$$

$$\mathbf{d} = \mathbf{G}\mathbf{m} \qquad \mathbf{G} : (N \times M)$$

- In the app
  - M=100
  - N=20
  - So, M>N (underdetermined problem) → infinitely many solutions

Questions to consider

- Consider the simple problem that involves two unknowns: *x* and *y*
  - We have one datum: $x + y = 2$
- What is the value of *x* and *y*?
  - (1,1)
  - (2,0)
  - ….



How to pick one of infinitely many solutions?

Use prior knowledge

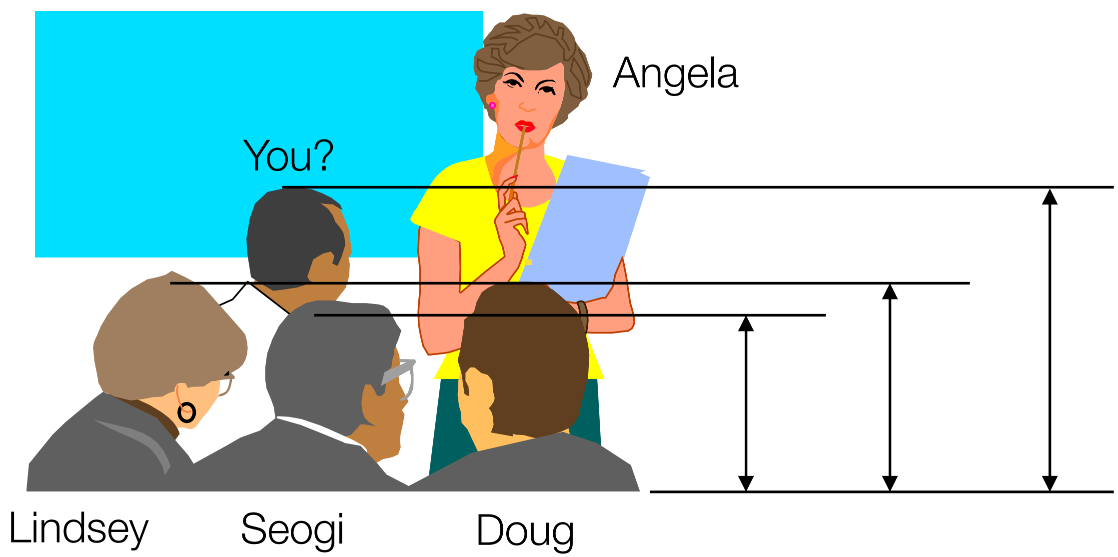
- Geophysical:
  - Values are positive, and/or within bounds
  - Physical Properties: Estimates for host rock properties
  - Point-location values from drill hole information

- Logical:
  - Find a “simple” result (as featureless as possible)

- Geologic:
  - Character of the model (smooth, sparse, blocky)
  - Some idea of scale length (or size) of the bodies
  - Structural constraints

Using prior information to choose optimal models

- Recall we are building an automated decision-making scheme
- Encode prior knowledge in a form that can be optimized
- *i.e.* build a mathematical ruler to test sizes of possible models, then choose the “smallest”
- The people-in-the-room analogy



Feasible model norms

- What "measures on the model" can be implemented?
  - "Size of the model"
  - "Flatness of the model"
- Consider the 4-parameter problem:
  - 4 unknowns: (m_1, m_2, m_3, m_4)
  - 2 data: (6, 2)

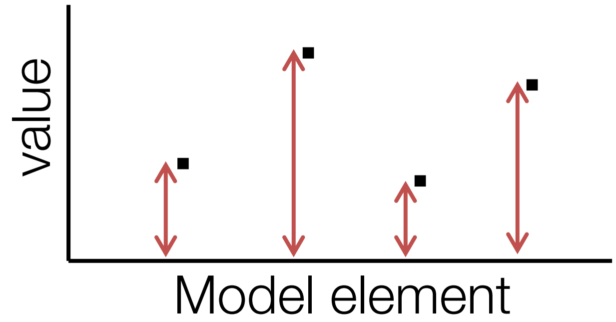
$$\begin{aligned} m_1 + 2m_2 - m_3 + m_4 &= 6 \\ -m_1 + m_2 + 2m_3 - m_4 &= 2 \end{aligned}$$

- It is underdetermined problem, so there is no unique solution
- For possibilities work as well as more…

$$\begin{aligned} \mathbf{m}^A &= (\ \ 2.000,\ \ 2.000,\ \ 2.000,\ \ 2.000) \\ \mathbf{m}^B &= (\ \ 0.444,\ \ 2.622,\ \ 0.133,\ \ 0.444) \\ \mathbf{m}^C &= (-2.408,\ \ 2.630,\ \ 0.109,\ \ 3.256) \\ \mathbf{m}^D &= (\ \ 2.002,\ \ 2.846, -0.537, -2.239) \end{aligned}$$

Choosing from many solutions

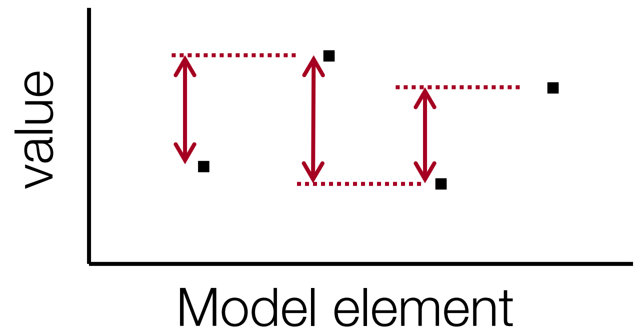
- Define a ruler to measure the model, and call it ϕ_m
- Values of the model can be plotted
- What norms or rulers are sensible?
- Norm #1:
  - Smallness: sum of squares



$$\phi_m = \|\mathbf{m}\|^2 = \sum_{j=1}^{4} m_j^2$$

Choosing from many solutions

- Define a ruler to measure the model, and call it ϕ_m
- Values of the model can be plotted
- What norms or rulers are sensible?



- Norm #1:
  - Smallness: sum of squares

$$\phi_m = \|\mathbf{m}\|^2 = \sum_{j=1}^{4} m_j^2$$

- Norm #2:
  - Smoothness: differences between adjacent model values

$$\phi_m = \|\frac{d\mathbf{m}}{d\mathbf{x}}\|^2 = \sum_{j=1}^{3} (m_{j+1} - m_j^2)$$

Numerical examples

- Use smallest model norm

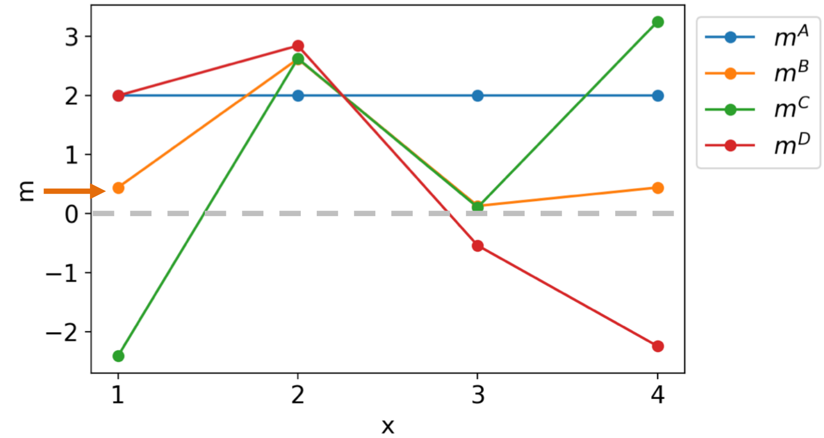
$$\phi_m = \|\mathbf{m}\|^2 = \sum_{j=1}^{4} m_j^2$$

$$\mathbf{m}^A = (\;\;2.000,\;\;2.000,\;\;2.000,\;\;2.000) \quad \phi_m^A = 16.00$$

$$\mathbf{m}^B = (\;\;0.444,\;\;2.622,\;\;0.133,\;\;0.444) \quad \phi_m^B = 7.29$$

$$\mathbf{m}^C = (-2.408,\;\;2.630,\;\;0.109,\;\;3.256) \quad \phi_m^C = 23.33$$

$$\mathbf{m}^D = (\;\;2.002,\;\;2.846,-0.537,-2.239) \quad \phi_m^D = 17.41$$

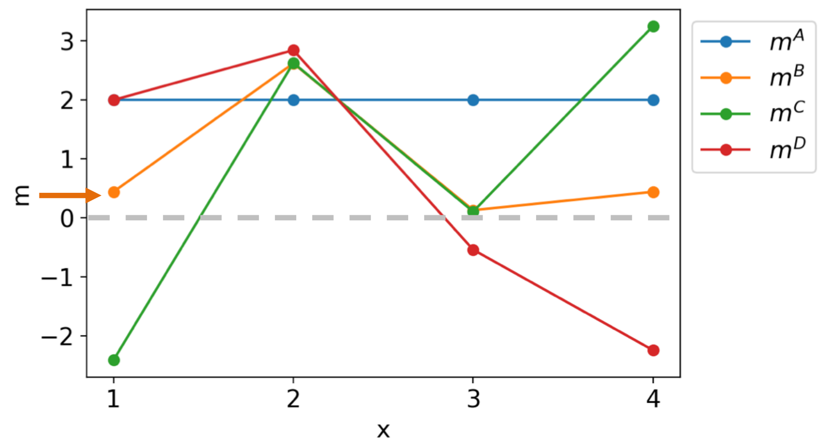


Numerical examples

- Use smallest model norm

$$\phi_m = \|\mathbf{m}\|^2 = \sum_{j=1}^{4} m_j^2$$

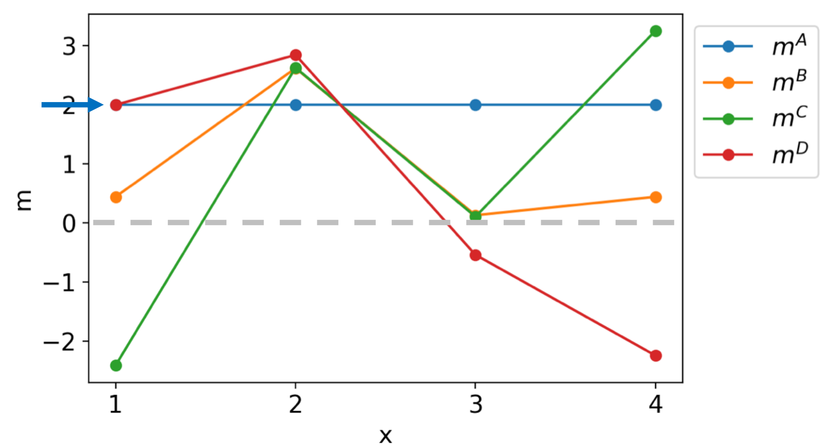
$$\begin{aligned}
\mathbf{m}^A &= (\ \ 2.000,\ \ 2.000,\ \ 2.000,\ \ 2.000) \quad \phi_m^A = 16.00 \\
\mathbf{m}^B &= (\ \ 0.444,\ \ 2.622,\ \ 0.133,\ \ 0.444) \quad \phi_m^B = 7.29 \\
\mathbf{m}^C &= (-2.408,\ \ 2.630,\ \ 0.109,\ \ 3.256) \quad \phi_m^C = 23.33 \\
\mathbf{m}^D &= (\ \ 2.002,\ \ 2.846, -0.537, -2.239) \quad \phi_m^D = 17.41
\end{aligned}$$



- Use smoothest model norm

$$\phi_m = \|\frac{d\mathbf{m}}{d\mathbf{x}}\|^2 = \sum_{j=1}^{3} (m_{j+1} - m_j^2)$$

$$\begin{aligned}
\mathbf{m}^A &= (\ \ 2.000,\ \ 2.000,\ \ 2.000,\ \ 2.000) \quad \phi_m^A = 0.00 \\
\mathbf{m}^B &= (\ \ 0.444,\ \ 2.622,\ \ 0.133,\ \ 0.444) \quad \phi_m^B = 11.04 \\
\mathbf{m}^C &= (-2.408,\ \ 2.630,\ \ 0.109,\ \ 3.256) \quad \phi_m^C = 41.64 \\
\mathbf{m}^D &= (\ \ 2.002,\ \ 2.846, -0.537, -2.239) \quad \phi_m^D = 15.05
\end{aligned}$$



Model norms

Smallest model: $$\phi_m = \int m^2 dx$$

Smallest with reference:

$$\phi_m = \int (m - m_{ref})^2 dx$$

Smoothest model: $$\phi_m = \int \left(\frac{dm}{dx}\right)^2 dx$$

Combination: $$\phi_m = \alpha_s \int (m - m_{ref})^2 dx + \alpha_x \int \left(\frac{dm}{dx}\right)^2 dx$$

Discretize: $$\phi_m = \alpha_s \|\mathbf{W}_s(\mathbf{m} - \mathbf{m}_{ref})\|_2^2 + \alpha_x \|\mathbf{W}_x(\mathbf{m})\|_2^2$$

Combining misfit and model norm

- A statement of the inverse problem is:

  Find the model *m* that

  - produces an acceptable misfit ($\phi_d < \phi_d^*$)
  - minimizes the model norm, ϕ_m

- Re-cast as an optimization:

$$\text{minimize} \quad \phi_d + \beta\phi_m$$

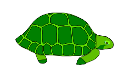
$$\text{where } 0 < \beta < \infty$$

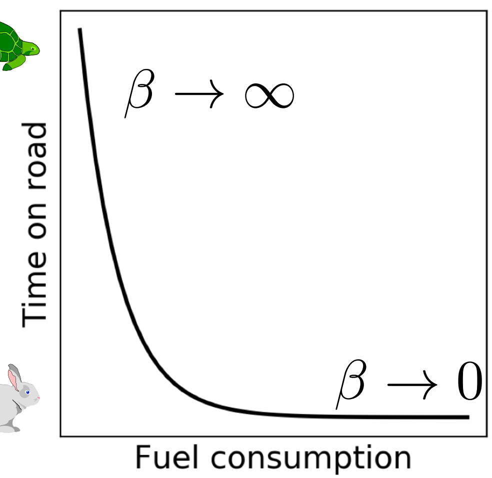
- β : trade-off (Tikhonov) parameter

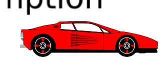
The role of β

Analogy: an optimization problem with two requirements

- Travelling from A to B
  - minimize **time** taken
  - minimize **fuel** consumption
- $\phi = \text{time} + \beta \cdot \text{fuel}$
  - both time and fuel consumption are functions of speed
- $\beta = 0$: minimize time (regardless of fuel)
- $\text{large } \beta$: minimize fuel (but still get there)





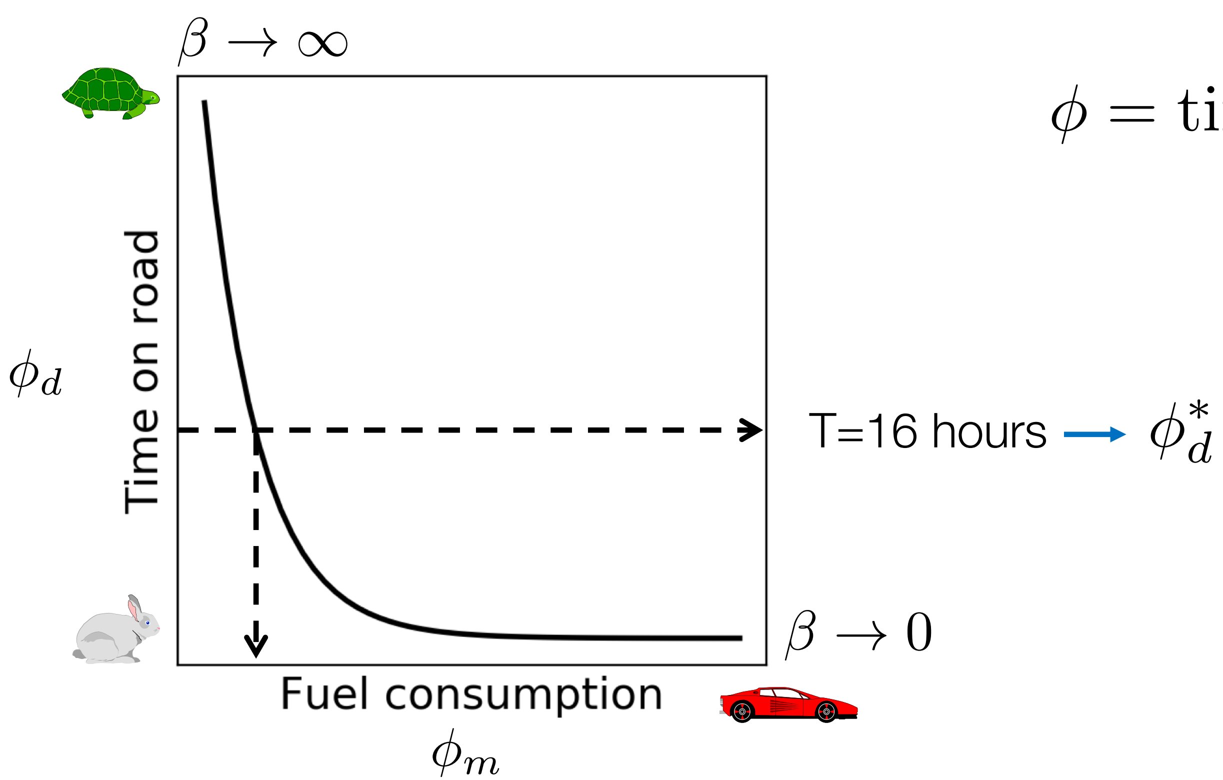


The role of β

- A typical problem might be:
  - Minimize fuel consumption
  - Subject to getting there in 16 hours

$$\phi = \text{time} + \beta \cdot \text{fuel}$$

time $\rightarrow \phi_d$, fuel $\rightarrow \phi_m$



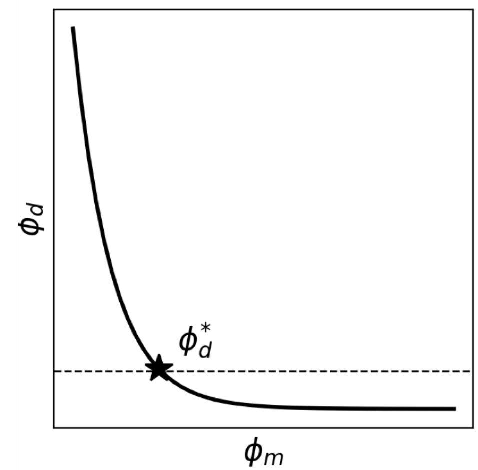
The role of β : managing misfit

- Our inverse problem
  - Find the model (m)

$$\text{minimize} \quad \phi_d + \beta\phi_m$$

  - Which beta to use?
  - If standard deviations of data are known,

Tikhonov curve



$$E[\phi_d] = N$$

$E[\phi_d]$: expected value of ϕ_d

  - Desired misfit is $\phi_d^* \simeq N$
  - Choose β so that $\phi_d(m) = \phi_d^*$

Inversion App (demo)

maxIter 20

m0 0.00

mref 0.00

percentage 0.2

floor 0.001

chifact 1

beta0_ratio 100

coolingFactor 2.00

coolingRate 1

alpha_s 1

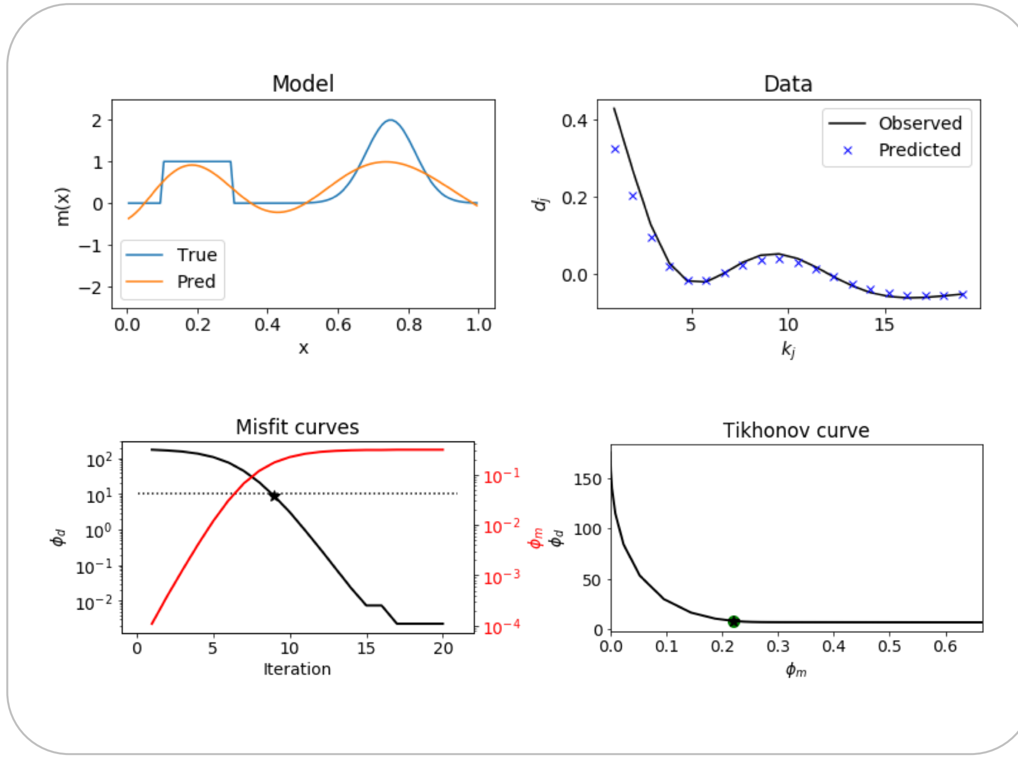
alpha_x 0

use_target

run

option misfit tikhonov

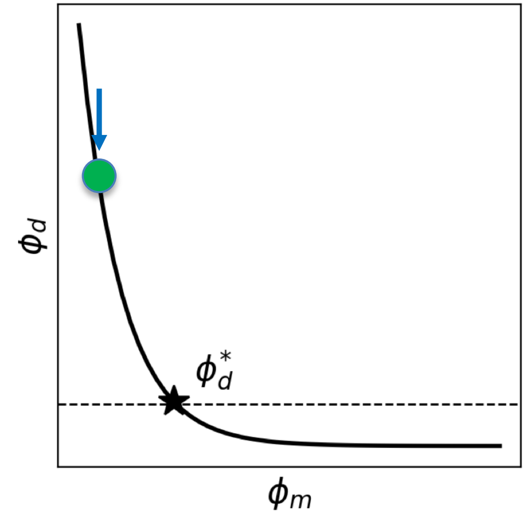
i_iteration 11

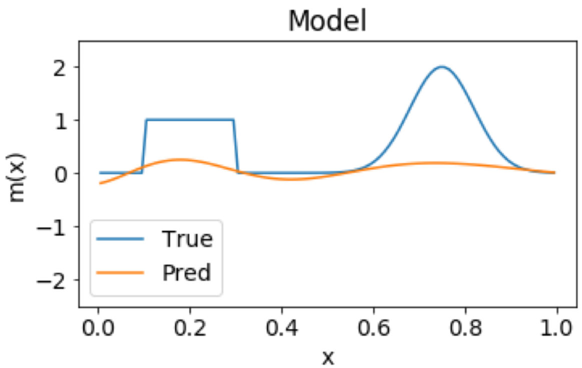


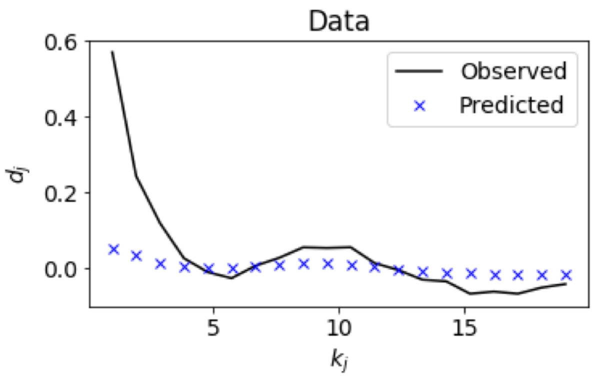
β is the trade-off parameter

- Solve: $\text{minimize} \quad \phi_d + \beta\phi_m$
- β too large → **underfitting**
  - Structural information lost

Tikhonov curve

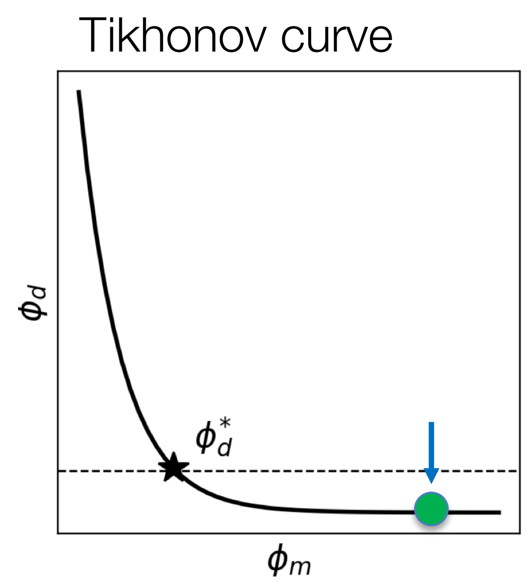


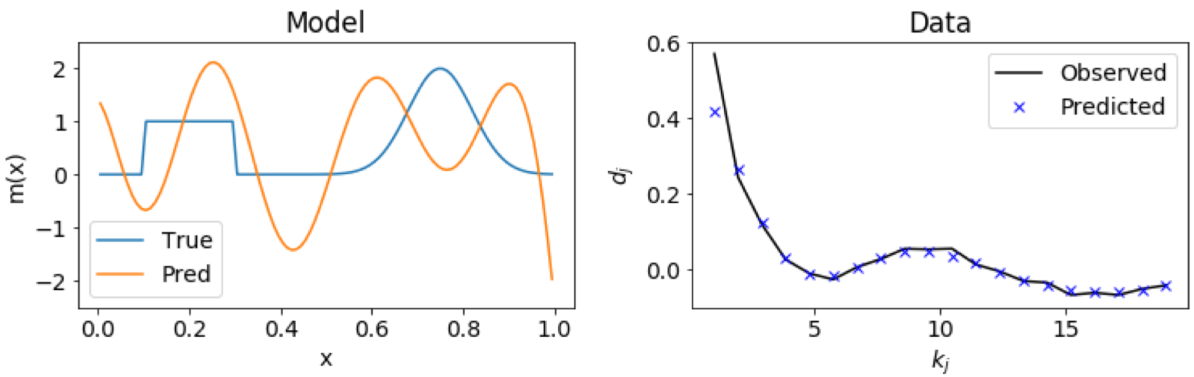




β is the trade-off parameter

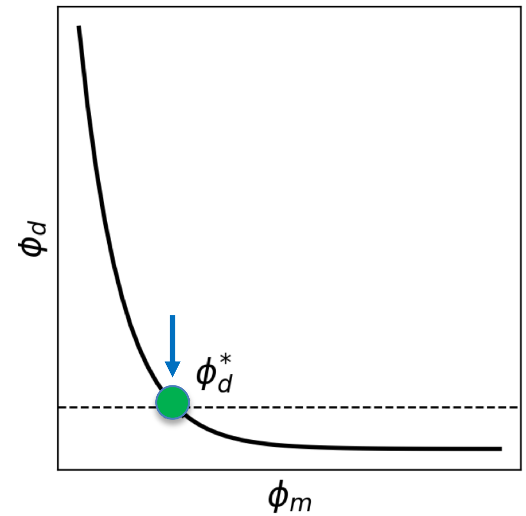
- Solve: $\text{minimize} \quad \phi_d + \beta\phi_m$
- β too large → **underfitting**
  - Structural information lost
- β too small → **overfitting**
  - Structure created to fit noise

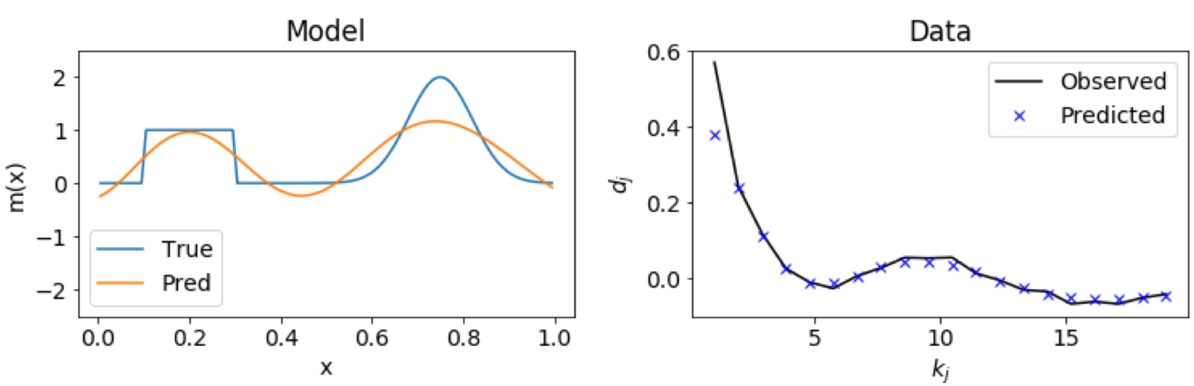




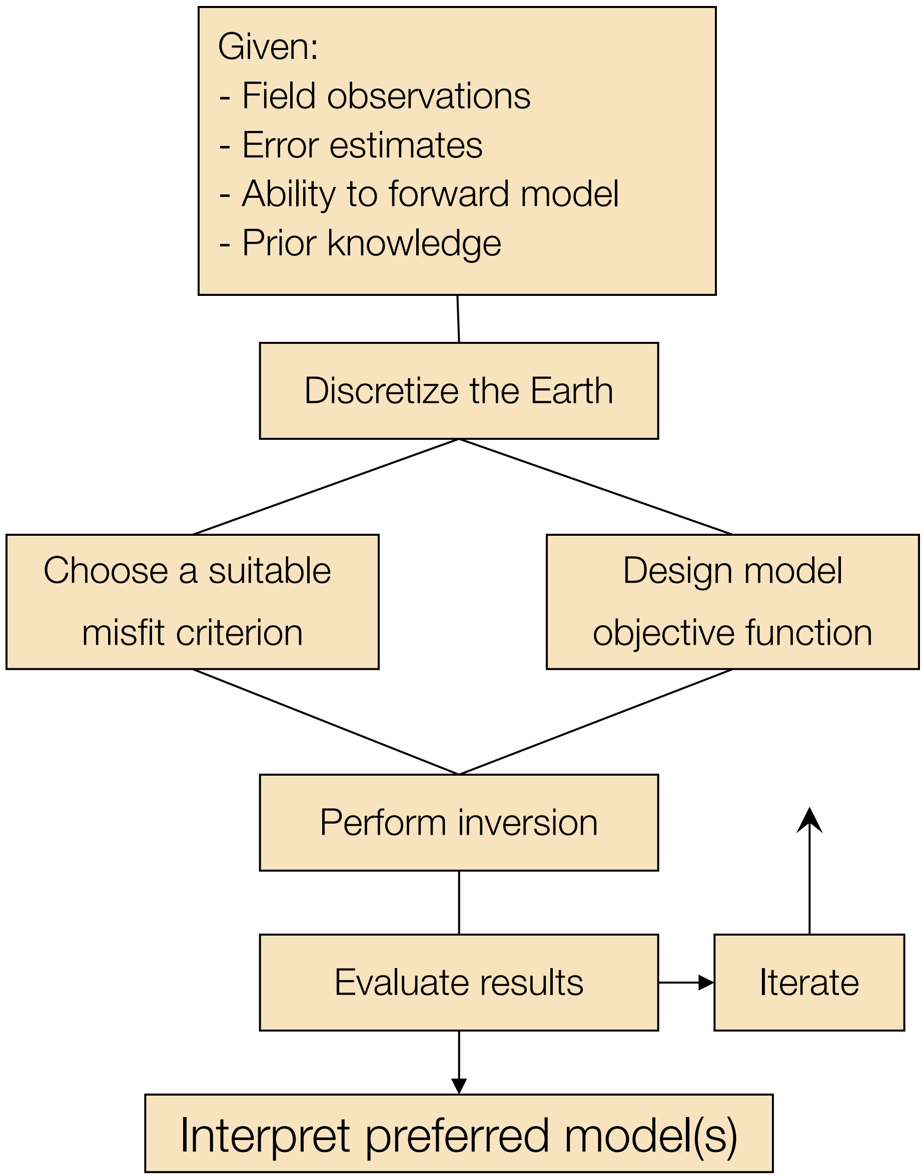
β is the trade-off parameter

- Solve: $\text{minimize} \quad \phi_d + \beta\phi_m$
- β too large → **underfitting**
  - Structural information lost
- β too small → **overfitting**
  - Structure created to fit noise
- β just right ($\phi_d(m) \simeq N$) → optimal fit





Flow chart for inverse problem

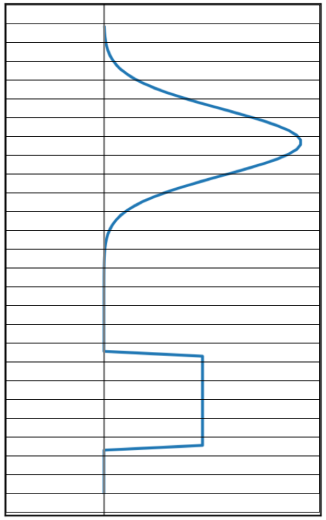


Summary

- Model and discretization

$$\mathbf{m} = (m_1, m_2, ..., m_M)$$

Physical property
(e.g. conductivity)



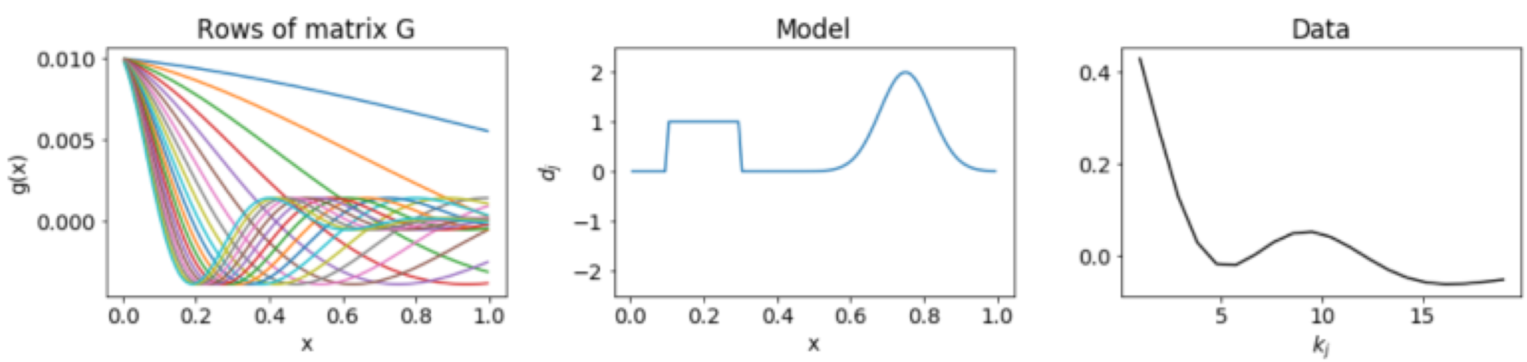
Summary

- Model and discretization
- Data and kernels

$$d_i = \int_v g_i(x)m(x)dx$$

$$d_i = \sum_{j=1}^{M} G_{ij} m_j \qquad i = 1, 2, ..., N$$

$$\mathbf{d} = \mathbf{Gm} \qquad \mathbf{G} : (N \times M)$$

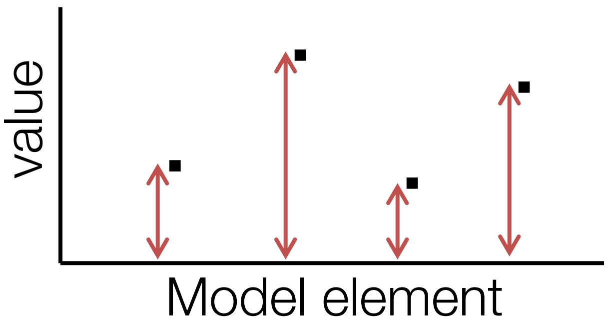


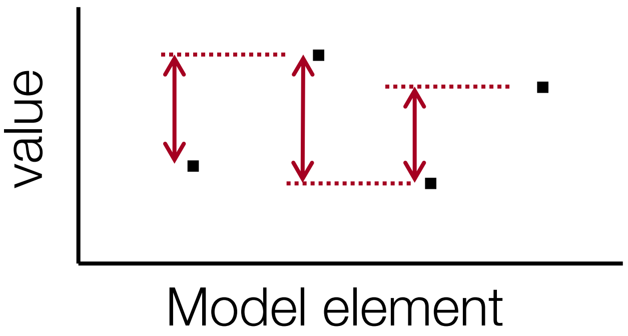
Summary

- Model and discretization
- Data and kernels
- Non-uniqueness and model norms

smallest   smoothest

$$\phi_m = \alpha_s \int (m - m_{ref})^2 dx + \alpha_x \int \left(\frac{dm}{dx}\right)^2 dx$$



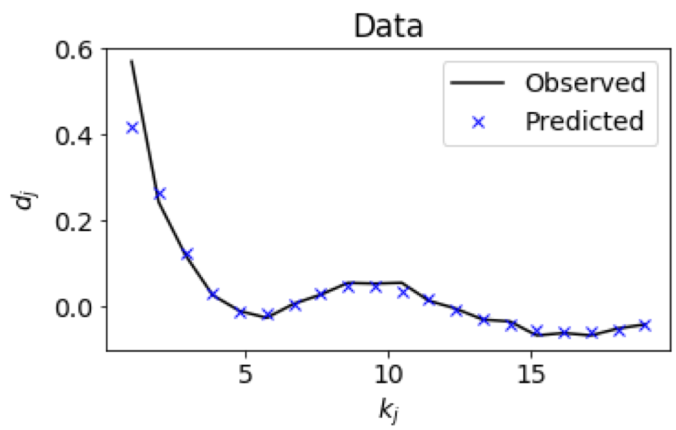


Summary

- Model and discretization
- Data and kernels
- Non-uniqueness and model norms
- Misfit

$$\phi_d = \sum_{j=1}^{N} \left(\frac{d_j - d_j^{obs}}{\epsilon_j} \right)^2$$

$$\phi_d^* = N$$

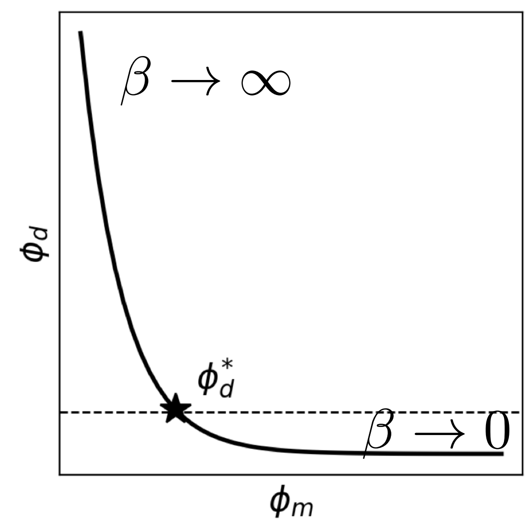


Summary

- Model and discretization
- Data and kernels
- Non-uniqueness and model norms
- Misfit
- Inversion as an optimization
- Choice of emphasis: misfit vs. model norm

$$\text{minimize} \quad \phi_d + \beta\phi_m$$

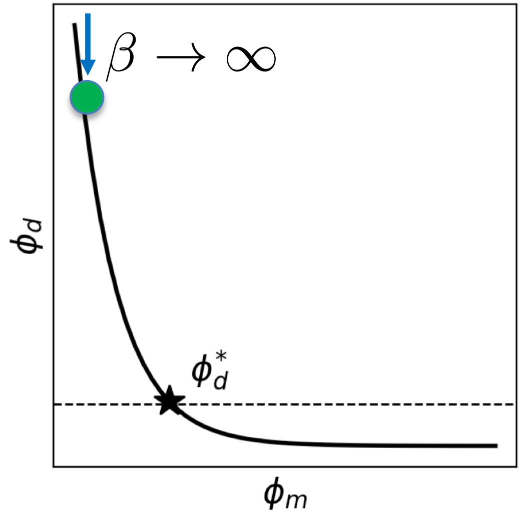
$$\text{where } 0 < \beta < \infty$$



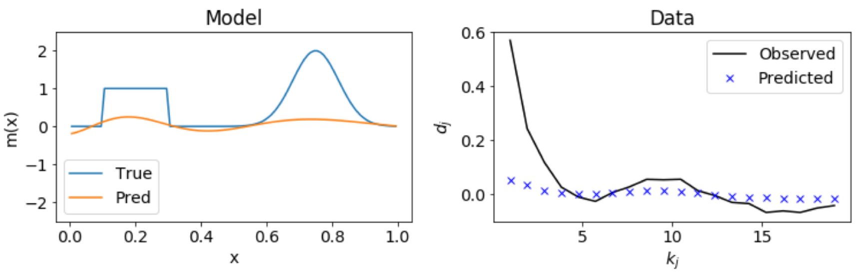
Summary

- Model and discretization
- Data and kernels
- Non-uniqueness and model norms
- Misfit
- Inversion as an optimization
- Choice of emphasis: misfit vs. model norm

Tikhonov curve



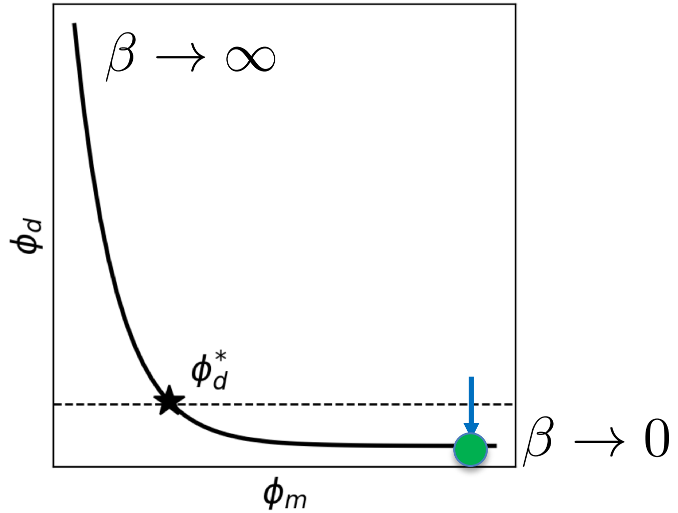
Underfit



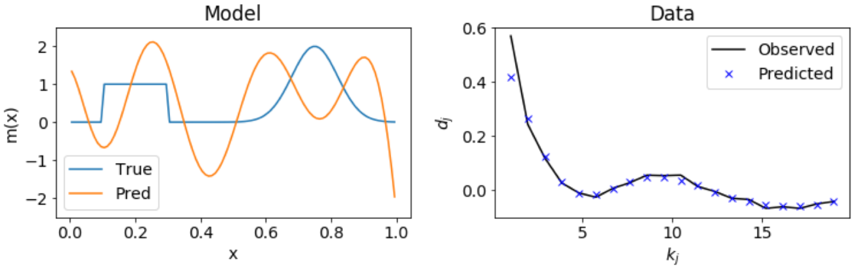
Summary

- Model and discretization
- Data and kernels
- Non-uniqueness and model norms
- Misfit
- Inversion as an optimization
- Choice of emphasis: misfit vs. model norm

Tikhonov curve



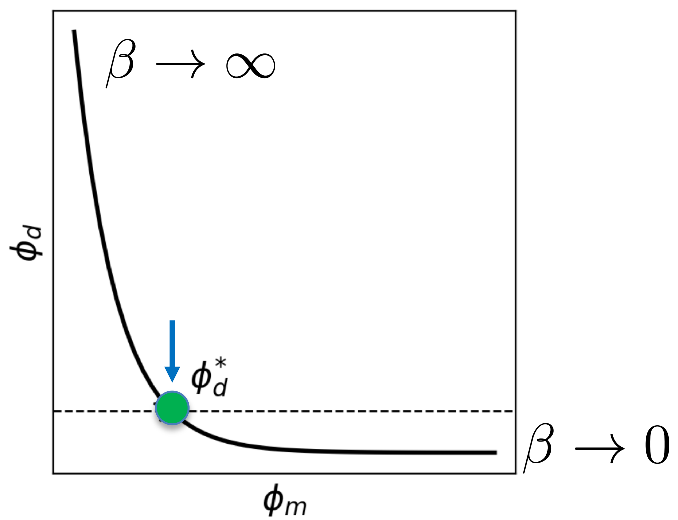
Overfit



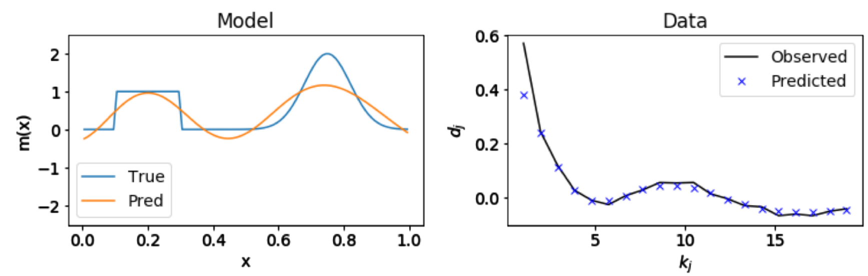
Summary

- Model and discretization
- Data and kernels
- Non-uniqueness and model norms
- Misfit
- Inversion as an optimization
- Choice of emphasis: misfit vs. model norm

Tikhonov curve



Optimal fit



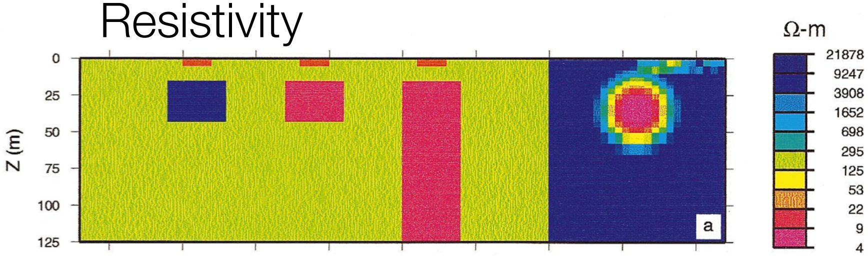
Summary

- Model and discretization
- Data and kernels
- Non-uniqueness and model norms
- Misfit
- Inversion as an optimization
- Choice of emphasis: misfit vs. model norm
- Most geophysical problems are non-linear
  - DC resistivity
  - EM
  - MT
  - …

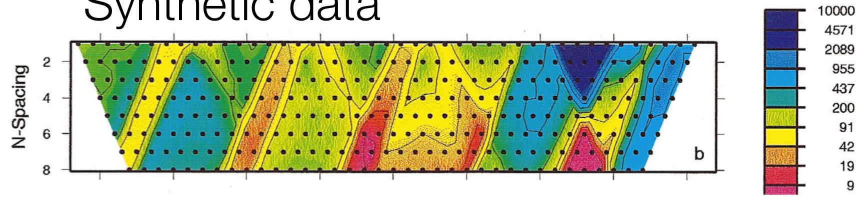
DC resistivity

- Oldenburg and Li (1999)

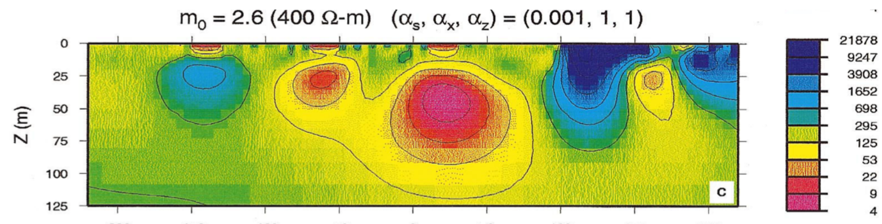
Resistivity



Synthetic data

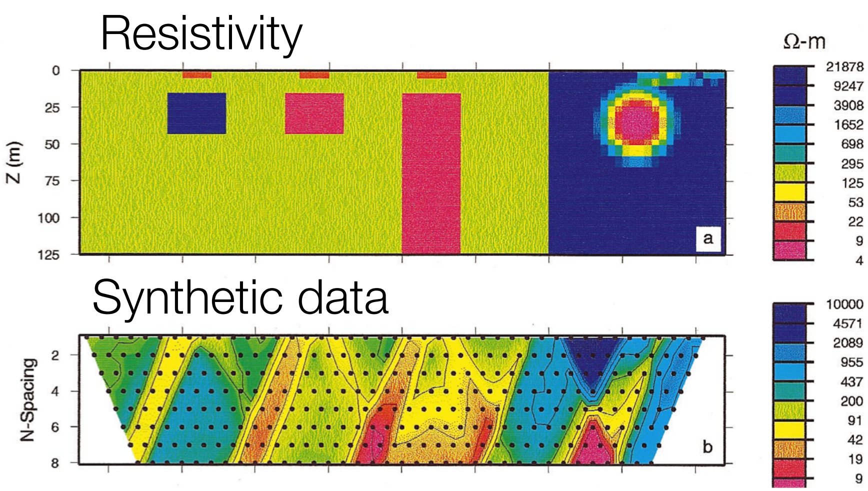


Recovered resistivity

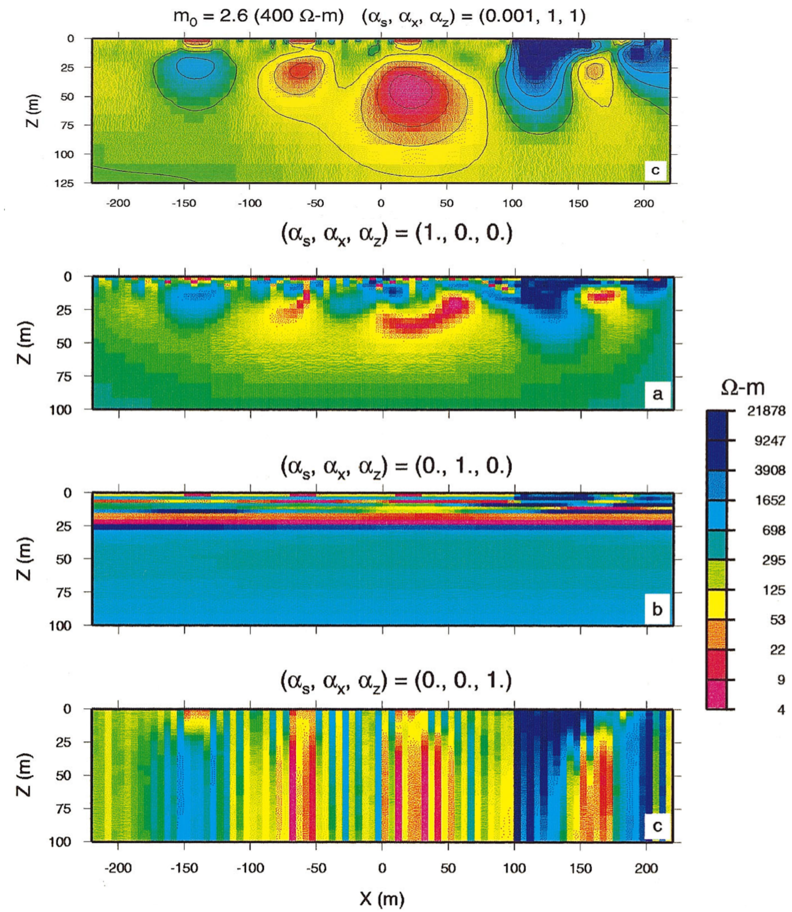


DC resistivity: non-uniqueness

- Oldenburg and Li (1999)



Recovered resistivities

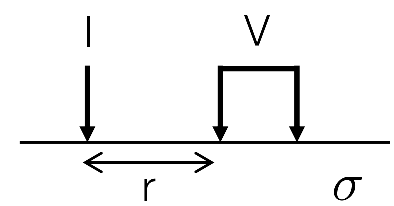


*All models fit the data well!

Nonlinear problem

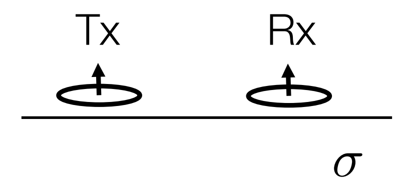
- Examples
  - DC:

$$\nabla \cdot \sigma \nabla V = I_0 \delta(\mathbf{r} - \mathbf{r}_s)$$



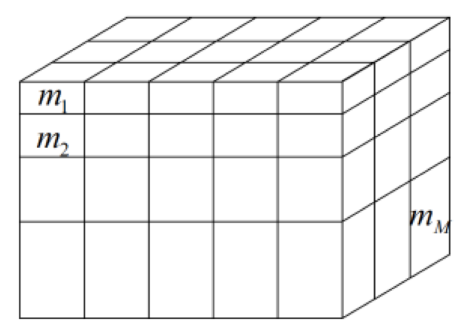
  - or EM:

$$\nabla \times \mu^{-1} \times \mathbf{e} + \sigma \frac{\partial \mathbf{e}}{\partial t} = -\frac{\partial \mathbf{s}}{\partial t}$$



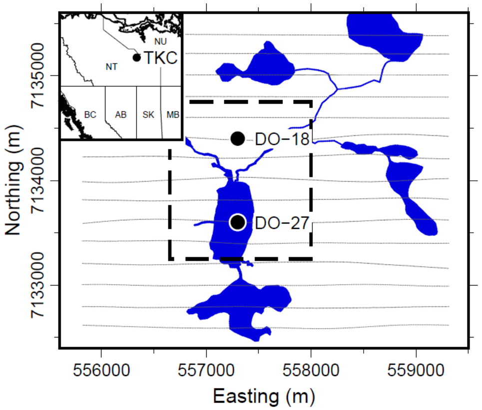
- Solve
  - Discretize Maxwell's equations onto a mesh
  - Solve system to find fields (e.g. $V, \ \mathbf{e}$)
  - Evaluate Datum: $d = f[V]$ or $d = f[\mathbf{e}]$

- Lot of details
  - size of cells
  - size of mesh
  - …

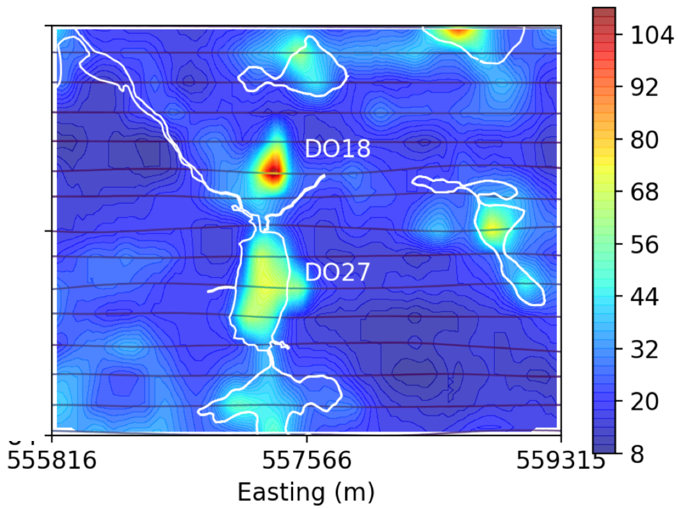


Airborne EM: Tli Kwi Cho (TKC) kimerlites

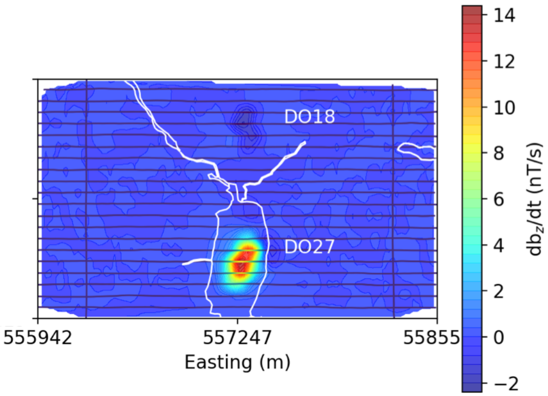
Location map at TKC



DIGHEM (Quadrature 56kHz)



VTEM (90 µs)



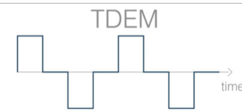
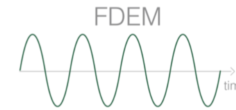
DIGHEM (1992)

Configuration	HCP
Frequency	900Hz-56kHz
Data unit	ppm
Line spacing	200 m
Line km	52 km
# of sounding	6274

VTEM (2003)

Configuration	Colocated-loop
Off time channel	90-6340 (µs)
Data unit	pV/A-m^4
Line spacing	75 m
Line km	39 km
# of sounding	26342

Basic Equations

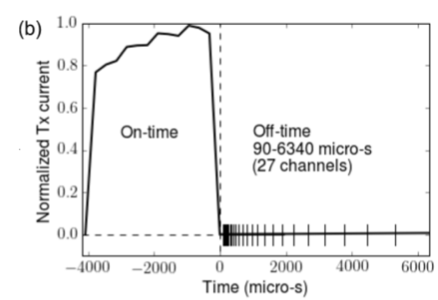
	Time 	Frequency 
Faraday's Law	$\nabla \times \mathbf{e} = -\frac{\partial \mathbf{b}}{\partial t}$	$\nabla \times \mathbf{E} = -i\omega \mathbf{B}$
Ampere's Law	$\nabla \times \mathbf{h} = \mathbf{j} + \frac{\partial \mathbf{d}}{\partial t}$	$\nabla \times \mathbf{H} = \mathbf{J} + i\omega \mathbf{D}$
No Magnetic Monopoles	$\nabla \cdot \mathbf{b} = 0$	$\nabla \cdot \mathbf{B} = 0$
Constitutive Relationships (non-dispersive)	$\mathbf{j} = \sigma \mathbf{e}$ $\mathbf{b} = \mu \mathbf{h}$ $\mathbf{d} = \varepsilon \mathbf{e}$	$\mathbf{J} = \sigma \mathbf{E}$ $\mathbf{B} = \mu \mathbf{H}$ $\mathbf{D} = \varepsilon \mathbf{E}$

* Solve with sources and boundary conditions

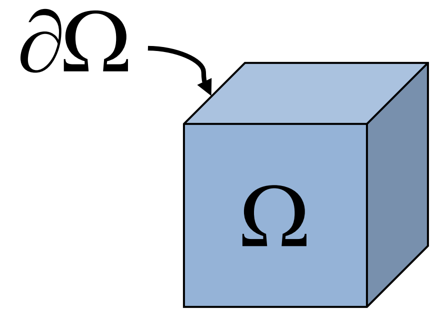
Why difficult: Forward Problem

- Discretize in frequency or time
- Discretize in space: (1D vs 3D)
- Solve system of equations
- Many transmitters

Time (TDEM)	Frequency (FDEM)
$\nabla \times \mathbf{e} = -\frac{\partial \mathbf{b}}{\partial t}$	$\nabla \times \mathbf{E} = -i\omega \mathbf{B}$
$\nabla \times \mathbf{h} = \mathbf{j} + \frac{\partial \mathbf{d}}{\partial t}$	$\nabla \times \mathbf{H} = \mathbf{J} + i\omega \mathbf{D}$
$\nabla \cdot \mathbf{b} = 0$	$\nabla \cdot \mathbf{B} = 0$
$\mathbf{j} = \sigma \mathbf{e}$	$\mathbf{J} = \sigma \mathbf{E}$
$\mathbf{b} = \mu \mathbf{h}$	$\mathbf{B} = \mu \mathbf{H}$
$\mathbf{d} = \varepsilon \mathbf{e}$	$\mathbf{D} = \varepsilon \mathbf{E}$



Time Domain: Mathematical Setup



time: $[0, t_f]$

Maxwell's equations

$$\nabla \times \mathbf{e} + \frac{\partial \mathbf{b}}{\partial t} = 0$$

$$\nabla \times \mu^{-1}\mathbf{b} - \sigma\mathbf{e} = \mathbf{s}(t)$$

Boundary conditions

$$\mathbf{n} \times \mathbf{b} = 0$$

Initial conditions

$$\mathbf{e}(x, y, z, t = 0) = \mathbf{e}_0$$

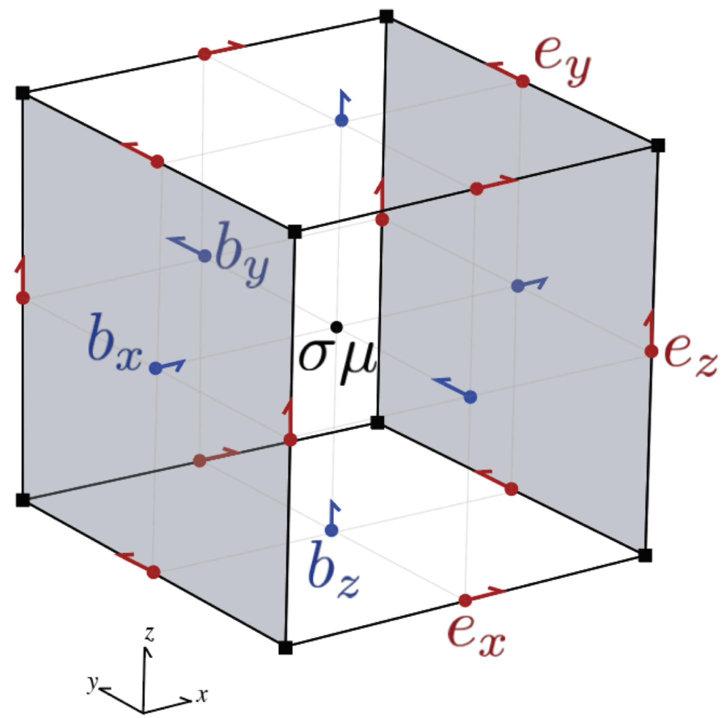
$$\mathbf{b}(x, y, z, t = 0) = \mathbf{b}_0$$

Need to solve in space and time

Semi-discretization in space

Staggered Grid

- Physical properties: cell centers
- Fields: edges
- Fluxes: faces



Continuous second-order equations

$$\nabla \times \mu^{-1} \nabla \times \mathbf{e} + \sigma \frac{\partial \mathbf{e}}{\partial t} = -\frac{\partial \mathbf{s}}{\partial t}$$

Semi-discretized second order equations

$$\mathbf{C}^\top \mathbf{M}^f_{\mu^{-1}} \mathbf{C} \mathbf{e} + \mathbf{M}^e_\sigma \frac{\partial \mathbf{e}}{\partial t} = -\frac{\partial \mathbf{s}}{\partial t}$$

Discretizing in time

First order backwards difference (implicit)

- $\mathbf{e}^{n+1}$ depends upon $\mathbf{e}^{n}$

$$\mathbf{C}^{\top}\mathbf{M}^{f}_{\mu^{-1}}\mathbf{C}\mathbf{e} + \mathbf{M}^{e}_{\sigma}\frac{\partial \mathbf{e}}{\partial t} = -\frac{\partial \mathbf{s}}{\partial t}$$

- Time-step: $\Delta t = t_{n+1} - t_n$

$$\left(\mathbf{C}^{\top}\mathbf{M}^{f}_{\mu^{-1}}\mathbf{C} + \frac{1}{\Delta t}\mathbf{M}^{e}_{\sigma}\right)\mathbf{e}^{n+1} = -\frac{\mathbf{s}^{n+1} - \mathbf{s}^{n}}{\Delta t} + \frac{1}{\Delta t}\mathbf{M}^{e}_{\sigma}\mathbf{e}^{n}$$

$$\mathbf{A}_{n+1}\mathbf{u}_{n+1} = -\mathbf{B}_{n}\mathbf{u}_{n} + \mathbf{q}_{n+1}$$

Solve system at each time step

Factor $\mathbf{A}_{n+1} = \mathbf{L}\mathbf{L}^{\top}$

Solving a TDEM Problem

Solve with forward elimination

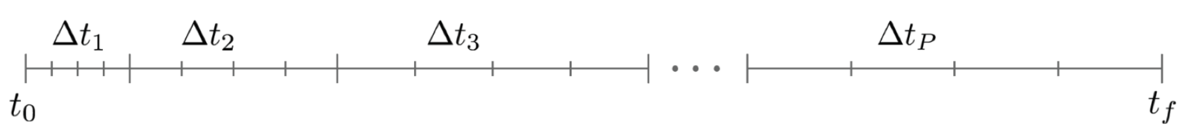
- Initial conditions provide $\mathbf{u}_0$
- To propagate forward, solve

$$\mathbf{A}_{n+1}\mathbf{u}_{n+1} = -\mathbf{B}_n\mathbf{u}_n + \mathbf{q}_{n+1}$$

$$\begin{pmatrix} \mathbf{A}_0 & & & & & \\ \mathbf{B}_1 & \mathbf{A}_1 & & & & \\ & \mathbf{B}_2 & \mathbf{A}_2 & & & \\ & & \ddots & \ddots & & \\ & & & \mathbf{B}_{n-1} & \mathbf{A}_{n-1} & \\ & & & & \mathbf{B}_n & \mathbf{A}_n \end{pmatrix} \begin{pmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_{n-1} \\ \mathbf{u}_n \end{pmatrix} = \begin{pmatrix} \mathbf{q}_0 \\ \mathbf{q}_1 \\ \mathbf{q}_2 \\ \vdots \\ \mathbf{q}_{n-1} \\ \mathbf{q}_n \end{pmatrix}$$

Some details of solving system

- Refactor only if $\mathbf{A}_{n+1}(\sigma, \triangle t)$ changes
- Divide modelling time into P partitions



- Total computation time:

$$T = P(N_{\Delta t} N_{TX} t_{\text{solve}} + t_{\text{factor}})$$

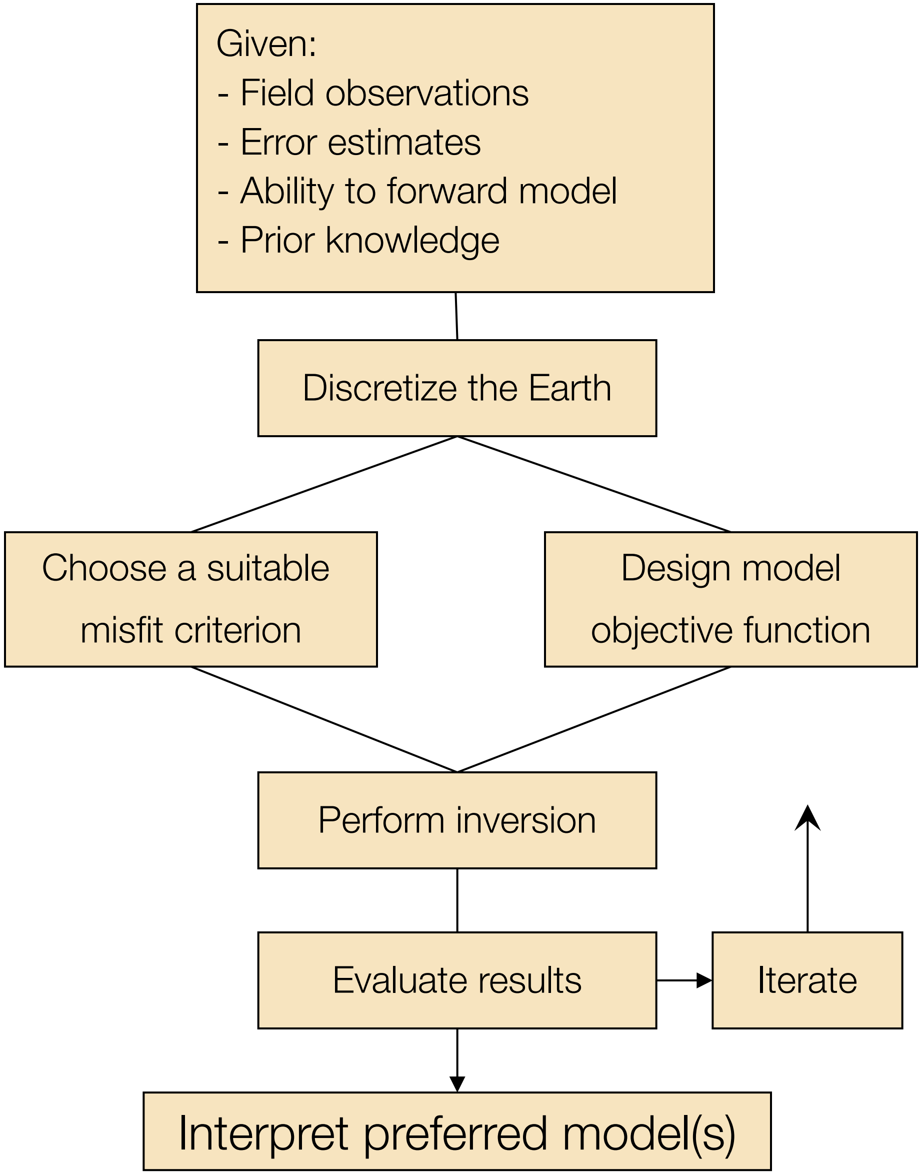
Time to solve factored system

Time to factor system

That was challenging…

What about the inverse problem?

Flow chart for inverse problem



Inverse problem

- Minimize

$$\phi(\mathbf{m}) = \phi_d(\mathbf{m}) + \beta\phi_m(\mathbf{m})$$

$$\text{subject to} \quad \mathbf{m}_{lower} < \mathbf{m} < \mathbf{m}_{upper}$$

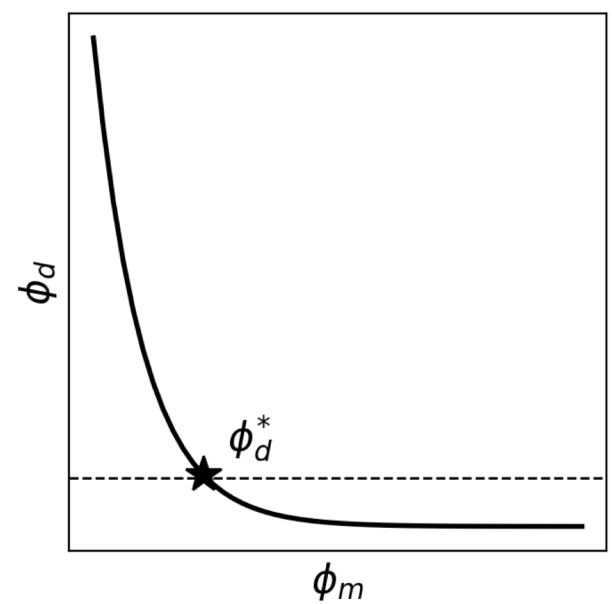
Data misfit

$$\phi_d(\mathbf{m}) = \frac{1}{2}||\mathbf{W}_\mathrm{d}(F[\mathbf{m}] - \mathbf{d}_\mathrm{obs})||_2^2.$$

Regularization

$$\phi_m(\mathbf{m}) = \frac{1}{2}||\mathbf{W}_\mathrm{m}(\mathbf{m} - \mathbf{m}_\mathrm{ref})||_2^2.$$

Tikhonov curve



Gauss-Newton approach

- Inverse problem

$$\min_{\mathbf{m}} \phi(\mathbf{m}) = \phi_d(\mathbf{m}) + \beta\phi_m(\mathbf{m})$$
$$= \frac{1}{2}\|\mathbf{W}_d(F[\mathbf{m}]) - \mathbf{d}^{obs}\|^2 + \frac{\beta}{2}\|\mathbf{W}_m(\mathbf{m} - \mathbf{m}_{ref})\|^2$$

- Gradient

$$\mathbf{g}(\mathbf{m}) = \mathbf{J}^\top \mathbf{W}_d^\top \mathbf{W}_d(F[\mathbf{m}] - \mathbf{d}^{obs}) + \beta \mathbf{W}_m^\top \mathbf{W}_m(\mathbf{m} - \mathbf{m}_{ref})$$

- Taylor expand: Gauss Newton equation

$$(\mathbf{J}^\top \mathbf{W}_d^\top \mathbf{W}_d \mathbf{J} + \beta \mathbf{W}_m^\top \mathbf{W}_m)\delta\mathbf{m} = -\mathbf{g}(\mathbf{m})$$

- Use inexact PCG to solve for model update (N_{CG} iterations)

$$\mathbf{m}_{k+1} = \mathbf{m}_k + \delta\mathbf{m}$$

Number of forward modellings: $2(N_{CG} + 1) \sim 20$

Gauss-Newton approach

$$\min_{\mathbf{m}} \phi(\mathbf{m}) = \phi_d(\mathbf{m}) + \beta\phi_m(\mathbf{m})$$

$$= \frac{1}{2}\|\mathbf{W}_d(F[\mathbf{m}]) - \mathbf{d}^{obs}\|^2 + \frac{\beta}{2}\|\mathbf{W}_m(\mathbf{m} - \mathbf{m}_{ref})\|^2$$

Choose β_0, $\mathbf{m}_{ref}$

Evaluate $\boldsymbol{\phi}(\mathbf{m}_{\mathbf{ref}})$, $\mathbf{g}(\mathbf{m}_{\mathbf{ref}})$, matrices $\mathbf{W}_{\mathbf{d}}$, **W...**

for i in range([0, max_beta_iter]):

for k in range([0, max_inner_iterations]):

- IPCG to solve $(\mathbf{J}^\top \mathbf{W}_d^\top \mathbf{W}_d \mathbf{J} + \beta \mathbf{W}_m^\top \mathbf{W}_m)\delta \mathbf{m} = -\mathbf{g}(\mathbf{m})$
- line search for step length α
- Update model $\mathbf{m}_{k+1} = \mathbf{m}_k + \alpha\delta\mathbf{m}$
- Exit if $\phi < \phi_d^*$ or $\frac{\|\mathbf{g}(\mathbf{m}_{k+1})\|}{\|\mathbf{g}(\mathbf{m}_k)\|} < \mathrm{tol}$

Reduce β

Tally up the computations

Number of transmitters	1000
Number of time steps	50
Solving a GN step	20
Number of GN iterations	20

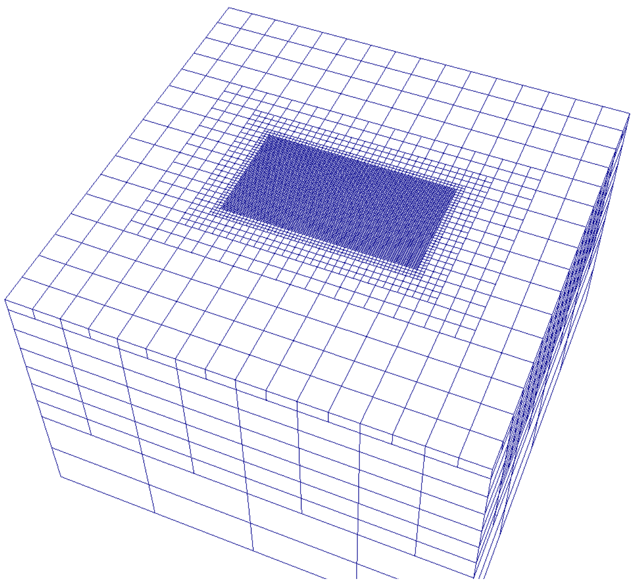
- Total number of Maxwell solutions is 20,000,000
- Suppose: t_{factor}=1 sec
  - 100 processors: 55 hours
  - 1000 processors 5.5 hours

Need:
- Fast forward modelling
- Multiple cpu

Mesh

- Trade off (accuracy vs. computation)
- Consider a 3D airborne EM simulation (1000 sources)

Octree mesh



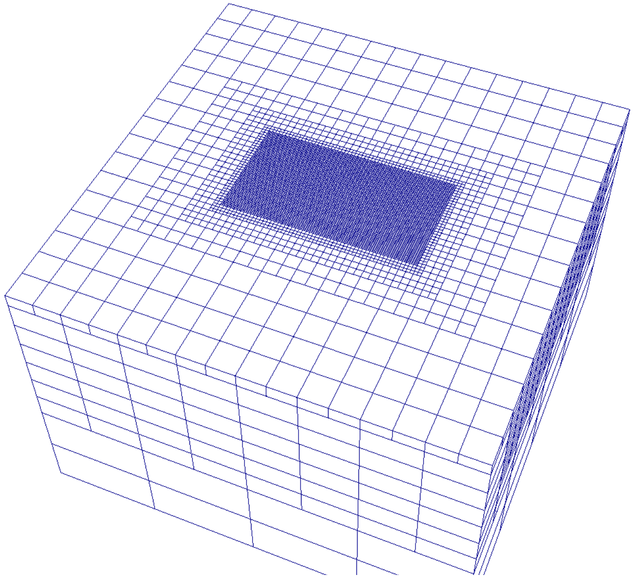
> 1,000,000 cells (this is big!)

How do we tackle this?

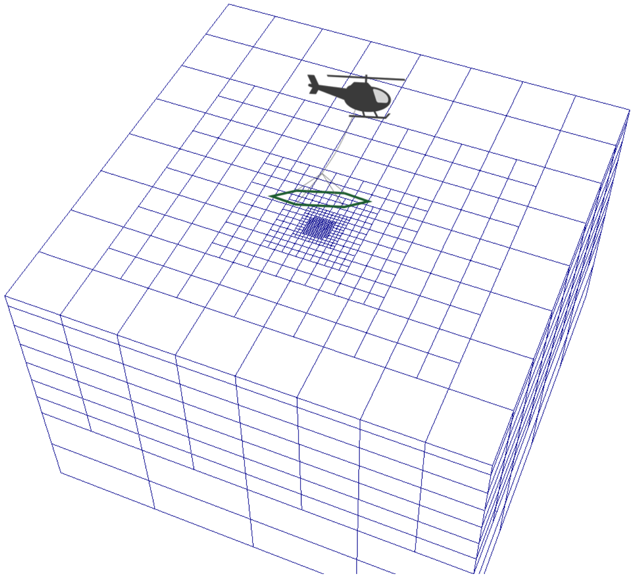
Mesh decomposition

- Separate forward modelling mesh for each transmitter

Global mesh



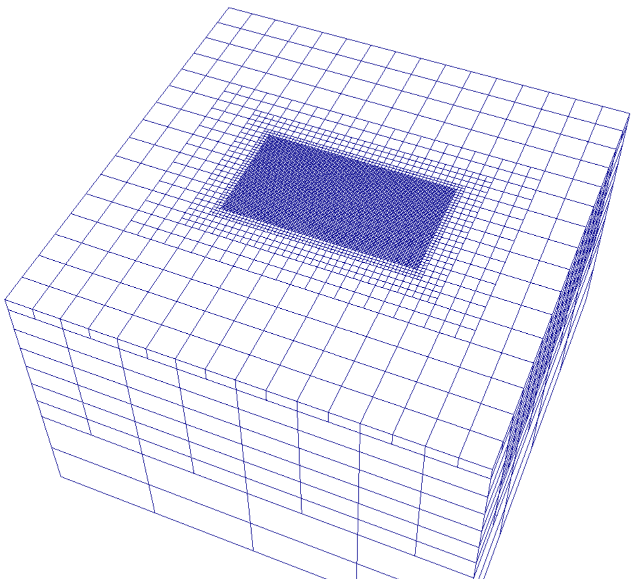
Local mesh



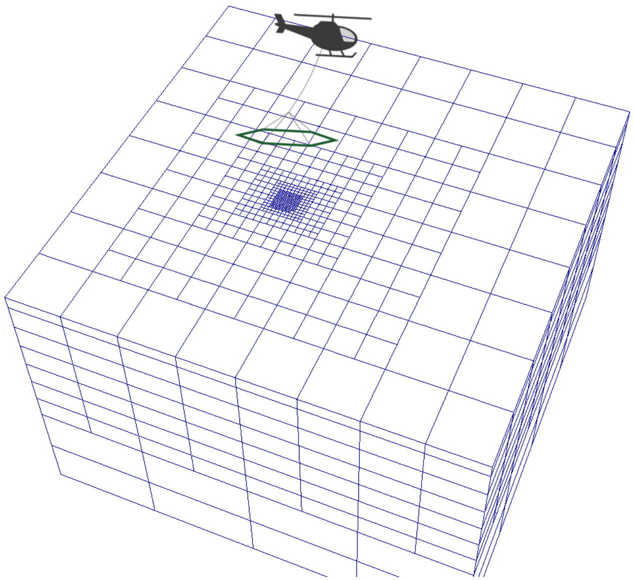
Mesh decomposition

- Separate forward modelling mesh for each transmitter

Global mesh



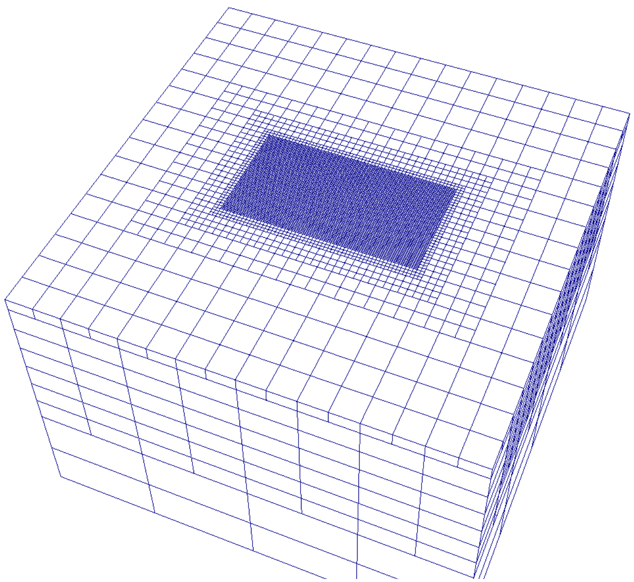
Local mesh



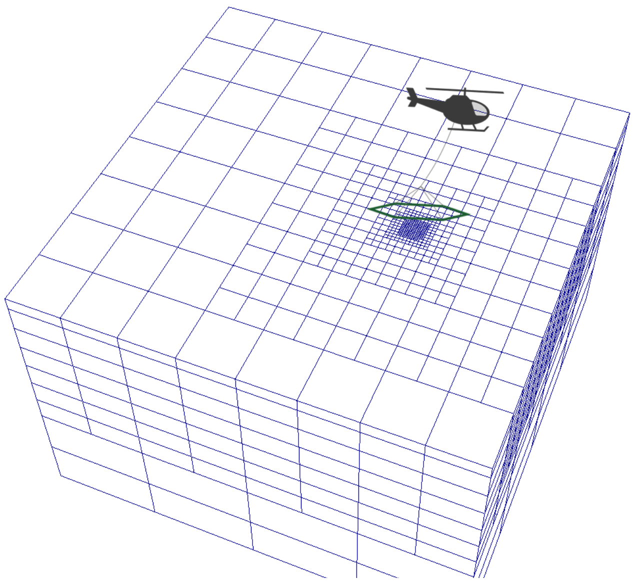
Mesh decomposition

- Separate forward modelling mesh for each transmitter

Global mesh

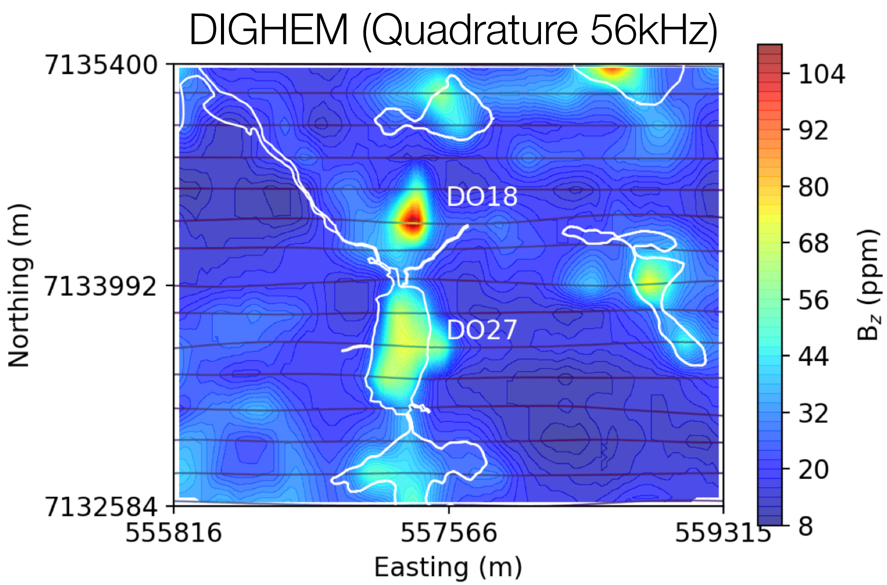


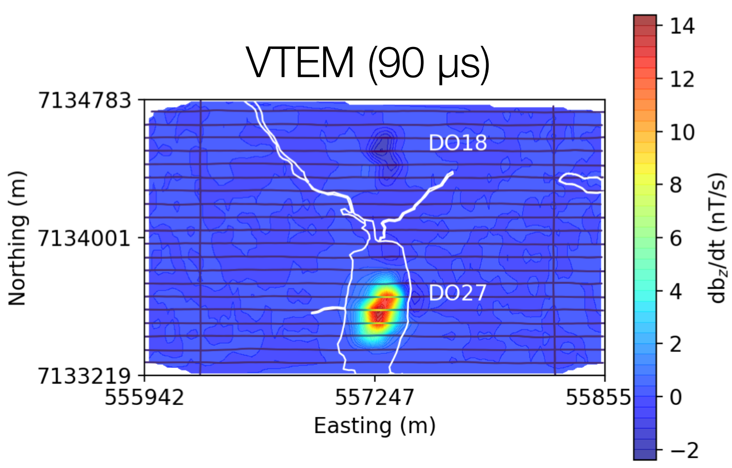
Local mesh



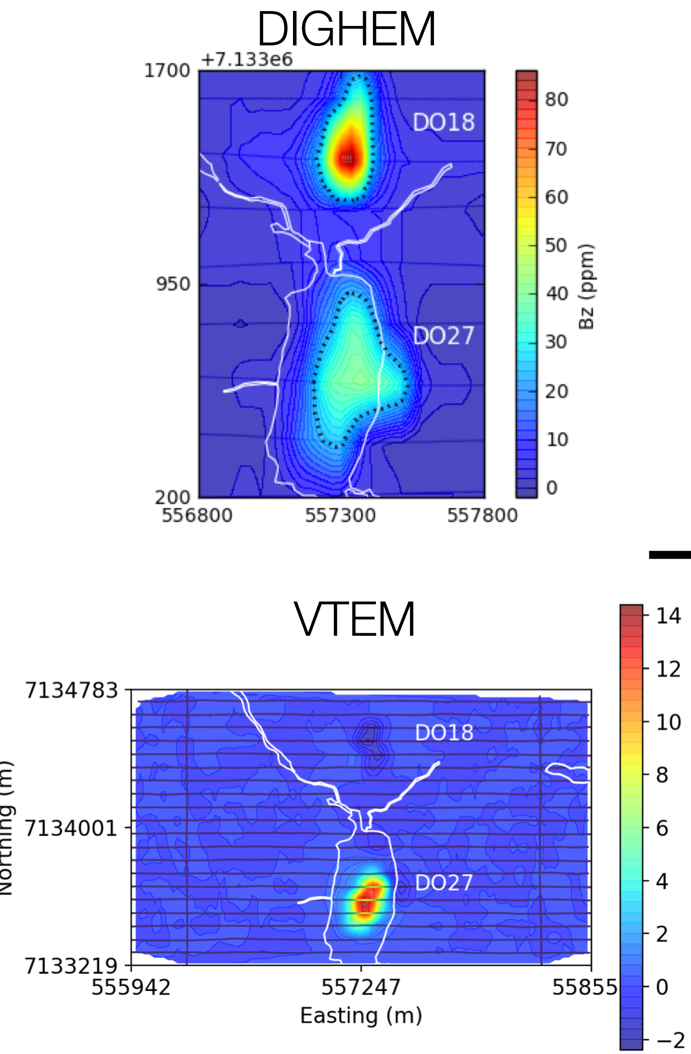
Advances

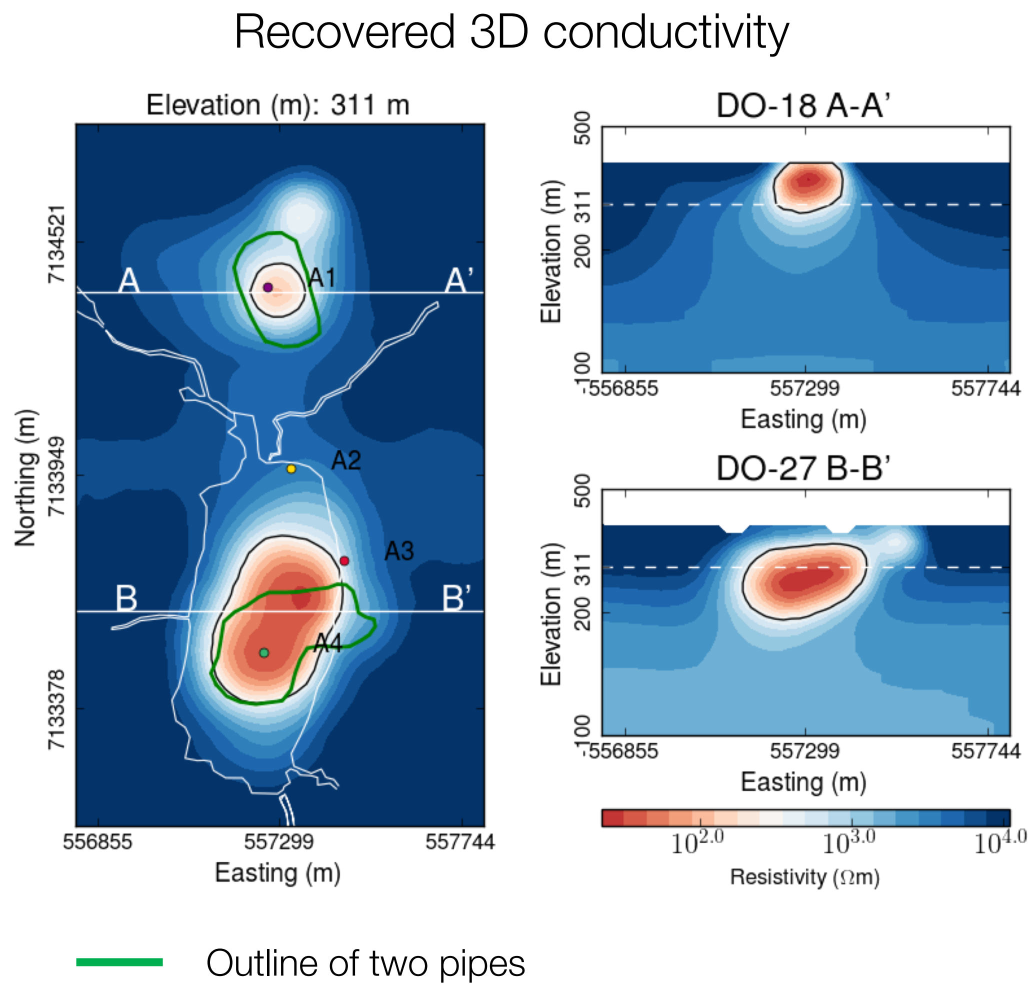
- Direct solvers (factor Maxwell operator)
- Semi-structured meshes (OcTree, reduce the # of variables)
- Separating forward and inverse meshes
- Handling the sensitivity matrix
- Access to multi-cores





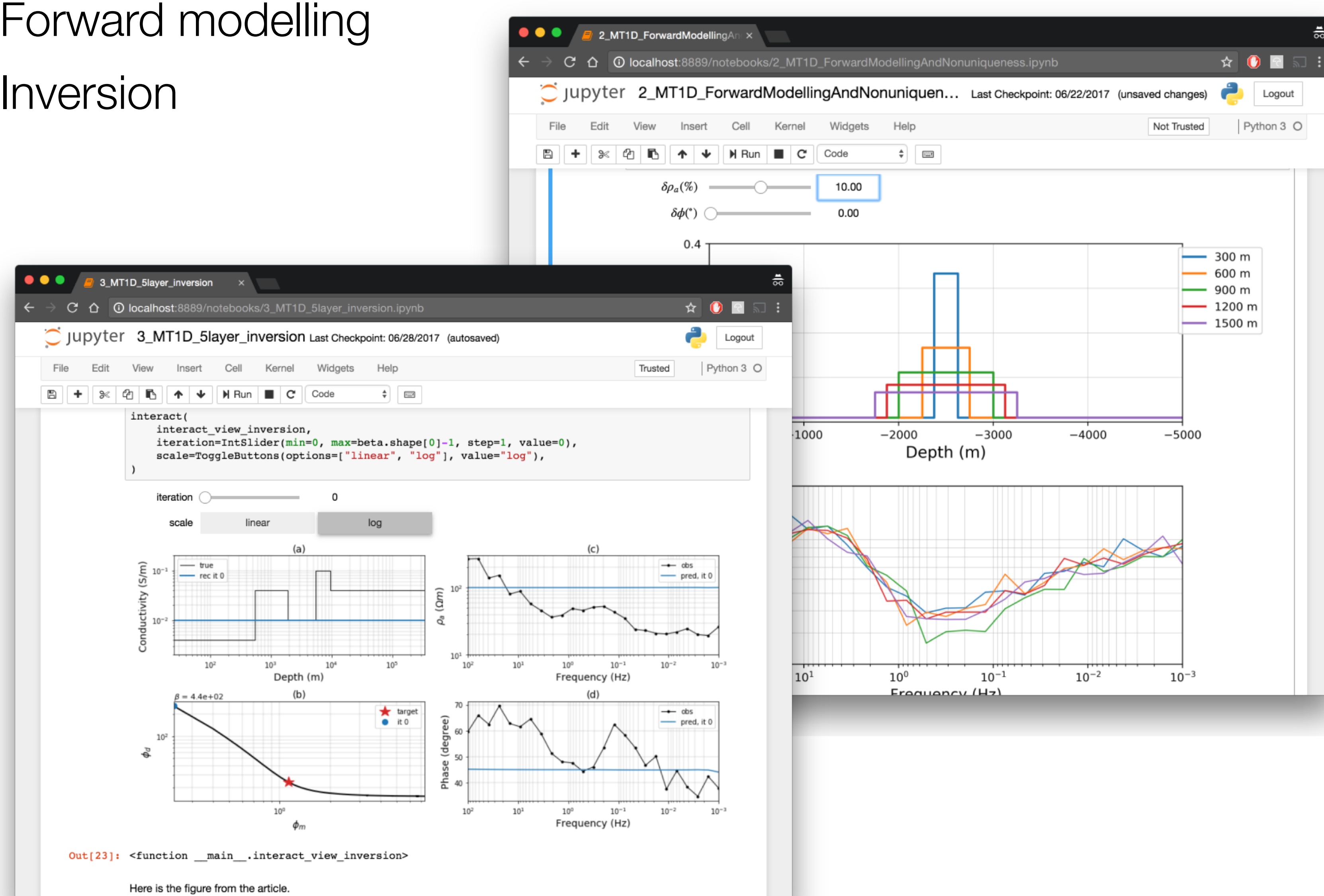
Example 3D inversion (TKC)





MT Tutorials

- Forward modelling
- Inversion



End of Inverse Theory