Tomography and Location

In this lecture, we discuss several aspects of the very general problem of INVERSION, based on examples of cross-well seismic travel-time tomography and earthquake location

- Forward and Inverse travel-time problems
- Seismic tomography
- Generalised Linear Inverse
- Least Squares inverse
    - Regularized, weighed, smoothed
- Iterative inverse
    - Back-projection method
- Resolution
- Statistical testing of results
- Location of seismic sources
- Data norms

- <u>Reading:</u>

Shearer, 5.6-5.7

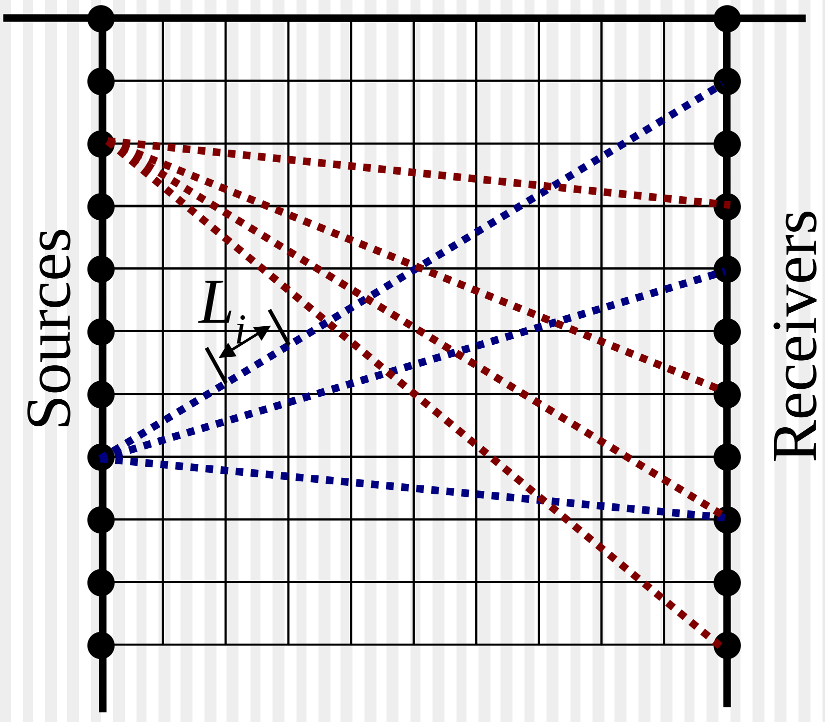
Seismic (velocity) tomography

- Tomography
  - The name derived from the Greek for “section drawing” - the idea is that the section appears *almost* automatically...
  - Using multitude of source-receiver pairs with rays crossing the area of interest.
  - Looking for an unknown velocity structure.
  - Depending on the type of recording used, it could be:
    - *Transmission tomography* (nearly straight rays between boreholes);
    - *Reflection tomography* (reflected rays; in this case, positions of the reflectors could be also found);
    - *Diffraction tomography* (using least-time travel paths according to Fermat rather than Snell's law; this is actually more a waveform inversion technique).

Cross-well tomography

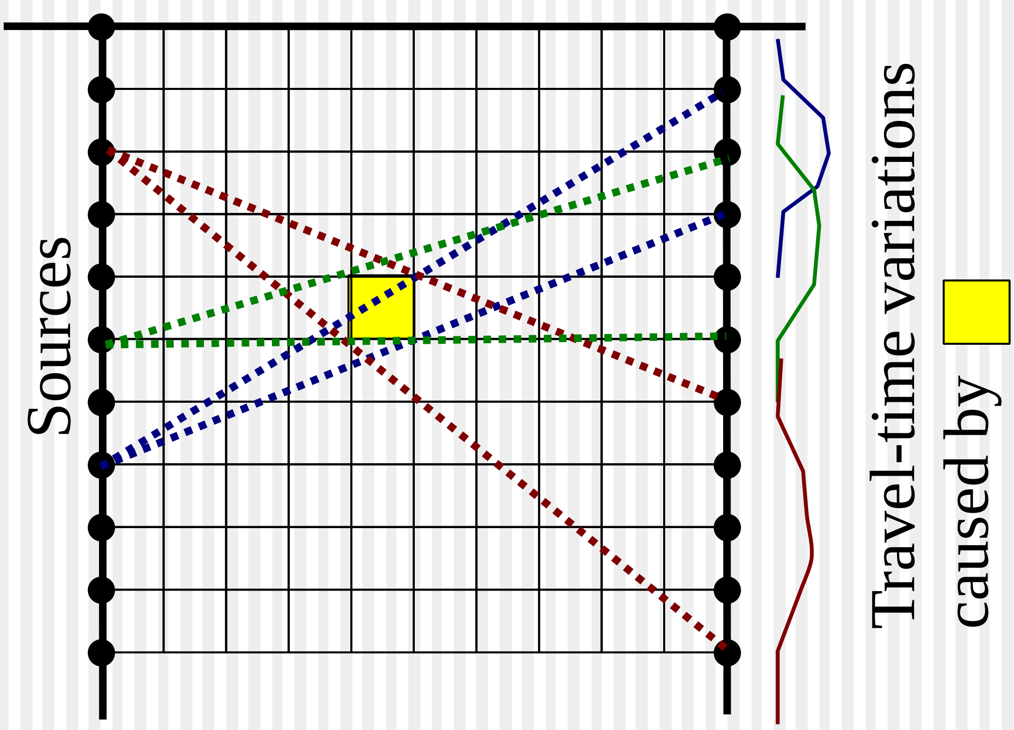
- Consider the case of transmission "cross-well" tomography
  - This is the simplest case – rays may be considered nearly straight, the data are abundant, and the coverage is *relatively uniform*

These are the three principal concerns in tomography:
1) linearity of the problem;
2) density of data coverage;
3) good azimuthal coverage.



Principle of travel-time tomography

- Velocity perturbations are considered as small
  - Therefore, rays are approximated as straight
- Each velocity cell leads to characteristic travel-time variations at the receivers ("impulse response")
  - These are inverted for velocity value at



Travel-time inversion as a *linear inverse problem*

- First, we parameterize the velocity model
  - Typically, the parameterization is a grid of constant-velocity blocks (sometimes continuous spline functions are used instead of the blocks).
  - This parameterization gives us a *model vector*, $\mathbf{m}$, consisting of slownesses in each cell:

$$\mathbf{m} = \begin{pmatrix} s_1 = 1/V_1 \\ s_2 = 1/V_2 \\ \cdots \\ s_{N_{\text{model}}} = 1/V_N \end{pmatrix}$$

- Second, we measure all available travel times and arrange them into a single data vector:

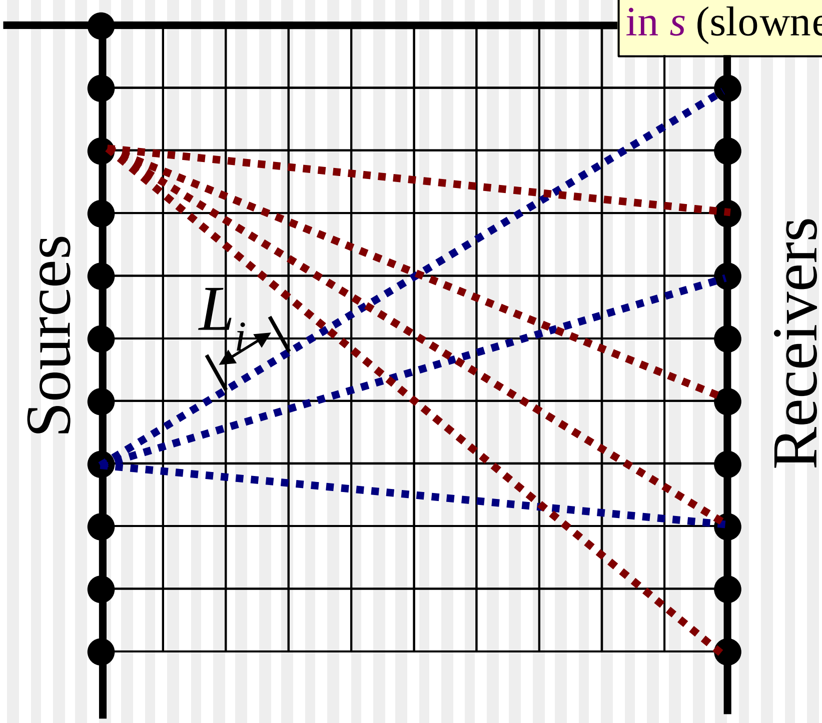
$$\mathbf{d} = \begin{pmatrix} t_1 \\ t_2 \\ \cdots \\ t_{N_{\text{data}}} \end{pmatrix}$$

Forward model

- Third, we formulate the *forward model* to predict **d** from **m**. To achieve this, we need to *trace rays* through the model and measure the length of every ray's segment in each model block, L_{ij}.
  - The travel time for *i*-th ray is then:

$$t_i = \sum_j L_{ij} \frac{1}{V_j} = \sum_j L_{ij} s_j .$$

Note that the expression is non-linear in V but linear in s (slowness).



Generalized Linear Inverse

- The model for travel times: $t_i = \sum_j L_{ij} s_j$ can be written in matrix form:

$$\mathbf{d} = \mathbf{L}\mathbf{m}$$

- Now, we want to substitute $\mathbf{d} = \mathbf{d}^{\text{observed}}$ and solve for unknown $\mathbf{m}$. This is called the *inverse problem*
- Typically, matrix $\mathbf{L}$ is not invertible (it is not square), and so it is inverted in some *generalized* (averaged, approximate) sense
- Any solution in the linear form

$$\mathbf{m} = \mathbf{L}_g^{-1}\mathbf{d}^{\text{observed}}$$

  is called the generalized linear inverse.
- The key idea of generalized inverse is that model m is sought as a linear combination (matrix product) of data values) ($\mathbf{d}^{\text{observed}}$, travel times in our case)
- The key problem is thus in finding a suitable form for $\mathbf{L}_g^{-1}$

Projection into model space

- Tomography problems are typically overdetermined (contain many more ray paths than grid model blocks)
- In such cases, the following approach to constructing $\mathbf{L}_g^{-1}$ works well:
  - multiply on the left by transposed $\mathbf{L}^{\mathrm{T}}$:

$$\mathbf{L}^T \mathbf{d}^{\text{observed}} = \mathbf{L}^T \mathbf{L} \mathbf{m}$$

This operation "back-projects" the redundant data onto model space

  - The matrix $\mathbf{L}^T\mathbf{L}$ is square and often invertible
  - By inverting matrix $\mathbf{L}^T\mathbf{L}$, we find solution giving m is a product of data $\mathbf{d}$ with a matrix:

$$\mathbf{m} = \left(\mathbf{L}^{\mathrm{T}}\mathbf{L}\right)^{-1} \mathbf{L}^T \mathbf{d}^{\text{observed}}$$

This is the "least-squares" solution
It is used in the well-known GLI3D program for refraction statics

Least Squares Inverse

- Note that the solution is a linear combination of data values:

$$\mathbf{m} = \left(\mathbf{L}^T\mathbf{L}\right)^{-1}\mathbf{L}^T\mathbf{d}^{\mathrm{observed}} = \mathbf{L}_g^{-1}\mathbf{d}^{\mathrm{observed}}$$

$$\mathbf{L}_g^{-1} = \left(\mathbf{L}^T\mathbf{L}\right)^{-1}\mathbf{L}^T$$

This is the generalized inverse for LEAST SQUARES method

- The reason for its name of "Least Squares" is in minimizing the mean square of data misfit function $\Phi(\mathbf{m})$:

$$\Phi(\mathbf{m}) = \left(\mathbf{d}^{\mathrm{observed}} - \mathbf{Lm}\right)^T\left(\mathbf{d}^{\mathbf{observed}} - \mathbf{Lm}\right)$$

- Exercise: show this!

Hints:

1) Write the misfit above in subscript form, as function of multiple variables m_i:

$$\Phi(\mathbf{m}) = \left(d_i^{observed} - L_{ij}m_j\right)\left(d_i^{observed} - L_{ik}m_k\right)$$

Summations over repeated indices implied!

2) Write equations for minimizing the misfit:

$$\frac{\partial\Phi}{m_l} = 0$$

3) Present these equations back in matrix form.

Damped Least Squares

- Sometimes the matrix $\mathbf{L}^T\mathbf{L}$ is singular and its inverse does not exist or unstable.
  - This happens, *e.g.*, when:
  1) Some model cells are not crossed by any rays, or
  2) There are groups of cells traversed by the same rays only.
- In such cases, the inversion can be *regularized* by adding a small positive diagonal term to $\mathbf{L}^T\mathbf{L}$:

$$\mathbf{m} = \left(\mathbf{L}^T\mathbf{L} + \varepsilon\mathbf{I}\right)^{-1} \mathbf{L}^T \mathbf{d}^{\text{observed}}$$

  - This is also a generalized inverse. This form of inverse is called the *Damped Least Squares* solution.
  - In this solution, ε is chosen such that stability is achieved and the non-zero contributions in $\mathbf{L}^T\mathbf{L}$ are affected only slightly.

Weighted Least Squares

- Often, different types of data are included in $\mathbf{d}$
  - For example, different travel times, t_i, may be measured with different uncertainties δt_i
- In such cases, it is useful to apply weights to the equations:

$$\mathbf{Wd} = \mathbf{WLm}$$

where $\mathbf{W}$ is a diagonal weight matrix:

$$\mathbf{W} = \mathrm{diag}\left(\frac{1}{\delta t_1}, \frac{1}{\delta t_2}, \frac{1}{\delta t_3}, \ldots\right)$$

- This weight matrix simply means that each equation for travel time t_i is multiplied by $1/\delta t_i$. As a result uncertainties of scaled data in each equations become equal 1, and they should have equal contributions to the resulting model

$$\mathbf{m} = \mathbf{L}_g^{-1}\mathbf{d}^{\mathrm{observed}}$$

Weighted Least Squares (cont.)

- This corresponds to a modified least-squares misfit function:

$$\Phi(\mathbf{m}) = \left(\mathbf{d}^{\text{observed}} - \mathbf{Lm}\right)^T \mathbf{W}^T \mathbf{W} \left(\mathbf{d}^{\text{observed}} - \mathbf{Lm}\right)$$

and solution:

$$\mathbf{m} = \mathbf{L}_g^{-1} \mathbf{d}^{observed}$$

$$\mathbf{L}_g^{-1} = \left(\mathbf{L}^T \mathbf{W}^T \mathbf{W} \mathbf{L} + \varepsilon \mathbf{I}\right)^{-1} \mathbf{L}^T \mathbf{W}$$

Smoothness constraints

- When using finely-sampled models...
  - some cells may be poorly constrained;
  - solutions can become 'rough' (highly variable, noisy – see below)
- To remove roughness, additional 'smoothness constraint' equations can be added
  - These equations will be additional rows in **L**, for example:

$$m_i = \text{Average\_of\_some\_adjacent\_points\_}\left(m_j\right)$$

This equation makes the inverse favor models in which model slowness m_i is close to adjacent points

- Zero Laplacian:

$$\nabla^2 m_i = 0$$

Recall that Laplacian of a function is the sum of second derivatives. These derivatives are small in a smooth model:

$$\nabla^2 f \equiv \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2}$$

- These equations must be used with small *weights* w, which are often tricky to select

Simple Iterative Inverse

- Sometimes matrix $\mathbf{L}^{\mathrm{T}}\mathbf{L}$ is also too large to invert, or even to store
- It can the be approximated by its diagonal:

$$m = \left[\mathrm{diag}\left(\mathbf{L}^T \mathbf{L} \right) + \varepsilon \mathbf{I} \right]^{-1} \mathbf{L}^T \mathbf{d}^{\mathrm{observed}}$$

  - The diagonal only contains one value per model cell (sum of squared distances for all rays crossing it)
  - Contributions to $\mathbf{m}$ can be evaluated during a pass through all data and without storing matrices $\mathbf{L}$ or $\mathbf{L}^T\mathbf{L}$
- Variants of this method are known as:
  - Back-projection method;
  - SIRT (Simultaneous Iterative Reconstruction technique)
  - ART (Algebraic Reconstruction Technique)

Simple Iterative Inverse

(how it works)

- Iteration to reduce data error:

Travel times in "background model"

$$\delta_1 \mathbf{d} = \mathbf{d}^{\text{observed}} - \mathbf{d}_0$$

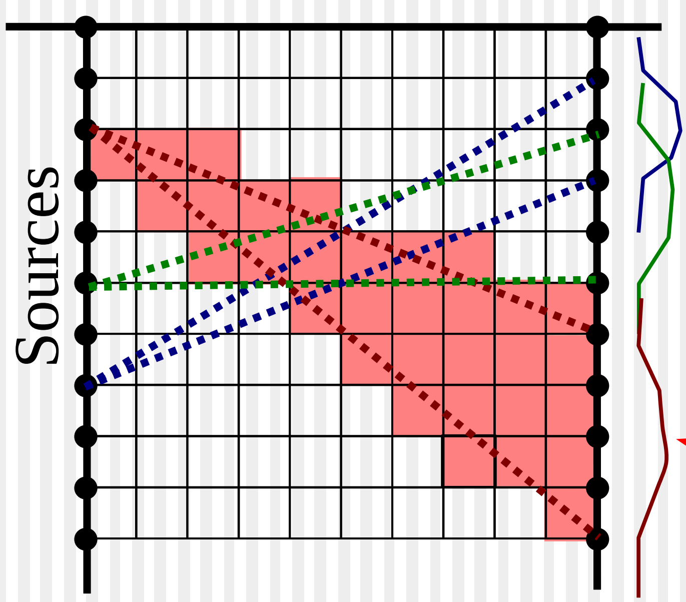
$$\delta_1 \mathbf{m} = \mathbf{L}_g^{-1} \delta_1 \mathbf{d}$$

Approximate inverse of any kind

$$\delta_2 \mathbf{d} = \delta_1 \mathbf{d} - L \delta_1 \mathbf{m}$$

$$\delta_2 \mathbf{m} = \mathbf{L}_g^{-1} \delta_2 \mathbf{d}$$

...



For each ray, the observed travel-time perturbation is thus "back-projected" into the slowness model

Resolution matrix

- For any form of the inverse, assessment of the *quality of inversion method* is often done by using the *Resolution Matrix:*

$$\mathbf{R} = \mathbf{L}_g^{-1}\mathbf{L}$$

- The resolution matrix can be understood like this:

    1) To obtain j^{th} column of the resolution matrix, perturb j^{th} parameter (slowness value) of a zero model by a unit value (e.g., 1 s/m for slowness). Let us denote this perturbed model $\mathbf{m}_{\text{test}}^{j}$
    2) Perform forward modeling (generate synthetic data);
    3) Perform the inverse. The result of this inversion will be

$$\mathbf{m}_{\text{test}}^{j\ \text{reproduced}} = \mathbf{L}_g^{-1}\mathbf{L}\mathbf{m}_{\text{test}}^{j} = \mathbf{R}\mathbf{m}_{\text{test}}^{j}$$

    This is the j^{th} column of matrix **R.**

- Thus, j^{th} column in matrix **R** shows how the j^{th} cell is reproduced by the inversion. Ideally, cell j should be reproduced perfectly (with value $R_{jj} = 1$), and other R_{ij} should equal zero (cell j should not be misrepresented as different "i" after inversion).
- Note that **R** *does not* depend on data values but depends on sampling (matrix **L**)

Checkerboard resolution test

- Test of the resolution in the model when computation of the *Resolution Matrix* is impossible or impractical
- Method:
  - Create an artificial model perturbation in the form of alternating positive and negative anomalies ("checkerboard")
  - Predict the data in this model:

$$\mathbf{d}' = \mathbf{L}\mathbf{m}_{\mathrm{checker}}$$

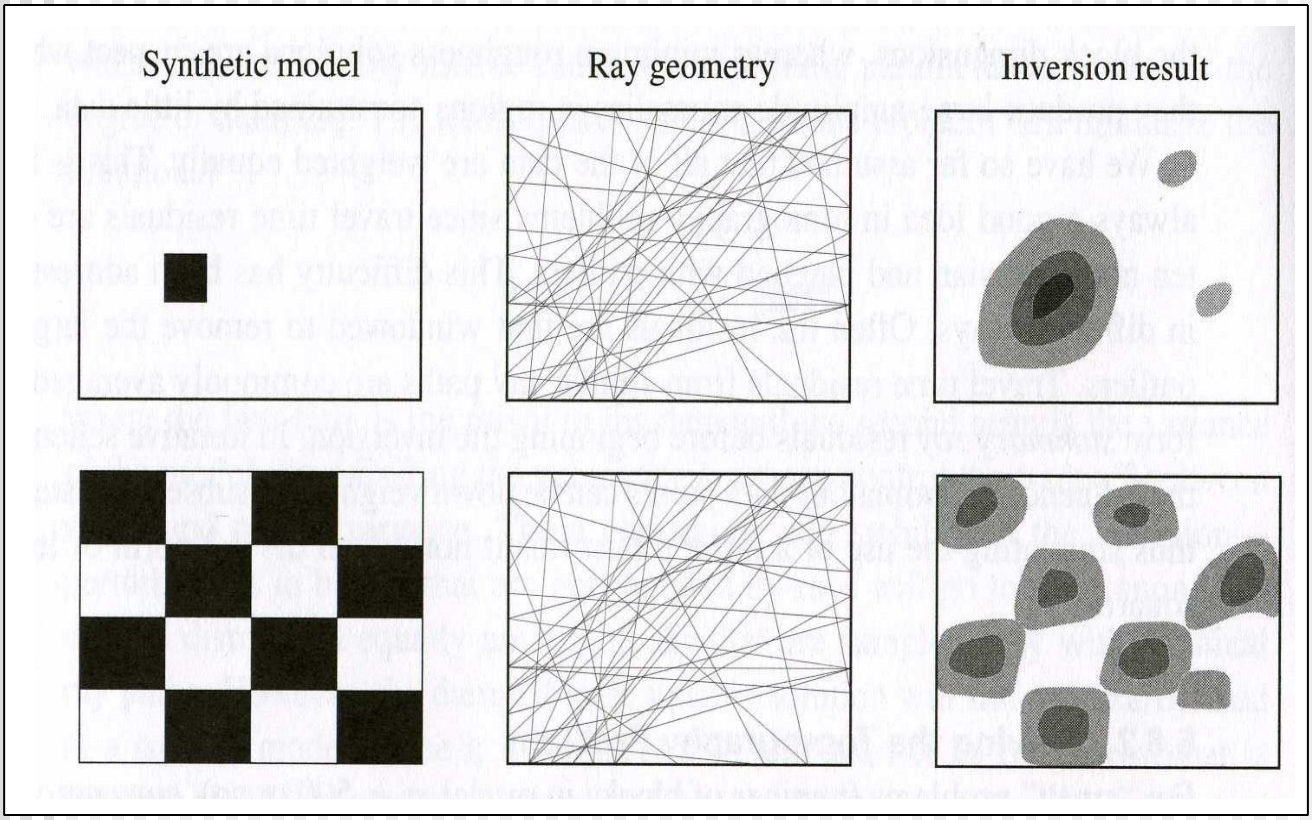
  - Invert the resulting synthetic data:

$$\mathbf{m}' = \mathbf{L}_g^{-1}\mathbf{d}' = \mathbf{L}_g^{-1}\mathbf{L}\mathbf{m}_{\mathrm{checker}}$$

  - Compare the result to the input model
    - The ability to reproduce the input "checker" anomalies indicates the quality of inversion
    - This quality varies within different parts of the model

Checkerboard resolution test (cont.)

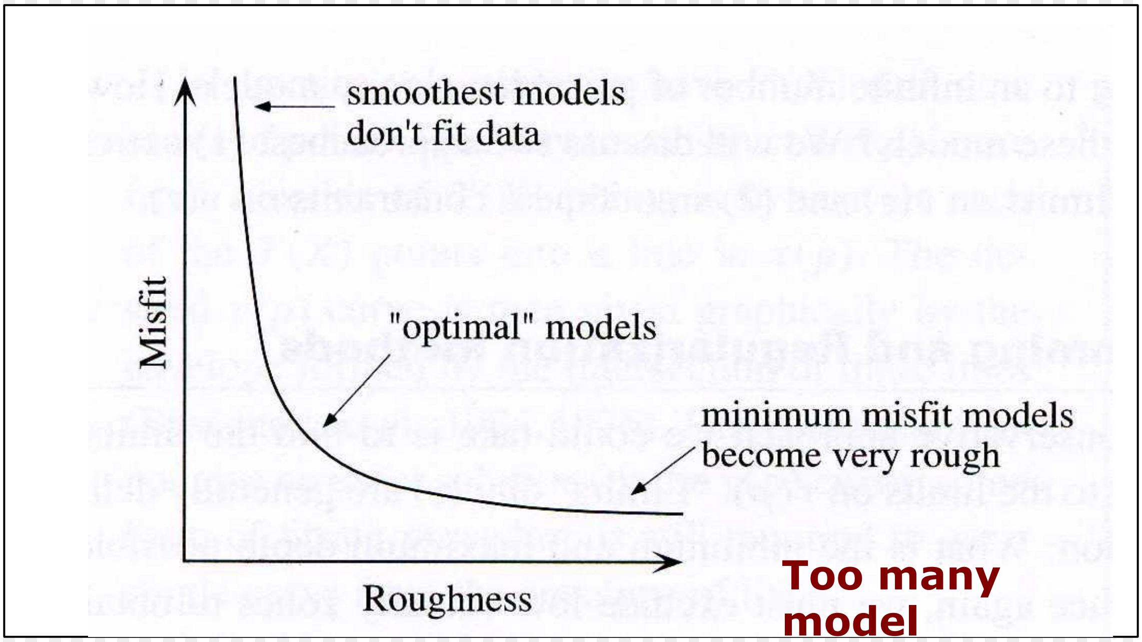
- Schematic example from travel-time tomography:



Trade-off between data fit and model simplicity

- Too simple models often cannot explain the data
- However, excessively detailed models are also not good:
  - They can “over-fit” the data (fit travel times too much, better than warranted by errors in picking the times)
  - Model complexity may be spurious and caused by data noise
- We need to look for “optimally” complex models

Too few model parameters



Too many model parameters

Test for statistical significance of data fitting

- How can we verify that the model fits the data within reasonable error?
  - Complex models (with large numbers of unknowns) would often fit the data well;
  - Because the data contains *noise*, we should not over-fit the data!
- The χ^2 test is commonly used to determine whether the remaining data misfit is likely to be random:

$$\chi^2 = \frac{\sum_{i=1}^{N} \left(t_i - t_i^{\text{observed}} \right)^2}{\sigma^2}$$

  - Here, σ is the estimated data-measurement uncertainty
  - This uncertainty needs to be somehow measured from the data, prior to inversion (see eq. 5.31 in Shearer)

χ^2 test (cont.)

- The p.d.f. of χ^2 is controlled by the "number of data degrees of freedom" in the model:

$$N_{\mathrm{df}} = N_{\text{travel times}} - N_{\text{model parameters}}$$

  - this value means the number of travel times (constraints) not already satisfied by solving for model parameters
- For a given N_{df}, tabulated percentage points of p.d.f.(χ^2) can be used to determine whether the residual data misfit is likely to be random:

.

N_{df}	At 5%	At 50%	At 95%
5	1.15	4.35	11.07
10	3.94	9.34	18.31
20	10.85	19.34	31.41
50	34.76	49.33	67.5
100	77.03	99.33	124.34

  - The 95-% level is commonly used

χ^2 test (cont.)

- Here is how the χ^2 test is conducted (see lab #2):

1) Estimate measurement error σ for your data (travel times);
2) For a given model, calculate data errors (data minus data predicted by the model);
3) Divide the errors by σ, square, and sum to produce the χ^2 quantity ("statistic");
4) Determine N_{df};
5) For this N_{df}, look up in the table on the preceding slide the expected value of χ^2 at 95% confidence. Let us denote this value $\chi^2_{95\%}$.
6) Check how your χ^2 from the data and model compares to $\chi^2_{95\%}$:
   - If $\chi^2 > \chi^2_{95\%}$, your model poorly explains data; you need to increase the detail in the model;
   - If $\chi^2 \ll \chi^2_{95\%}$, the model is overfitted and likely "rough". Reduce model detail.
   - If $\chi^2 < \chi^2_{95\%}$ by not much, the model is good, and the errors are random (with 95% confidence).

Source Location Problem

- When using a natural (impulsive) source, its location can also be determined by a similar approach.
  - This method is used for locating earthquakes worldwide
  - For monitoring creep of mine walls (potash exploration)
  - Monitoring reservoirs during injection (Weyburn)

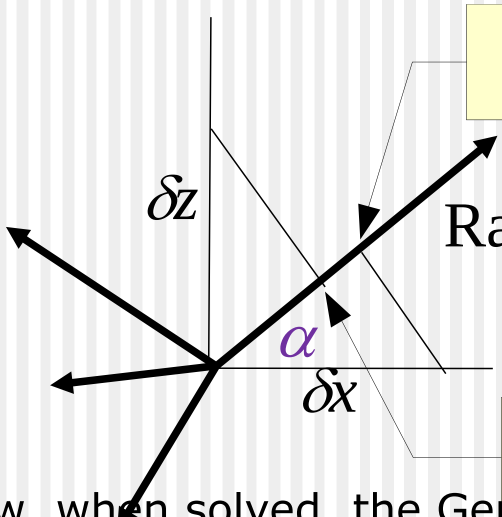
- To solve this problem, we:
  - Start from some reasonable approximation for source coordinates and solve the velocity tomography problem.
  - Include the coordinates and time of the source in model vector $\mathbf{m}$:

$$\mathbf{m} = \begin{pmatrix} s_1 = 1/V_1 \\ s_2 = 1/V_2 \\ \ldots \\ s_N = 1/V_N \\ x_{source} \\ z_{source} \\ t_{source} \end{pmatrix}.$$

(In two dimensions)

Source Location (cont.)

- Include into the matrix $\mathbf{L}$ time delays associated with shifting the source by δx or δz:



$$\delta t = \frac{-\cos\alpha}{V}\delta x \equiv L_{i,N+1}\delta x$$

$$\delta t = \frac{-\sin\alpha}{V}\delta z \equiv L_{i,N+2}\delta z$$

  - Now, when solved, the Generalized Inverse will yield the corrections to the location (δx, δz).
- This process is <u>iterated</u>: with the new source location, velocities are recomputed, and sources relocated again, etc.
  - Iterations are needed because ray shapes change after we shift the source and modify velocities (rays are not straight!)

Measures of data misfit ("data norms")

- The Least-Squares norm (called "L2") can be highly sensitive to data outliers:

$$\varepsilon_{L2} = \sum_{i=1}^{N} \left(t_i - t_i^{\text{observed}} \right)^2$$

  - However, it is the easiest to use (only for this norm, L^{-1}_g exists).

- Other useful norms:

  - L_n norms: $\varepsilon_{L_n} = \sum_{i=1}^{N} \left| t_i - t_i^{\text{observed}} \right|^n$

  - L_∞ norm: $\varepsilon_{L_\infty} = max_i \left(\left| t_i - t_i^{\text{observed}} \right| \right)$

- The "L_1" norm is less sensitive to outliers (*i.e.*, anomalous errors), and therefore also often preferred:

$$\varepsilon_{L_1} = \sum_{i=1}^{N} \left| t_i - t_i^{\text{observed}} \right|$$

L_1-norm inversion

- Solutions minimizing L_1 and similar norms are derived from L_2 by *iterative reweighting*:

  1) Use the least-squares inverse to minimize

  $$\varepsilon_{L2} = \sum_{i=1}^{N} \left(t_i - t_i^{\text{observed}} \right)^2$$

  2) Apply weights based on current data errors:

  $$W_i = \frac{1}{\sqrt{\left| t_i - t_i^{\text{observed}} \right|}}$$

  - The misfit then approximates ε_{L1}:

$$\text{Weighted } \varepsilon_{L2} = \sum_{i=1}^{N} W_i^2 \left(t_i - t_i^{\text{observed}} \right)^2 \approx \sum_{i=1}^{N} \left| t_i - t_i^{\text{observed}} \right| = \varepsilon_{L1}$$

  3)<u>Iterate</u> to converge to L1 solution