00

A Very Short Course in Seismic Tomography

Excerpts (shortened and modified) from
G. Nolet, A Breviary of Seismic Tomography
(Cambridge University Press, 2008)

Guust Nolet

October 26, 2012

Contents

Chapter 1

Ray tracing

To find the correct geometry of a ray in realistic models of the Earth or Sun, we need to trace rays following Snel's law. This is comparatively easy in the case of layered or spherically symmetric media. On the other hand, if the seismic velocity is also a function of one or two horizontal coordinates, it may be very difficult. Fortunately, Fermat's Principle allows us often to use background models with lateral homogeneity. This principle states that we make only a second order error if we compute the travel time along a trajectory that is different from the actual minimum time path. Since it is much easier to find the raypath for a layered model than for a 3D model, we ignore the fact that the ray does not exactly follow the minimum time path for the 3D Earth. By assuming the raypath to be insensitive to the 3D anomalies imposed by a tomographic inversion, we effectively render this inversion linear.

In extreme cases, however, the seismic velocities may change sufficiently fast that the ray computed for a layered Earth is too far away from the ray in the true, heterogeneous Earth. In that case we must use full 3D ray tracing. In this chapter we take a look at the most promising algorithms available for both cases but warn the reader that accurate ray tracing in 3D is still an active area of research that has not yet converged to one 'ideal' method. In fact all methods still have shortcomings.

1.1 The shooting method

To find the correct ray geometry between a given source and receiver location we must find a way to determine which initial condition (ray orientation at the source) satisfies the end condition (ray arriving in the receiver). The term 'shooting' refers to the latter: we aim, compute, and aim again until we 'hit' the receiver.

Seismic waves follow raypaths just like optical rays and satisfy Snell's law:

$$\frac{sini}{c} = \text{constant} = p \tag{1.1}$$

where i is the angle of the ray with the vertical, c is the speed of seismic waves in the rock, and the constant p is known as the *ray parameter*. The ray parameter has units of s/km and is also known as the *horizontal slowness*. Differentiating Snell's law with respect to the distance s along the ray path and applying simple geometry we find for a ray in the $x - z$ plane:

$$\begin{aligned}
\frac{di}{ds} &= p\,\frac{dc}{\cos i\, ds} = p\,\frac{dc}{dz} && (1.2)\\
\frac{dz}{ds} &= \cos i && (1.3)\\
\frac{dT}{ds} &= c^{-1}\,. && (1.4)
\end{aligned}$$

Thing are a little more complicated in spherical coordinates. We recall the equations for a ray in the equatorial plane $\theta = \pi/2$:

$$\frac{di}{ds} = \frac{\sin i}{c}\left(\frac{dc}{dr} - \frac{c}{r}\right) \tag{1.5}$$

$$\frac{dr}{ds} = \cos i \tag{1.6}$$

$$\frac{d\phi}{ds} = \frac{\sin i}{r} \tag{1.7}$$

$$\frac{dT}{ds} = c^{-1}\,. \tag{1.8}$$

In the case of elastic waves, the velocity c may denote either V_P or V_S.

This system is of the form $d\boldsymbol{v}/ds = \boldsymbol{F}[\boldsymbol{v}]$ for a 'vector' $\boldsymbol{v} = (i, r, \phi, T)$ that is a function of the distance s along the path and can be solved using well established numerical methods for the solution of ordinary differential equations. The shooting method consists of trying out starting values for the angle i_0 at the source until the ray crosses the receiver location at $r = a$, $\phi = \phi_r$ within a specified precision. Subroutines solving the differential equations often assume the source to be located at $\phi = 0$ in the equatorial plane, in which case the longitudinal coordinate ϕ is equivalent to the epicentral distance Δ. Because of the spherical symmetry, we can simply rotate the ray to fit a specific source-receiver pair.

The epicentral distance Δ between a receiver at longitude φ_r and co-latitude[1] ϑ_r and a source at (φ_s, ϑ_s) is found from the dot product between the (unit) vectors $\hat{\boldsymbol{r}}_r$ and $\hat{\boldsymbol{r}}_s$ that point from the Earth's centre to the receiver and source location, respectively:

$$\hat{\boldsymbol{r}}_s \cdot \hat{\boldsymbol{r}}_r = \cos\Delta\,.$$

With the expression for spherical coordinates

$$\begin{aligned} x &= r\sin\varphi\sin\vartheta \\ y &= r\cos\varphi\sin\vartheta \\ z &= r\cos\vartheta\,, \end{aligned}$$

we can work out the dot product $\hat{\boldsymbol{r}}_s \cdot \hat{\boldsymbol{r}}_r$ with $r = 1$ and obtain:

$$\cos\Delta = \cos\vartheta_r\cos\vartheta_s + \sin\vartheta_r\sin\vartheta_s\cos(\varphi_r - \varphi_s)\,. \tag{1.9}$$

More than one ray may arrive in a receiver. A suitable strategy is to compute a table with epicentral distances for a specific source depth, with dynamic increments in incidence angle Δi such that $\Delta\phi < \epsilon$, where ϵ is a specified precision, e.g. $0.3°$. A Newton interpolation scheme is usually effective in quickly locating the ray arriving at $\phi_1 < \phi < \phi_2$ by iterating:

$$i_{\text{next}} = i_1 + \frac{\phi - \phi_1}{\phi_2 - \phi_1}(i_2 - i_1), \tag{1.10}$$

and narrowing down the interval (i_1, i_2) such that the interval (ϕ_1, ϕ_2) always contains ϕ (see Fig. 1.1). Note that ϕ is not necessarily a monotonously decreasing function of i, and that the iteration may fail if the interval contains a point where $d\phi/di = 0$. In tomographic inversions with 10^6 or more data it is almost inevitable that some rays fail to converge. Rejecting them usually has no significant effect on the resulting image, while trying to salvage them may consume an inordinate amount of time.

[1]The co-latitude is the spherical coordinate, i.e. measured from the North Pole ($\vartheta = 0$) on a spherical Earth. Thus colatitude = $\pi/2-$ latitude. We denote geographical longitude and colatitude by (φ, ϑ) but use (ϕ, θ) for arbitrary spherical coordinate systems. The symmetry of the Earth provides extra flexibility. For body waves the coordinate system is usually rotated such that the ray is in the equatorial plane and starts in $(\phi = 0, \theta = \pi/2)$, so that ϕ equals epicentral distance. On the other hand, when we deal with normal modes we often define $\theta = 0$ as the source location, so that θ is the epicentral distance.

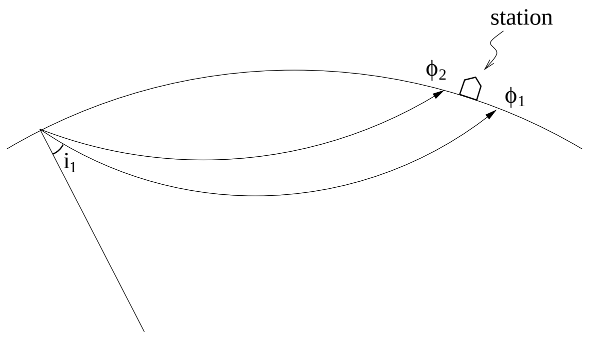


Figure 1.1: The shooting method: the first try uses angle i_1 but overshoots the seismograph in the station, whereas i_2 gives a ray that arrives at a distance that is too short compared to the epicentral distance ϕ of the station. The next try would be given by (1.10).

1.2 Ray bending

Tracing rays between fixed sources and receivers in three-dimensional media is harrowingly difficult. If the velocity c depends on the horizontal coordinates as well as on z or r, the shooting method usually fails so often to converge that it becomes useless. For this reason we prefer to use a ray bending method. As the name indicates, we set up a set of equations to 'bend' the ray until its travel time is stationary as prescribed by Fermat's principle.

The only method that is guaranteed to yield the fastest raypath in every circumstance is Dijkstra's method, first used in seismic tomography by Nakanishi and Yamaguchi [18] but better known in the efficient variant developed by Moser [16] as the 'shortest path method'. Unfortunately, the stability of the shortest path method is offset by two important disadvantages: its travel time is only approximate and in case more than one ray arrives in the receiver, only the fastest ray can be found. Whereas the first disadvantage can be mitigated using the bending methods that we shall describe, the second can only be circumvented if the later arrivals have clearly different properties from the first arrival, e.g. when they reflect from some surface and their path can be broken down in different components. Yet the shortest path method has its use in seismic tomography, because we often are only interested in the first arriving wave. For that reason we shall give a brief introduction to the basic principles of the method.

The shortest path method makes use of a weighted graph. Imagine that we place markers inside the Earth, much like road signs in a town, and we impose the restriction that a seismic ray can only travel by going from marker to marker. There are roads between the marker and several of its neighbouring markers, and the positive travel time between a marker and these neighbours is specified. In graph theory, the markers are called 'nodes' or 'vertices', the roads between a pair of markers are 'edges' or 'arcs'. The collection of all connected neighbours of a marker is denoted as its 'forward star'. The collection of all nodes constitutes a 'graph'. Suppose we wish to find the shortest travel time to go from a source node A to a receiver node B. We could try to find all permutations, compute the time needed, and select the shortest one (note that it makes no sense to visit one node twice, since eliminating the loop in between two visits will always result in a shorter time). This, however, would take too much time. A faster method was discovered by the Dutch mathematician Edsger Dijkstra (1930-2002).

Dijkstra's algorithm uses the fact that once the shortest path from the source node to a particular node is known, we can use this information when searching for the shortest path to other nodes. Let $\mathcal{Q}$ be the set of nodes for which we have not yet determined the shortest path, though some preliminary travel times may have been calculated already. We define the set $\mathcal{P}$ as the complement of $\mathcal{Q}$, i.e. it contains all nodes with known shortest paths to the source. Initially, all nodes are in $\mathcal{Q}$ with infinite travel time. The algorithm is then described by the following steps:

1. Set the travel time to the source node to 0.
2. If $\mathcal{Q}$ is empty, stop
3. Choose the node s with the smallest travel time in $\mathcal{Q}$

4. For each node i in the forward star of s, update the travel time such that $T_i \leftarrow \min\{T_i, T_s + \Delta T_{si}\}$
5. Move node s from set $\mathcal{Q}$ to set $\mathcal{P}$
6. Goto [2]

A simple Fortran implementation of the algorithm is available in the software repository[2]. Note that the 'path' itself is fully prescribed by assigning to each node the preceding node along its shortest path to the source. For a proof that the algorithm works, see a textbook in discrete mathematics, e.g. Johnsonbaugh [12]. The algorithm can be speeded up considerably by sorting the nodes in $\mathcal{Q}$ into a 'heap', which is defined as a set of N travel times T_j such that:

$$T_j \geq T_{j/2} \text{ for } 1 \leq j/2 < j \leq N\,.$$

A heap is a comparatively fast way to find the fastest path among a set of paths without having to sort the whole set. For details, including how to set up the heap in the first place, see Press et al. [20] or subroutine `cheap.f` in the software repository.

The precision of the shortest path algorithm depends on the complexity of the forward star, but is never very great. Because the ray is represented by straight line segments between nodes, its travel time is not very accurate and, because of Fermat's Principle, always an overestimate of the shortest time that is possible for an arbitrary ray. To obtain a more precise estimate of the travel time, we 'bend' the resulting ray to get rid of the angle discretization imposed by the structure of the forward star. Moser et al. [17] present a method to bend this ray, using the graph nodes as beta spline supports that can be moved around. An alternative (and sometimes stabler) bending can be effected by discarding the spline interpolation and using linear connections instead. The node locations x_i, y_i and z_i satisfy a set of linear equations if they are to minimize the total travel time, given by

$$T = \sum_{i=2}^{N} \frac{L_i}{\bar{c}_i}\,,$$

for a ray consisting of N segments of length L_i:

$$L_i = \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2 + (z_i - z_{i-1})^2}\,,$$

where we defined the average velocity $\bar{c}_i = \frac{1}{2}(c_i + c_{i-1})$ between two nodes (see Fig. 1.2). According to Fermat's principle $\partial T/\partial x_k = 0$ for $k = 2, ..., N-1$ (the end nodes are fixed), and similarly for y_k and z_k:

$$\frac{\partial T}{\partial x_k} = \frac{\partial L_k}{\bar{c}_k \partial x_k} + \frac{\partial L_{k+1}}{\bar{c}_{k+1}\partial x_k} - \frac{L_k}{\bar{c}_k^2}\frac{\partial \bar{c}_k}{\partial x_k} - \frac{L_{k+1}}{\bar{c}_{k+1}^2}\frac{\partial \bar{c}_{k+1}}{\partial x_k}\,, \tag{1.11}$$

where

$$\frac{\partial L_k}{\partial x_k} = \frac{x_k - x_{k-1}}{L_k}\,,$$

$$\frac{\partial L_{k+1}}{\partial x_k} = \frac{x_k - x_{k+1}}{L_{k+1}}\,,$$

and

$$\frac{\partial \bar{c}_k}{\partial x_k} = \frac{\partial \bar{c}_{k+1}}{\partial x_k} = \frac{1}{2}\frac{\partial c_k}{\partial x_k}\,.$$

This gives a system of equations of the form:

$$\alpha_k x_{k-1} + \beta_k x_k + \gamma_k x_{k+1} = r_k\,,$$

with similar equations for y_k and z_k. Since it is tri-diagonal it is efficient to solve. (1.11) is not exactly linear, since a relocation of the nodes may change the average velocity $\bar{c}_k$, but it generally iterates quickly to a minimum. The coordinate changes must be orthogonalized to the local ray direction to avoid the problem of the travel time being optimized by collocating many nodes in regions of high velocity (making lower velocities invisible to the interpolation).

[2] `http://geodynamics.org/cig/software/packages/seismo/`.

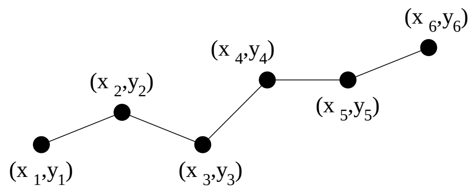
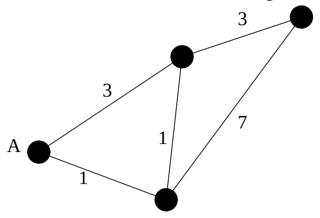


Figure 1.2: The shortest path algorithm gives rays in which individual segments are oriented in a finite set of discrete angles. Bending involves changing the segment coordinates (x_i, y_i) to minimize the travel time T. The coordinates of the endpoints are held fixed.

The precision of travel times and amplitudes can be tested by applying the principle of reciprocity which implies that the travel time from $A \rightarrow B$ should be the same as the time from $B \rightarrow A$. The reciprocity test is not conclusive – for example in the case of the shortest path algorithm the travel time reciprocity is satisfied automatically. One should therefore always test a ray tracing algorithm against simple models for which analytical solutions are available, generally layered models. If reciprocity is not satisfied, something is clearly wrong. For bent ray travel times or for amplitudes in 3D media reciprocity is usually a very good diagnostic. In my own experience, relative precision in travel times is usually better than 10^{-4}, i.e. an error of 0.1 s on a typical teleseismic travel time of 1000 s, or an error of one ms in a crustal reflection that arrives after 10 s. This is sufficient precision in view of the measurement accuracy.

Exercises

Exercise 1.1 The following is a very simple graph of only 4 nodes with specified travel times for 5 connections:



Use the shortest path algorithm to find the travel times and ray paths from node A to the three other nodes.

Chapter 2

Model parametrization

We must represent the model by a finite set of numbers in order to perform calculations. One could, of course, simply *discretize* a model by sampling it at a sufficiently dense set of pixels (sometimes called 'voxels' in 3D). This has the advantage that one does not restrict the smoothness of the model, but the price to be paid is a significant loss of computational efficiency, and this is something we can ill afford. The proper approach is to *parametrize* the model - taking care, however, that the imposed smoothness does not rule out viable classes of models. In addition, the model parametrization should allow for the data to be fit to the error level attributed to them. Note that these two conditions are not identical! In practice, one does well to overparametrize and allow for more parameters than can be resolved. This reduces the risk that the limitations of the parameter space influence the inversion appreciably. Overparametrization poses some problems to the inverse problem, but these can be overcome. We shall deal with that in Chapter 3. If one is forced to underparametrize, effects of bias can be suppressed by using an 'anti-leakage' operator such as proposed by Trampert and Snieder [26].

The formal expression of parametrization is through a set of basis functions $h_j(\boldsymbol{r})$, $j = 1, ..., N$.

Once a choice of basis functions is made, the model is defined by a finite set of numbers, the M model parameters m_j:

$$m(\boldsymbol{r}) = \sum_{j=1}^{M} m_j h_j(\boldsymbol{r})\,.$$

We assume the model $m(\boldsymbol{r})$ represents perturbations in the slowness (1/velocity). The change δT in the travel time of a body wave is then simply given by a line integral along the raypath:

$$\delta T = \int_{\text{raypath}} m(\boldsymbol{r}) ds\,,$$

which allows us to formulate the inverse problem for N observed delay times, ranked in a vector $\boldsymbol{d}$, in matrix form:

$$\begin{aligned} d_i &= \int_{\text{path i}} m(\boldsymbol{r}) d^3\boldsymbol{r} \\ &= \sum_{j=1}^{M} m_j \int_{\text{path i}} h_j(\boldsymbol{r}) ds \\ &= \sum_{j=1}^{M} A_{ij} m_j\,, \end{aligned}$$

where the integral is over the total volume V of Earth or Sun. In matrix notation:

$$\boldsymbol{A}\boldsymbol{m} = \boldsymbol{d}\,, \tag{2.1}$$

with the matrix elements given by

$$A_{ij} = \int_{\text{path i}} h_j(\boldsymbol{r}) ds \,. \tag{2.2}$$

2.1 Global parametrization

When the basis functions h_j are nonzero over all or most of space, the parametrization is called 'global'. A frequently used global parametrization is in terms of spherical harmonics:

$$h_{k\ell m}(\boldsymbol{r}) = f_k(r) \begin{cases} \sqrt{2} X_\ell^{|m|}(\theta) \cos m\phi & -\ell \leq m < 0 \\ X_\ell^0 & m = 0 \\ \sqrt{2} X_\ell^m \sin m\phi & 0 < m \leq \ell \end{cases} , \tag{2.3}$$

where the X_ℓ^m are the (real) colatitudinal harmonics:

$$X_\ell^m = (-1)^m \left[\frac{(2\ell+1)(l-m)!}{4\pi(l+m)!} \right]^{\frac{1}{2}} P_\ell^m(\cos\theta) \,,$$

with the orthogonality property:

$$\int_0^\pi X_\ell^m X_{\ell'}^m \sin\theta \, d\theta = \frac{1}{2\pi} \delta_{\ell\ell'} \,.$$

The radial basis functions f_k are often chosen to be orthogonal, although there is no compelling reason to do so, except that it allows for an easy decomposition (see Exercise 2.1). Masters et al. [15] use natural cubic splines for the radial parametrization.

The spherical harmonic parametrization was first used by Dziewonski [5] and others in the pioneering days of seismic tomography. It has the advantage that it allows for an easy low-pass filtering of the data and comparison with similarly filtered maps of the geoid, the gravity field, or the heat flux, all of which are available as spherical harmonic expansions. For each $\ell = 0, 1, ...$, there are $2\ell + 1$ zonal harmonics with m ranging from $-\ell$ to $+\ell$. As is immediately evident from the term $e^{im\phi}$, the smallest wavelength resolvable is therefore $2\pi/\ell_{max}$ radians, or $40,030/\ell_{max}$ km at the Earth's surface. Because every m effectively provides two basis functions, with a dependence $\sin m\phi$ and $\cos m\phi$ respectively, the smallest structure that can be resolved is equal to *half* the wavelength: about $20,000/\ell_{max}$ km for the Earth and $2.2 \times 10^6/\ell_{max}$ km for the Sun, which has a radius of about 700,000 km.

The disadvantage of spherical harmonics is that many basis functions are needed to resolve features of geodynamic interest in the Earth. To obtain a horizontal resolution of 100 km, $\ell_{max} = 200$ and the total number of spherical harmonics is about $2\ell_{max}^2$ or 80,000. This has to be multiplied by the number of depth basis functions $f_k(r)$, so one easily ends up with more than 10^6 basis functions that are global: to compute the model value at one particular location all spherical harmonics must be evaluated at that location. As the resolution of seismic tomography increased, attention shifted to local parametrizations instead. The choice need not be absolute: Kuo et al. [13] use a hybrid approach, in which they invert first for a low-order spherical harmonic parametrization, then use a local parametrization for the remaining data residuals.

Exercise 2.1 If we expand a model $m(\boldsymbol{r})$ into an orthogonal basis $h_k(\boldsymbol{r})$:

$$m(\boldsymbol{r}) = \sum_k a_k h_k(\boldsymbol{r}) \,,$$

show that we can find the coefficients a_k from:

$$a_k = \int_V m(\boldsymbol{r}) h_k(\boldsymbol{r}) d^3\boldsymbol{r} \,.$$

2.2 Local parametrization

The simplest example of a local parametrization is to divide the earth up into cells, e.g.:

$$h_j(\boldsymbol{r}) = 1 \quad \text{if } \boldsymbol{r} \text{ in cell } i \tag{2.4}$$
$$= 0 \quad \text{elsewhere}\,, \tag{2.5}$$

Homogeneous cell parametrizations were applied in the very first local studies, but quickly found their way into more global inversions. If the cells do not overlap, the basis functions scaled in this way are orthogonal:

$$\int_V h_i(\boldsymbol{r}) h_j(\boldsymbol{r}) d^3\boldsymbol{r} = \delta_{ij} \Delta V_j\,,$$

where ΔV_j is the volume of cell j.

Often, the cells are equidistant in the latitude- and longitude directions, at least over wide latitude bands. Cells can then be uniquely ordered in order of increasing coordinate and it is easy to find the cell that contains a specific location $\boldsymbol{r}$. From (2.2) we see that: the matrix element A_{ij} simply the length of ray i within cell j.

Chapter 3

Linear inversion

In Chapter 2 we saw how the parametrization of a continuous model allows us to formulate a discrete linear relationship between data $\boldsymbol{d}$ and model $\boldsymbol{m}$:

$$\boldsymbol{Am} = \boldsymbol{d} \quad \text{(2.1) again.}$$

Assuming we have M_1 model parameters and M_2 corrections, this is a system of N equations (data) and $M = M_1 + M_2$ unknowns. For more than one reason the solution of the system is not straightforward:

- Even if we do not include multiple measurements along the same path, many of the N rows will be dependent. Since the data always contain errors, this implies we cannot solve the system exactly, but have to minimize the misfit between $\boldsymbol{Am}$ and $\boldsymbol{d}$. For this misfit we can define different norms, and we face a choice of options.

- Despite the fact that we have (usually) many more data than unknowns (i.e. $N \gg M$), the system is almost certainly ill-posed in the sense that small errors in $\boldsymbol{d}$ can lead to large errors in $\boldsymbol{m}$; a parameter m_j may be completely undetermined ($A_{ij} = 0$ for all i) if it represents a node that is far away from any raypath. We cannot escape making a subjective choice among an infinite set of equally satisfactory solutions by imposing a *regularization* strategy.

- For large M, the numerical computation of the solution has to be done with an iterative matrix solver which is often halted when a satisfactory fit is obtained. Such efficient shortcuts interfere with the regularization strategy.

We shall deal with each of these aspects in succession. The appendix introduces some concepts of probability theory that are needed in this chapter.

3.1 Maximum likelihood estimation and least squares

In experimental sciences, the most commonly used misfit criterion is the criterion of least squares, in which we minimize χ^2 ('chi square') as a function of the model:

$$\chi^2(\boldsymbol{m}) = \sum_{i=1}^{N} \left(\frac{\sum_{j=1}^{M} |A_{ij}m_j - d_i|^2}{\sigma_i^2} \right) = \text{ min,} \tag{3.1}$$

where σ_i is the standard deviation in datum i. χ^2 is a direct measure of the data misfit, in which we weigh the misfits inversely with their standard errors σ_i.

For uncorrelated and normally distributed errors, the principle of maximum likelihood leads naturally to the least squares definition of misfit. If there are no sources of bias, the expected value $E(d_i)$ of d_i (the average of infinitely many observations of the same observable) is equal to the 'correct' or error-free value. In practice, we have only one observation for each datum, but we usually have an educated guess of the magnitude of the errors. We almost always use a normal distribution for errors, and assume errors to be uncorrelated, such that the probability density is given by a Gaussian or 'normal' distribution of the form:

$$P(d_i) = \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left(-\frac{|d_i - E(d_i)|^2}{2\sigma_i^2}\right) . \tag{3.2}$$

The joint probability density for the observation of a N-tuple of data with independent errors $\boldsymbol{d} = (d_1, d_2, ..., d_N)$ is found by multiplying the individual probability densities for each datum:

$$P(\boldsymbol{d}) = \prod_{i=1}^{N} \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left(-\frac{|d_i - E(d_i)|^2}{2\sigma_i^2}\right) . \tag{3.3}$$

If we replace the expected values in (3.3) with the predicted values from the model parameters, we obtain again a probability, but now one that is conditional on the model parameters taking the values m_j:

$$P(\boldsymbol{d}|\boldsymbol{m}) = \prod_{i=1}^{N} \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left(-\frac{|d_i - \sum_j A_{ij}m_j|^2}{2\sigma_i^2}\right) . \tag{3.4}$$

We usually assume that there are no extra errors introduced by the modelling (e.g. we ignore the approximation errors introduced by linearizations, neglect of anisotropy, or the shortcomings of ray theory etc.). In fact, if such modelling errors are also uncorrelated, unbiased and normally distributed, we can take them into account by including them in σ_i[1] – but this is a big 'if'.

Clearly, one would like to have a model that is associated with a high probability for its predicted data vector. This leads to the definition of the likelihood function $\mathcal{L}$ for the model $\boldsymbol{m}$ given the observation of the data $\boldsymbol{d}$:

$$\mathcal{L}(\boldsymbol{m}|\boldsymbol{d}) = P(\boldsymbol{d}|\boldsymbol{m}) \propto \exp\left(-\frac{1}{2}\chi^2(\boldsymbol{m})\right) .$$

Thus, maximizing the likelihood for a model involves minimizing χ^2. Since this involves minimizing the sum of squares of data misfit, the method is more generally known as the *method of least squares*. The strong point of the method of least squares is that it leads to very efficient methods of solving (2.1). Its major weakness is the reliance on a normal distribution of the errors, which may not always be the case. Because of the quadratic dependence on the misfit, outliers – misfits of several standard deviations – have an influence on the solution that may be out of proportion, which means that errors may dominate in the solution. For a truly normal distribution, large errors have such a low probability of occurrence that we would not worry about this. In practice however, many data do suffer from outliers. For picked arrival times Jeffreys [11] already observed that the data have a tail-like distribution that deviates from the Gaussian for large deviations from the mean t_m, mainly because a later arrival is misidentified as P or S:

$$P(t) = \frac{1-\epsilon}{\sigma\sqrt{2\pi}} \mathrm{e}^{-(t-t_m)^2/2\sigma^2} + \epsilon g(t),$$

where the probability density $g(t)$ varies slowly and where $\epsilon \ll 1$. A simple method to bring the data distribution close to normal is to reject outliers with a delay that exceeds the largest delay time to be expected from reasonable effects of lateral heterogeneity. This decision can be made after a first trial inversion: for example, one may reject all data that leave a residual in excess of 3σ after a first inversion attempt.

If we divide all data – and the corresponding row of $\boldsymbol{A}$ – by their standard deviations, we end up with a data vector that is univariant, i.e. all standard deviations are equal to 1. Thus, without loss of generality, we may assume that the data are univariant, in which case we see from (3.1) that χ^2 is simply the squared length of the residual vector $|\boldsymbol{r}| = |\boldsymbol{d} - \boldsymbol{A}\boldsymbol{m}|$. From Fig. 3.1 we see that $\boldsymbol{r}$ is then perpendicular to the subspace spanned by all vectors $\boldsymbol{A}\boldsymbol{y}$ (the 'range' $R(\boldsymbol{A})$ of $\boldsymbol{A}$). For if it was not, we could add a $\delta\boldsymbol{m}$ to $\boldsymbol{m}$ such that $\boldsymbol{A}\delta\boldsymbol{m}$ reduces the length of $\boldsymbol{r}$. Thus, for all $\boldsymbol{y}$ the dot product between $\boldsymbol{r}$ and $\boldsymbol{A}\boldsymbol{y}$ must be zero:

$$\boldsymbol{r} \cdot \boldsymbol{A}\boldsymbol{y} = \boldsymbol{A}^T\boldsymbol{r} \cdot \boldsymbol{y} = \boldsymbol{A}^T(\boldsymbol{d} - \boldsymbol{A}\boldsymbol{m}) \cdot \boldsymbol{y} = 0,$$

[1]See Tarantola [24] for a much more comprehensive discussion of this issue.

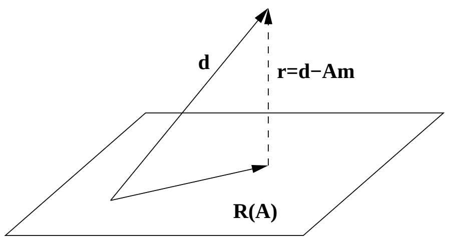


Figure 3.1: If the data vector $\boldsymbol{d}$ does not lie in the range of $\boldsymbol{A}$, the best we can do is minimize the length of the residual vector $\boldsymbol{r}$. This implies that $\boldsymbol{r}$ must be perpendicular to any possible vector $\boldsymbol{Ay}$.

where $\boldsymbol{A}^T$ is the transpose of $\boldsymbol{A}$ (i.e. $A^T_{ij} = A_{ji}$). Since this dot product is 0 for *all* $\boldsymbol{y}$, clearly $\boldsymbol{A}^T(\boldsymbol{d} - \boldsymbol{Am}) = 0$, or:

$$\boldsymbol{A}^T\boldsymbol{Am} = \boldsymbol{A}^T\boldsymbol{d}, \tag{3.5}$$

which is known as the set of 'normal equations' to solve the least squares problem.

χ^2 is an essential statistical measure of the goodness of fit. In the hypothetical case that we satisfy every datum with a misfit of one standard deviation we find $\chi^2 = N$; clearly values much higher than N are unwanted because the misfit is higher than could be expected from the knowledge of data errors, and values much lower than N indicate that the model is trying to fit the data errors rather than the general trend in the data. For example, if two very close rays have travel time anomalies differing by only 0.5 s and the standard deviation is estimated to be 0.7 s, we should accept that a smooth model predicts the same anomaly for each, rather than introducing a steep velocity gradient in the 3D model to try to satisfy the difference. Because we want $\chi^2 \approx N$, it is often convenient to work with the reduced χ^2 or χ^2_{red}, which is defined as χ^2/N, so that the optimum solution is found for $\chi^2_{\text{red}} \approx 1$.

But how close should χ^2 be to N? Statistical theory shows that χ^2 itself has a variance of $2N$, or a standard deviation of $\sqrt{2N}$. Thus, for 1,000,000 data the true model would with 67% confidence be found in the interval $\chi^2 = 1,000,000 \pm 1,414$. Such theoretical bounds are almost certainly too narrow because our estimates of the standard deviations σ_i are themselves uncertain. For example, if the true σ_i are equal to 0.9 but we used 1.0 to compute χ^2, our computed χ^2 itself is in error (i.e. too low) by almost 20%, and a model satisfying this level of misfit is probably not good enough. It is therefore important to obtain accurate estimates of the standard errors. Provided one is confident that the estimated standard errors are unbiased, one should still aim for a model that brings χ^2 very close to N, say to within 20 or 30%.

An additional help in deciding how close one wishes to be to a model that fits at a level given by $\chi^2 = N$ is to plot the trade-off between the model norm and χ^2 (sometimes called the L-curve), schematically shown in Fig. 3.2. If the trade-off curve shows that one could significantly reduce the norm of the model while paying only a small price in terms of an increase in χ^2 (point A in Fig. 3.2), this is an indication that the standard errors in the data have been underestimated. For common data errors do not correlate between nearby stations, but the true delays should correlate – even if the Earth's properties vary erratically (because of the overlap in finite-frequency sensitivity). The badly correlating data can only be fit by significantly increasing the norm and complexity of the model, which is what we see happening on the horizontal part of the trade-off curve. Conversely, if we notice that a significant decrease in χ^2 can be obtained at the cost of only a minor increase in model norm (point B), this indicates an overestimate of data errors and tells us we may wish to accept a model with $\chi^2 < N$. If the deviations required are unexpectedly large, this is an indication that the error estimation for the data may need to be revisited.

Depending on where on the L-curve we find that $\chi^2 = N$, we find that we do or do not have a strong constraint on the norm of the model. If the optimal data fit is obtained close to point B where the L-curve is steep, even large changes in χ^2 have little effect on the model norm. On the other hand, near point A even large changes in the model give only a small improvement of the data fit. Both A and B represent unwanted situations, since at A we are trying to fit data errors, which leads to erratic features in the model, whereas at B we are damping too strongly. In a well designed tomography experiment, $\chi^2 \approx N$ near the bend in the L-curve.

We used the term 'model norm' here in a very general sense – one may wish to inspect the Euclidean $|\boldsymbol{m}|^2$ as well as more complicated norms that we shall encounter in section 3.4.

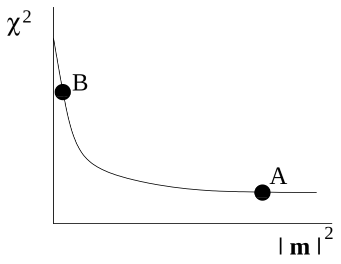


Figure 3.2: The L- or trade-off curve between χ^2 and model norm $|\boldsymbol{m}|^2$.

Early tomographic studies often ignored a formal statistical appraisal of the goodness of fit, and merely quoted how much better a 3D tomographic model satisfies the data when compared to a 1D (layered or spherically symmetric) background or 'starting' model, using a quantity named 'variance reduction', essentially the reduction in the Euclidean norm of the misfit vector. This reduction is as much a function of the fit of the 1D starting model as of the data fit itself – i.e. the same 3D model can have different variance reductions depending on the starting model – and is therefore useless as a statistical measure of quality for the tomographic model.

Exercise 3.1 Derive the normal equations by differentiating the expression for χ^2 with respect to m_k for $k = 1, ..., M$. Assume univariant data ($\sigma_i = 1$).

Exercise 3.2 Why can we not conclude from (3.5) that $\boldsymbol{Am} \equiv \boldsymbol{d}$?

3.2 Singular value decomposition

Though the least squares formalism handles the incompatibility problem of data in an overdetermined system, we usually find that $\boldsymbol{A}^T\boldsymbol{A}$ has a determinant equal to zero, i.e. eigenvalues equal to zero, and its inverse does not exist. Even though in tomographic applications $\boldsymbol{A}^T\boldsymbol{A}$ is often too large to be diagonalized, we shall analyse the inverse problem using singular values ('eigenvalues' of a non-square matrix), since this formalism gives considerable insight.

Let $\boldsymbol{v}_i$ be an eigenvector of $\boldsymbol{A}^T\boldsymbol{A}$ with eigenvalue λ_i^2, so that $\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{v} = \lambda_i^2\boldsymbol{v}$. We may use squared eigenvalues because $\boldsymbol{A}^T\boldsymbol{A}$ is symmetric and has only non-negative, real eigenvalues. Its eigenvectors are orthogonal. The choice of λ^2 instead of λ as eigenvalue is for convenience: the notation λ_i^2 avoids the occurrence of $\sqrt{\lambda_i}$ later in the development. We can arrange all M eigenvectors as columns in an $M \times M$ matrix $\boldsymbol{V}$ and write:

$$\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{V} = \boldsymbol{V}\boldsymbol{\Lambda}^2 \tag{3.6}$$

The eigenvectors are normalized such that $\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{V}\boldsymbol{V}^T = \boldsymbol{I}$.

With (3.6) we can study the underdetermined nature of the problem $\boldsymbol{Am} = \boldsymbol{d}$, of which the least squares solution is given by the system $\boldsymbol{A}^T\boldsymbol{Am} = \boldsymbol{A}^T\boldsymbol{d}$. The eigenvectors $\boldsymbol{v}_i$ span the M-dimensional model space so $\boldsymbol{m}$ can be written as a linear combination of eigenvectors: $\boldsymbol{m} = \boldsymbol{V}\boldsymbol{y}$. Since $\boldsymbol{V}$ is orthonormal, $|\boldsymbol{m}| = |\boldsymbol{y}|$ and we can work with $\boldsymbol{y}$ instead of $\boldsymbol{m}$ if we wish to restrict the norm of the model. Using this:

$$\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{V}\boldsymbol{y} = \boldsymbol{V}\boldsymbol{\Lambda}^2\boldsymbol{y} = \boldsymbol{A}^T\boldsymbol{d}\,,$$

or, multiplying both on the left with $\boldsymbol{V}^T$ and using the orthogonality of $\boldsymbol{V}$:

$$\boldsymbol{\Lambda}^2\boldsymbol{y} = \boldsymbol{V}^T\boldsymbol{A}^T\boldsymbol{d}\,.$$

Since $\boldsymbol{\Lambda}$ is diagonal, this gives y_i (and with that $\boldsymbol{m} = \boldsymbol{V}\boldsymbol{y}$) simply by dividing the i-th component of the vector on the right by λ_i^2. But clearly, any y_i which is multiplied by a zero eigenvalue can take any value without affecting the data fit! We find the *minimum norm solution*, the solution with the smallest $|\boldsymbol{y}|^2$, by setting such components of $\boldsymbol{y}$ to 0. If

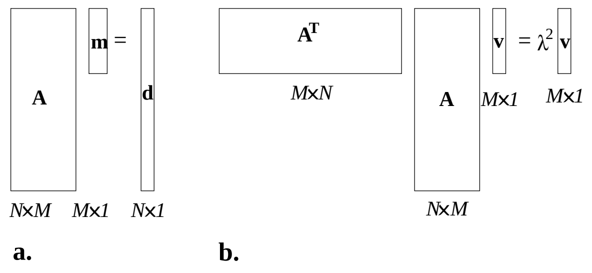


Figure 3.3: (a) The original matrix system $\boldsymbol{Am} = \boldsymbol{d}$. (b) The eigenvalue problem for the least-squares matrix $\boldsymbol{A}^T\boldsymbol{A}$.

we rank the eigenvalues $\lambda_1^2 \geq \lambda_2^2 \geq ...\lambda_K^2 > 0, 0, ..., 0$, then the last $M - K$ columns of $\boldsymbol{V}$ belong to the nullspace of $\boldsymbol{A}^T\boldsymbol{A}$. We truncate the matrices $\boldsymbol{V}$ and $\boldsymbol{\Lambda}$ to an $M \times K$ matrix $\boldsymbol{V}_K$ and a $K \times K$ diagonal matrix $\boldsymbol{\Lambda}$ to obtain the minimum norm estimate:

$$\hat{\boldsymbol{m}}_{\text{min norm}} = \boldsymbol{V}_K\boldsymbol{\Lambda}_K^{-2}\boldsymbol{V}_K^T\boldsymbol{A}^T\boldsymbol{d} \tag{3.7}$$

Note that the inverse of $\boldsymbol{\Lambda}_K$ exists because we have removed the zero eigenvalues. The orthogonality of the eigenvectors still guarantees $\boldsymbol{V}_K^T\boldsymbol{V}_K = \boldsymbol{I}_K$, but now $\boldsymbol{V}_K\boldsymbol{V}_K^T \neq \boldsymbol{I}_M$.

To see how errors in the data propagate into the model, we use the fact that (3.7) represents a linear transformation of data with a covariance matrix $\boldsymbol{C}_d$. The posteriori covariance of transformed data $\boldsymbol{Td}$ is equal to $\boldsymbol{TC}_d\boldsymbol{T}^T$ (see equation 3.35 in the appendix). In our case we scaled the data such that $\boldsymbol{C}_d = \boldsymbol{I}$ so that the posteriori model covariance is:

$$\begin{aligned}\boldsymbol{C}_{\hat{m}} &= \boldsymbol{V}_K\boldsymbol{\Lambda}_K^{-2}\boldsymbol{V}_K^T\boldsymbol{A}^T\boldsymbol{IAV}_K\boldsymbol{\Lambda}_K^{-2}\boldsymbol{V}_K^T \\ &= \boldsymbol{V}_K\boldsymbol{\Lambda}_K^{-2}\boldsymbol{\Lambda}_K^{2}\boldsymbol{\Lambda}_K^{-2}\boldsymbol{V}_K^T \\ &= \boldsymbol{V}_K\boldsymbol{\Lambda}_K^{-2}\boldsymbol{V}_K^T.\end{aligned} \tag{3.8}$$

Thus the posteriori variance of the estimate for parameter m_i is given by[2]:

$$\sigma_{m_i}^2 = \sum_{j=1}^{K} \frac{V_{ij}^2}{\lambda_j^2}. \tag{3.9}$$

This equation makes it clear that removing zero singular values is not sufficient, since the errors blow up as λ_j^{-2}, rendering the incorporation of small λ_j very dangerous. Dealing with small eigenvalues is known as *regularization* of the problem. Before we discuss this in more detail, we need to show the connection between the development given here and the theory of singular value decomposition which is more commonly found in the literature.

One way of looking at the system $\boldsymbol{Am} = \boldsymbol{d}$ is to see the components m_i as weights in a summation of the columns of $\boldsymbol{A}$ to fit the data vector $\boldsymbol{d}$. The columns make up the range of $\boldsymbol{A}$ in the data space (Fig. 3.4). Similarly, the rows of $\boldsymbol{A}$ – the columns of $\boldsymbol{A}^T$ – make up the range of the backprojection $\boldsymbol{A}^T$ in the model space. The rest of the model space is the nullspace: if $\boldsymbol{m}$ is in the nullspace, $\boldsymbol{Am} = 0$. Components in the nullspace do not contribute to the data fit, but add to the norm of $\boldsymbol{m}$. We find the minimum norm solution by avoiding any components in the nullspace, in other words by selecting a model in the range of $\boldsymbol{A}^T$:

$$\hat{\boldsymbol{m}} = \boldsymbol{A}^T\boldsymbol{y}$$

and find $\boldsymbol{y}$ by solving for:

$$\boldsymbol{AA}^T\boldsymbol{y} = \boldsymbol{d}$$

The determinant of $\boldsymbol{AA}^T$ is likely to be zero, so just as in the case of least squares we shall wish to eliminate zero eigenvalues. Let the eigenvectors of $\boldsymbol{AA}^T$ be $\boldsymbol{u}_i$ with eigenvalues $\tilde{\lambda}_i^2$:

$$\boldsymbol{AA}^T\boldsymbol{U} = \boldsymbol{U}\tilde{\boldsymbol{\Lambda}}^2 \tag{3.10}$$

[2]To distinguish data uncertainty from model uncertainty we denote the model standard deviation as σ_{m_i} and the data standard deviation as σ_i.

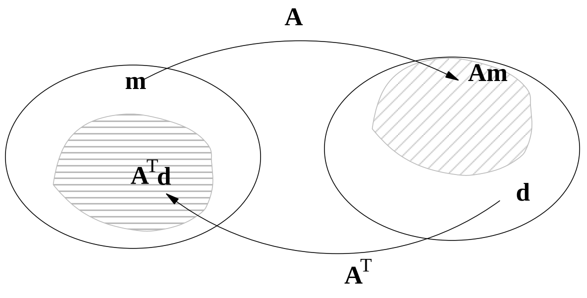


Figure 3.4: Mappings between the model space (left) and the data space (right). The range of $\boldsymbol{A}$ is indicated by the grey area within the data space. The range of the backprojection $\boldsymbol{A}^T$ is indicated by the grey area in the model space.

Since $\boldsymbol{A}\boldsymbol{A}^T$ is symmetric, the eigenvectors are orthogonal and we can scale them to be orthonormal, such that $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{I}$. Multiplying (3.10) on the left by $\boldsymbol{A}^T$ and grouping $\boldsymbol{A}^T\boldsymbol{U}$ we see that $\boldsymbol{A}^T\boldsymbol{U}$ is an eigenvector of $\boldsymbol{A}^T\boldsymbol{A}$:

$$\boldsymbol{A}^T\boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{U}) = (\boldsymbol{A}^T\boldsymbol{U})\tilde{\boldsymbol{\Lambda}}^2$$

and comparison with (3.6) shows that $\boldsymbol{A}^T\boldsymbol{u}_i$ must be a constant$\times\boldsymbol{v}_i$, and $\tilde{\lambda}_i = \lambda_i$. We choose the constant to be λ_i, so that

$$\boldsymbol{A}^T\boldsymbol{U} = \boldsymbol{V}\boldsymbol{\Lambda}\,. \tag{3.11}$$

Multiplying this on the left by $\boldsymbol{A}$ we obtain:

$$\boldsymbol{A}\boldsymbol{A}^T\boldsymbol{U} = \boldsymbol{U}\boldsymbol{\Lambda}^2 = \boldsymbol{A}\boldsymbol{V}\boldsymbol{\Lambda}\,,$$

or, dividing by λ_i for $\lambda_i \neq 0$ and setting $\boldsymbol{A}\boldsymbol{v}_i = \lambda_i\boldsymbol{u}_i$ if $\boldsymbol{v}_i$ is in the nullspace::

$$\boldsymbol{A}\boldsymbol{V} = \boldsymbol{U}\boldsymbol{\Lambda}\,. \tag{3.12}$$

In the same way, by multiplying (3.12) on the right by $\boldsymbol{V}^T$ we find:

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^T \tag{3.13}$$

which is the *singular value decomposition* of $\boldsymbol{A}$. Note that in this development we have carefully avoided using the inverse of $\boldsymbol{\Lambda}$, so there is no need to truncate it to exclude zero singular values. However, because the tail of the diagonal matrix $\boldsymbol{\Lambda}$ contains only zeroes, (3.13) is equivalent to the truncated version (Fig. 3.5):

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^T = \boldsymbol{U}_K\boldsymbol{\Lambda}_K\boldsymbol{V}_K^T \tag{3.14}$$

We use this to define the generalized inverse $\boldsymbol{A}^-$:

$$\boldsymbol{A}^- = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^T = \boldsymbol{V}_K\boldsymbol{\Lambda}_K^{-1}\boldsymbol{U}_K^T \tag{3.15}$$

which exists because none of the elements of $\boldsymbol{\Lambda}$ is zero.

Exercise 3.3 Show that $\boldsymbol{A}^-\boldsymbol{A} = \boldsymbol{V}_K\boldsymbol{V}_K^T$ and that $\boldsymbol{A}\boldsymbol{A}^- = \boldsymbol{U}_K\boldsymbol{U}_K^T$. Are these products unit matrices?

Exercise 3.4 Show that the choice (3.11) indeed implies that $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}$. Hint: use (3.12).

Exercise 3.5 Show that the eigenvalues of $\boldsymbol{A}^T\boldsymbol{A}$ cannot be negative.

3.3 Tikhonov regularization

The truncation to include only nonzero singular values is an example of regularization of the inverse problem. Removing zero λ_i is not sufficient however, since small singular values may give rise to large modelling errors, as shown by

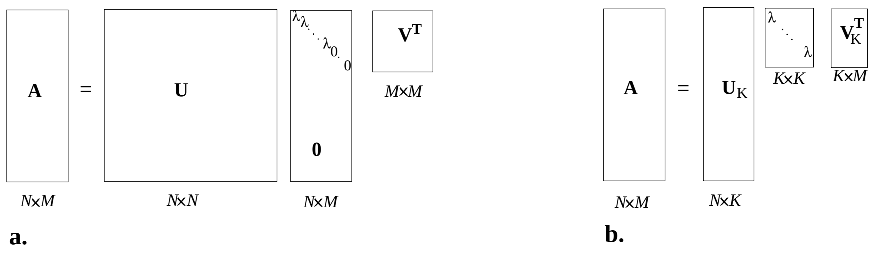


Figure 3.5: (a) The full eigenvalue problem for $\boldsymbol{A}\boldsymbol{A}^T$ leads to a matrix with small or zero eigenvalues on the diagonal. (b) removing zero eigenvalues has no effect on $\boldsymbol{A}$.

(3.9). This equation tells us that small errors in the data vector may cause very large excursions in model space in the direction of $\boldsymbol{v}_k$ if $\lambda_k \ll 1$. It seems thus wise to truncate $\boldsymbol{V}$ in (3.13) even further, and exclude eigenvectors belonging to small singular values. The price we pay is a small increase in χ^2, but we are rewarded by a significant reduction in the modelling error. We could apply a sharp cut-off by choosing K at some nonzero threshold level for the singular values. Less critical to the choice of threshold is a tapered cut-off. We show that the latter approach is equivalent to adding M equations of the form $\epsilon_n m_i = 0$, with ϵ_n small, to the tomographic system. Such equations act as artificial 'data' that bias the model parameters towards zero:

$$\begin{pmatrix} \boldsymbol{A} \\ \epsilon_n \boldsymbol{I} \end{pmatrix} \boldsymbol{m} = \begin{pmatrix} \boldsymbol{d} \\ \boldsymbol{0} \end{pmatrix}. \tag{3.16}$$

If the $j-$th column of $\boldsymbol{A}$ – associated with parameter m_j – has large elements, the addition of one additional constraint $\epsilon_n m_j = 0$ will have very little influence. But the more m_j is underdetermined by the undamped system, the more the damping will push m_j towards zero. The least squares solution of (3.16) is:

$$(\boldsymbol{A}^T\boldsymbol{A} + \epsilon_n^2 \boldsymbol{I})\boldsymbol{m} = \boldsymbol{A}^T\boldsymbol{d}. \tag{3.17}$$

The advantage of the formulation (3.16) is that it can easily be solved iteratively, without a need for singular value decomposition. But the solution of (3.16) does have a simple representation in terms of singular values, and it is instructive to analyse it with SVD. If $\boldsymbol{v}_k$ is an eigenvector of $\boldsymbol{A}^T\boldsymbol{A}$ with eigenvalue λ_k^2, then the damped matrix gives:

$$(\boldsymbol{A}^T\boldsymbol{A} + \epsilon_n^2 \boldsymbol{I})\boldsymbol{v}_k = (\lambda_k^2 + \epsilon_n^2)\boldsymbol{v}_k\,, \tag{3.18}$$

and we see that the damped system has the same eigenvectors but with raised eigenvalues $\lambda_k^2 + \epsilon_n^2 > 0$. The minimum norm solution (3.7) is therefore replaced by:

$$\hat{\boldsymbol{m}}_{\text{damped}} = \boldsymbol{V}_K(\boldsymbol{\Lambda}_K^2 + \epsilon_n^2 \boldsymbol{I})^{-1}\boldsymbol{V}_K^T\boldsymbol{A}^T\boldsymbol{d} \tag{3.19}$$

with the posteriori model variance given by:

$$\sigma_{m_i}^2 = \sum_{j=1}^{K} \frac{V_{ij}^2}{\lambda_j^2 + \epsilon_n^2}. \tag{3.20}$$

Since there are no zero eigenvalues, we may set $K = N$, but of course this maximizes the variance and some truncation may still be needed. For simplicity, we assumed a damping with the same ϵ_n everywhere on the diagonal. The method is often referred to as Tikhonov regularization, after its original discoverer [25]. Because one adds ϵ^2 to the diagonal of $\boldsymbol{A}^T\boldsymbol{A}$ it is also known as 'ridge regression'.

Spakman and Nolet [22] vary the damping factor ϵ_n along the diagonal. When corrections are part of the model, one should vary damping factors such that damping results in corrections that are reasonable in view of the prior uncertainty (for example, one would judge corrections as large as 100 km for hypocentral parameters usually unacceptable and increase ϵ_n for those corrections).

A comparison of (3.20) and with (3.9) shows that damped model errors blow up at most by a factor ϵ_n^{-1}. Thus, damping reduces the variance of the solution. This comes at a price however: by discarding eigenvectors, we reduce our ability to shape the model. The small eigenvalues are usually associated with vectors that are strongly oscillating in space: the positive and negative parts cancel upon integration and the resulting integral (**??**) is small. Damping small eigenvalues is thus expected to lead to smoother models. However, even long-wavelength features of the model may be biased towards zero because of regularization.

The fact that biased estimations produce smaller variances is a well known phenomenon in statistical estimation, and it is easily misunderstood: one can obtain a very small model parameter m_i with a very small posteriori variance σ_i^2, yet learn nothing about the model because the bias is of the order of the true m_i. We shall come back to this in the section on resolution, but first investigate a more powerful regularization method, based on Bayesian statistics.

Exercises

Exercise 3.6 Show that the minimization of $|\boldsymbol{A}\boldsymbol{m}-\boldsymbol{d}|^2+\epsilon^2|\boldsymbol{m}|^2$ leads to (3.17).

Exercise 3.7 In the L-curve for (3.19), indicate where $\epsilon = 0$ and where $\epsilon \to \infty$.

3.4 Bayesian inference

The simple Tikhonov regularization by norm damping we introduced in the previous section , while reducing the danger of excessive error propagation, is usually not satisfactory from a geophysical point of view. At first sight, this may seem surprising: for, when the m_i represent perturbations with respect to a background model, the damping towards 0 is defensible if we prefer the model values given by the background model in the absence of any other information. However, if the information given by the data is unequally distributed, some parts of the model may be damped more than others, introducing an apparent structure in $\boldsymbol{m}$ that may be very misleading. The error estimate (3.20) does not represent the full modelling error because it neglects the bias. In general, we would like the model to have a minimum of unwarranted *structure*, or detail. Jackson [10] and Tarantola [23], significantly extending earlier work by Franklin [7], introduced the Bayesian method into geophysical inversion to deal with this problem, named after the Reverend Thomas Bayes (1702-1761), a British mathematician whose theorem on joint probabilities is a cornerstone of this inference method.

We shall give a brief exposé of Bayesian estimation for the case of N observations in a data vector $\boldsymbol{d}^{\text{obs}}$. Let $P(\boldsymbol{m})$ be the prior probability density for the model $\boldsymbol{m}=(m_1, m_2, ..., m_M)$, e.g. a Gaussian probability of the form:

$$P(\boldsymbol{m}) = \frac{1}{(2\pi)^{M/2}} \frac{1}{|\det \boldsymbol{C}_m|^{1/2}} \exp\left(-\frac{1}{2}\boldsymbol{m}\cdot\boldsymbol{C}_m^{-1}\boldsymbol{m}\right) \tag{3.21}$$

Here, $\boldsymbol{C}_m$ is the *prior* covariance matrix for the model parameters. With 'prior' we mean that we generally have an idea of the allowable variations in the model values, e.g. how much the 3D Earth may differ from a 1D background model without violating more general laws of physics. We may express such knowledge as a prior probability density for the model values. The diagonal elements of $\boldsymbol{C}_m$ are the variances of that prior distribution. The off-diagonal elements reflect the correlation of model parameters – often it helps to think of them as describing the likely 'smoothness' of the model.

In a strict Bayesian philosophy such constraints may be 'subjective'. This, however, is not to say that we may impose constraints following the whim of an arbitrary person. An experienced geophysicist may often develop a very good intuition of the prior uncertainty of model parameters, perhaps because he has done experiments in the laboratory on analogue materials, or because he has experience with tomographic inversions in similar geological provinces. We shall classify such defensible subjective notions to be 'objective' after all.

The random errors in our observations make that the observed data vector $\boldsymbol{d}^{\text{obs}}$ deviates from the true (i.e. error-free) data $\boldsymbol{d}$. For the data we assume the normal distribution (3.2). Assuming the linear relationship $\boldsymbol{A}\boldsymbol{m}=\boldsymbol{d}$ has no errors (or incorporating those errors into σ_i as discussed before), we find the conditional probability density for the observed data, *given a model* $\boldsymbol{m}$:

$$P(\boldsymbol{d}|\boldsymbol{m}) = \frac{1}{(2\pi)^{N/2}} \frac{1}{|\det \boldsymbol{C}_d|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{Am} - \boldsymbol{d}^{\text{obs}}) \cdot \boldsymbol{C}_d^{-1}(\boldsymbol{Am} - \boldsymbol{d}^{\text{obs}})\right), \tag{3.22}$$

where $\boldsymbol{C}_d$ is the matrix with data covariance, usually taken to be diagonal with entries σ_i^2 because we have little knowledge about data correlations.

Though we have an expression for the data probability $P(\boldsymbol{d}|\boldsymbol{m})$, for solution of the inverse problem we are more interested in the probability of the model, *given the observed data* $\boldsymbol{d}^{\text{obs}}$. This is where Bayes' theorem is useful. It starts from the recognition that the joint probability can be split up in a conditional and marginal probability in two ways, assuming the probabilities for model and data are independent:

$$P(\boldsymbol{m}, \boldsymbol{d}^{\text{obs}}) = P(\boldsymbol{m}|\boldsymbol{d}^{\text{obs}})P(\boldsymbol{d}^{\text{obs}}) = P(\boldsymbol{d}^{\text{obs}}|\boldsymbol{m})P(\boldsymbol{m}),$$

from which we find Bayes' theorem:

$$P(\boldsymbol{m}|\boldsymbol{d}^{\text{obs}}) = \frac{P(\boldsymbol{d}^{\text{obs}}|\boldsymbol{m})P(\boldsymbol{m})}{P(\boldsymbol{d}^{\text{obs}})}. \tag{3.23}$$

using (3.21) and (3.22):

$$P(\boldsymbol{m}|\boldsymbol{d}^{\text{obs}}) \propto \exp\left[-\frac{1}{2}(\boldsymbol{Am} - \boldsymbol{d}^{\text{obs}}) \cdot \boldsymbol{C}_d^{-1}(\boldsymbol{Am} - \boldsymbol{d}^{\text{obs}}) - \frac{1}{2}\boldsymbol{m} \cdot \boldsymbol{C}_m^{-1}\boldsymbol{m}\right].$$

Thus, we obtain the maximum likelihood solution by minimizing:

$$(\boldsymbol{Am} - \boldsymbol{d}^{\text{obs}}) \cdot \boldsymbol{C}_d^{-1}(\boldsymbol{Am} - \boldsymbol{d}^{\text{obs}}) + \boldsymbol{m} \cdot \boldsymbol{C}_m^{-1}\boldsymbol{m} = \chi^2(\boldsymbol{m}) + \boldsymbol{m} \cdot \boldsymbol{C}_m^{-1}\boldsymbol{m} = \text{min},$$

or, differentiating with respect to m_i:

$$\boldsymbol{A}^T\boldsymbol{C}_d^{-1}(\boldsymbol{Am} - \boldsymbol{d}^{\text{obs}}) + \boldsymbol{C}_m^{-1}\boldsymbol{m} = 0.$$

One sees that this is – again – a system of normal equations belonging to the 'damped' system:

$$\begin{pmatrix} \boldsymbol{C}_d^{-\frac{1}{2}}\boldsymbol{A} \\ \boldsymbol{C}_m^{-\frac{1}{2}} \end{pmatrix} \boldsymbol{m} = \begin{pmatrix} \boldsymbol{C}_d^{-\frac{1}{2}}\boldsymbol{d} \\ \boldsymbol{0} \end{pmatrix}. \tag{3.24}$$

Of course, if we already scaled the data to be univariant the data covariance matrix is $\boldsymbol{C}_d = \boldsymbol{I}$. This simply shows that we are sooner or later obliged to scale the system with the data uncertainty. The prior smoothness constraint is unlikely to be a 'hard' constraint, and in practice we face again a trade-off between the data fit and the damping of the model, much as in Fig. 3.2. We obtain a manageable flexibility in the trade-off between smoothness of the model and χ^2 by scaling $\boldsymbol{C}_d^{-\frac{1}{2}}$ with a scaling factor ϵ. Varying ϵ allows us to tweak the model damping until $\chi^2 \approx N$. Equation (3.24) is thus usually encountered in the equivalent, simplified form:

$$\begin{pmatrix} \boldsymbol{A} \\ \epsilon\boldsymbol{C}_m^{-\frac{1}{2}} \end{pmatrix} \boldsymbol{m} = \begin{pmatrix} \boldsymbol{d} \\ \boldsymbol{0} \end{pmatrix}. \tag{3.25}$$

How should one specify $\boldsymbol{C}_m$? The model covariance essentially tells us how model parameters are correlated. Usually, such correlations are only high for nearby parameters. Thus, $\boldsymbol{C}_m$ smoothes the model when operating on $\boldsymbol{m}$. Conversely, $\boldsymbol{C}_m^{-1}$ roughens the model, and (3.25) expresses the penalization of those model elements that dominate after the roughening operation. The simplest roughening operator is the Laplacian ∇^2, which is zero when a model parameter is exactly the average of its neighbours. If we parametrize the model with tetrahedra or blocks, so that every node has well-defined nearest neighbours, we can minimize the difference between parameter m_i and the average of its neighbours (Nolet [19]):

$$\frac{1}{2}\sum_i \frac{1}{N_i} \sum_{j \in \mathcal{N}_i} (m_i - m_j)^2 = \text{min},$$

where $\mathcal{N}_i$ is the set of N_i nearest neighbours of mode i. Differentiating with respect to m_k gives M equations:

$$m_k - \frac{1}{N_k} \sum_{j \in \mathcal{N}_k} m_j = 0, \tag{3.26}$$

in which we recognize the k-th row of $\boldsymbol{C}_m^{-\frac{1}{2}}\boldsymbol{m}$ in (3.25).

Though it is not practical to invert the matrix $\boldsymbol{C}_m^{-\frac{1}{2}}$ that is implicit in (3.26) to find an exact expression for $\boldsymbol{C}_m^{\frac{1}{2}}$, many explicit smoothers of $\boldsymbol{m}$ may act as an appropriate 'correlation' matrix $\boldsymbol{C}_m^{\frac{1}{2}}$ for regularization purposes. After inversion for $\boldsymbol{m}'$, the tomographic model is obtained from the smoothing operation $\boldsymbol{m} = \boldsymbol{C}_m^{\frac{1}{2}}\boldsymbol{m}'$. The system (**??**) has the same form as the Tikhonov regularization (3.16). Despite this resemblance, in my own experience the acceleration of convergence is only modest compared to inverting (3.25) directly.

3.5 Appendix: some concepts of probability theory and statistics

I assume the reader is familiar with discrete probabilities, such as the probability that a flipped coin will come up with head or tail. If added up for all possible outcomes, the sum of all probabilities is 1.

This concept of probability cannot directly be applied to variables that can take any value within prescribed bounds. For such variables we use probability *density*. The probability density $P(X_0)$ for a random variable X at X_0 is equal to the probability that X is within the interval $X_0 \leq X \leq X_0 + dX$, divided by dX.

This can be extended to multiple variables. If $P(\boldsymbol{d})$ is the probability density for the data in vector $\boldsymbol{d}$, then the probability that we find the data within a small N-dimensional volume $\Delta\boldsymbol{d}$ in data space is given by $0 \leq P(\boldsymbol{d})\Delta\boldsymbol{d} \leq 1$. We only deal with normalized probability densities, i.e. the integral over all data:

$$\int P(\boldsymbol{d}) d^N\boldsymbol{d} = 1\,. \tag{3.27}$$

Joint probability densities give the probability that two or more random variables take a particular value, e.g. $P(\boldsymbol{m}, \boldsymbol{d})$. If the distributions for the two variables are independent, the *joint* probability density is the product of the individual densities:

$$P(\boldsymbol{m}, \boldsymbol{d}) = P(\boldsymbol{m})P(\boldsymbol{d}). \tag{3.28}$$

Conversely, one finds the *marginal* probability density of one of the variables by integrating out the second variable:

$$P(\boldsymbol{m}) = \int P(\boldsymbol{m}, \boldsymbol{d}) d^N\boldsymbol{d}. \tag{3.29}$$

The *conditional* probability density gives the probability of the first variable under the condition that the second variable has a given value, e.g. $P(\boldsymbol{m}|\boldsymbol{d}^{\text{obs}})$ gives the probability density for model $\boldsymbol{m}$ given an observed set of data in $\boldsymbol{d}^{\text{obs}}$.

The *expectation* or expected value $E(X)$ of X is defined as the average over all values of X weighted by the probability density:

$$\bar{X} \equiv E(X) = \int P(X)X\,dX. \tag{3.30}$$

The expectation is a linear functional:

$$E(aX + bY) = aE(X) + bE(Y)\,, \tag{3.31}$$

and for independent variables it is separable:

$$E(XY) = E(X)E(Y)\,. \tag{3.32}$$

The *variance* is a measure of the spread of X around its expected value:

$$\sigma_X^2 = E[(X - \bar{X})^2]\,, \tag{3.33}$$

where σ_X itself is known as the standard deviation. The covariance between two random variables X and Y is defined as

$$\text{Cov}(X, Y) = E[(X - \bar{X})(Y - \bar{Y})]\,. \tag{3.34}$$

In the case of an N-tuple of variables this defines an $N \times N$ covariance matrix, with the variance on the diagonal. The covariance matrix of a linear combination of variables is found by applying the linearity (3.31). Consider a linear transformation $\boldsymbol{x} = \boldsymbol{T}\boldsymbol{y}$.

Since the spread of a variable does not change if we redefine the average as zero, we can assume that $E(x_i) = 0$ without loss of generality. Then:

$$\text{Cov}(x_i, x_j) = E\left(\sum_k T_{ij} y_k \sum_l T_{jl} y_l\right) = \sum_{kl} T_{ij} T_{jl} E(y_k y_l) = \sum_{kl} T_{ij} T_{jl} \text{Cov}(y_k, y_l)\,,$$

or, in matrix notation:

$$\boldsymbol{C}_x = \boldsymbol{T}\boldsymbol{C}_y\boldsymbol{T}^T\,. \tag{3.35}$$

Chapter 4

Resolution and error analysis

One of the most important tasks of the seismic tomographer is to make sure he or she knows the limitations of the final model, and is able to convey that knowledge to others in a digestible form. This is not an easy problem: even within a narrow band of acceptable χ^2 values, there will be infinitely many models that satisfy the data at this misfit level. Yet some features will change little among those models. Such features are 'resolved' if the change is less than some pre-specified variance. Of course, one cannot calculate infinitely many models and usually resigns oneself to present one possible inversion outcome with an assessment of its resolution and uncertainty.

To estimate resolution and uncertainty is a major task that will usually consume far more time than the actual inversion. As we shall see, all of the methods we currently know have shortcomings. Our means to present the results in a accessible form are equally poorly developed. There exists also some confusion about the meaning of damping parameters and their role in resolution- and sensitivity tests. Many tomographers do not clearly distinguish between Bayesian constraints (damping parameters based on somewhat objective information) and damping parameters used to obtain a smooth model, which are inherently subjective if not based on prior information.

4.1 Resolution matrix

We restrict ourselves to the resolution and error analysis of linear problems of the form $\boldsymbol{Am} = \boldsymbol{d}$. Whatever method we use to find a solution $\hat{\boldsymbol{m}}$, the estimate can be formally expressed as a linear combination of the data:

$$\hat{\boldsymbol{m}} = \boldsymbol{A}^{-}\boldsymbol{d}\,. \tag{4.1}$$

In virtually all cases, $\boldsymbol{A}^{-}$ is not a true inverse and therefore called a 'generalized inverse'. This means that neither $\boldsymbol{A}^{-}\boldsymbol{A}$ nor $\boldsymbol{A}\boldsymbol{A}^{-}$ necessarily equals the identity matrix, but a well-designed solver will make sure that both expressions are at least close to $\boldsymbol{I}$. For the second product this implies that we impose that $\boldsymbol{A}\hat{\boldsymbol{m}} = \boldsymbol{A}\boldsymbol{A}^{-}\boldsymbol{d} \approx \boldsymbol{d}$ (good data fit). The consequences of the reverse product $\boldsymbol{A}^{-}\boldsymbol{A}$ not being equal to $\boldsymbol{I}$ become visible when we formally analyse how close $\hat{\boldsymbol{m}}$ is to the true Earth model $\boldsymbol{m}^{\text{true}}$. We write the observed data $\boldsymbol{d}$ as the sum of the 'true', i.e. error-free data and an error term $\boldsymbol{e}$:

$$\boldsymbol{d} = \boldsymbol{d}^{\text{true}} + \boldsymbol{e} = \boldsymbol{A}\boldsymbol{m}^{\text{true}} + \boldsymbol{e}.$$

For simplicity we assume that the linear mapping $\boldsymbol{Am}$ is adequate to predict the error-free data and does not introduce errors of its own, e.g. because the model parametrization is inadequate. From this we find the error in the solution $\hat{\boldsymbol{m}}$:

$$\hat{\boldsymbol{m}} - \boldsymbol{m}^{\text{true}} = \boldsymbol{A}^{-}\boldsymbol{d} - \boldsymbol{m}^{\text{true}} = (\boldsymbol{A}^{-}\boldsymbol{A} - \boldsymbol{I})\boldsymbol{m}^{\text{true}} + \boldsymbol{A}^{-}\boldsymbol{e}\,, \tag{4.2}$$

and it is clear that the total model error has two components: lack of resolution because $\boldsymbol{A}^{-}\boldsymbol{A} \neq \boldsymbol{I}$ and propagated data errors ($\boldsymbol{A}^{-}\boldsymbol{e}$).

Another way to look at the effect of resolution on model error is to recognize that in the error-free case we have:

$$\hat{\boldsymbol{m}} = \boldsymbol{A}^{-}\boldsymbol{A}\boldsymbol{m}^{\text{true}} = \boldsymbol{R}\boldsymbol{m}^{\text{true}}\,, \tag{4.3}$$

where $\boldsymbol{R} = \boldsymbol{A}^{-}\boldsymbol{A}$ is known as the resolution matrix, a kind of blurring window through which we can observe the true Earth. If the problem at hand is small enough for singular value decomposition, $\boldsymbol{R}$ can be constructed from the eigenvectors $\boldsymbol{v}_i$ of $\boldsymbol{A}^T\boldsymbol{A}$. Substituting the truncated SVD solution $\boldsymbol{A}^{-} = \boldsymbol{V}_K\boldsymbol{\Lambda}_K^{-1}\boldsymbol{U}_K^T$ we find:

$$\boldsymbol{R} = \boldsymbol{A}^{-}\boldsymbol{A} = \boldsymbol{V}_K\boldsymbol{\Lambda}_K^{-1}\boldsymbol{U}_K^T\boldsymbol{U}_K\boldsymbol{\Lambda}_K\boldsymbol{V}_K^T = \boldsymbol{V}_K\boldsymbol{V}_K^T\,. \tag{4.4}$$

The model estimate $\hat{m}_i$ is called unbiased if (4.3) yields true averages, i.e. if

$$\sum_{j=1}^{M} R_{ij} = 1\,. \tag{4.5}$$

It is clear that (4.4) does not guarantee bias-free estimates because the truncation to K eigenvectors causes $\boldsymbol{V}_K\boldsymbol{V}_K^T \neq \boldsymbol{I}$ and in fact, minimum-norm solutions are usually heavily biased. The problem is that $\boldsymbol{R}$ represents both the resolution and the bias, i.e. $\sum_j R_{ij} < 1$ (and sometimes much smaller than 1) so that $\sum_j R_{ij}m_j^{\text{true}}$ is not the true average over model parameters, but over the damped one. This reveals a shortcoming of resolution estimations using $\boldsymbol{R}$ that is serious, even though it is widely used for smaller tomographic experiments. Another shortcoming of the method is that most global tomographic problems are too large for singular value analysis, so that $\boldsymbol{R}$ cannot even be computed. However, both Vasco et al. [27] and Boschi et al. [1] have been able to compute the resolution matrix even for large systems, and for those with access to powerful parallel machines this problem may be less serious.

When calculating the posteriori model covariance $\boldsymbol{C}_{\hat{m}}$ with (3.8), we must be careful with the choice of the damping. From a Bayesian point of view, *we should only use objective constraints* on the model values itself or on their correlations (off-diagonal elements in the prior model covariance). In theory, there are no physical constraints on the correlation lengths, since the Earth may contain very sharp transitions between different compositions of the rock. This implies that we should assume the prior $\boldsymbol{C}_{\hat{m}}$ to be diagonal in (3.25) when computing $\boldsymbol{R}$ with a matrix that contains prior constraints. Thus, even if we obtain a preferred solution using smoothness constraints, it would give a falsely optimistic estimate of resolution if we include those constraints in the system when computing the resolution matrix. This confines us to simple ridge regression as in (3.16), in which the regression coefficient ϵ_n^2 assures us that the parameters of the solution remain within physically acceptable bounds. Usually, such physical constraints require an ϵ_n that is less than the one we used to obtain the preferred solution, i.e. it is less than the coefficient used to obtain a χ^2 close to N and it may not be a preferred 'smooth' solution.

The two roles of ϵ_n – the role of regularization parameter and of an objective prior constraint to the model – result in two options for the formulation of the resolution matrix. We write (3.16) in the form:

$$\boldsymbol{A}_\epsilon\boldsymbol{m} = \boldsymbol{d}_0\,, \qquad \text{with} \ \ \boldsymbol{A}_\epsilon = \begin{pmatrix}\boldsymbol{A}\\ \epsilon_n\boldsymbol{I}\end{pmatrix}, \qquad \boldsymbol{d}_0 = \begin{pmatrix}\boldsymbol{d}\\ \boldsymbol{0}\end{pmatrix}.$$

If the role of ϵ_n is only that of a damping parameter, with no prior connection to the true Earth, the trailing zero's of $\boldsymbol{d}_0$ are not considered as true 'data', and $\boldsymbol{d} = \boldsymbol{A}\boldsymbol{m}^{\text{true}}$. With that

$$\begin{aligned}\hat{\boldsymbol{m}} &= \boldsymbol{V}(\boldsymbol{\Lambda}^2+\epsilon_n^2\boldsymbol{I})^{-1}\boldsymbol{V}^T\boldsymbol{A}_\epsilon^T\boldsymbol{d}_0\\ &= \boldsymbol{V}(\boldsymbol{\Lambda}^2+\epsilon_n^2\boldsymbol{I})^{-1}\boldsymbol{V}^T\boldsymbol{A}^T\boldsymbol{d}\\ &= \boldsymbol{V}(\boldsymbol{\Lambda}^2+\epsilon_n^2\boldsymbol{I})^{-1}\boldsymbol{V}^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{m}^{\text{true}}\\ &= \boldsymbol{V}(\boldsymbol{\Lambda}^2+\epsilon_n^2\boldsymbol{I})^{-1}\boldsymbol{V}^T\boldsymbol{V}\boldsymbol{\Lambda}^2\boldsymbol{V}^T\boldsymbol{m}^{\text{true}}\\ &= \boldsymbol{V}(\boldsymbol{\Lambda}^2+\epsilon_n^2\boldsymbol{I})^{-1}\boldsymbol{\Lambda}^2\boldsymbol{V}^T\boldsymbol{m}^{\text{true}}\,,\end{aligned}$$

so that the resolution matrix for the damped solution is defined as

$$\boldsymbol{R}^{\text{damped}} = \boldsymbol{V}(\boldsymbol{\Lambda}^2+\epsilon_n^2\boldsymbol{I})^{-1}\boldsymbol{\Lambda}^2\boldsymbol{V}^T\,. \tag{4.6}$$

On the other hand, if we assume that the zero's at the trailing end of $\boldsymbol{d}_0$ and their standard deviations ϵ_n^{-1} contain real prior information about the allowed prior variance of the model parameters, then we must set $\boldsymbol{d}_0 = \boldsymbol{A}_\epsilon\boldsymbol{m}^{\text{true}}$. This gives:

$$\begin{aligned}\hat{\boldsymbol{m}} &= \boldsymbol{V}(\boldsymbol{\Lambda}^2+\epsilon_n^2\boldsymbol{I})^{-1}\boldsymbol{V}^T\boldsymbol{A}_\epsilon^T\boldsymbol{d}_0\\ &= \boldsymbol{V}(\boldsymbol{\Lambda}^2+\epsilon_n^2\boldsymbol{I})^{-1}\boldsymbol{V}^T\boldsymbol{A}_\epsilon^T\boldsymbol{A}_\epsilon\boldsymbol{m}^{\text{true}}\\ &= \boldsymbol{V}(\boldsymbol{\Lambda}^2+\epsilon_n^2\boldsymbol{I})^{-1}\boldsymbol{V}^T\boldsymbol{V}(\boldsymbol{\Lambda}^2+\epsilon^2\boldsymbol{I})\boldsymbol{V}^T\boldsymbol{m}^{\text{true}}\\ &= \boldsymbol{V}\boldsymbol{V}^T\boldsymbol{m}^{\text{true}}\,,\end{aligned}$$

with the expected outcome that the resolution matrix for a model estimate from Bayesian inference is the unit matrix:

$$\boldsymbol{R}^{\text{Bayes}} = \boldsymbol{V}\boldsymbol{V}^T = \boldsymbol{I}\,. \tag{4.7}$$

In this case we resolve every component of the model completely. Most likely, however, the posterior model variance given by (3.20) is now uncomfortably high, and we shall wish to truncate the number of eigenvectors, again introducing a bias.

One could conceivably defend a more permissive philosophy and argue that heavily oscillating solutions with strong gradients are 'unphysical', so that the regularization itself tells us what limits to impose on parameter derivatives; this would equate subjective damping parameters with objective ones and allow us to include smoothness damping even when analysing resolution. However, necessity may be the mother of invention in this case: Shearer and Earle [21], for example, analyse scattered wave energy and conclude that variations of V_P can be 3-4 per cent over distances of 4 km in the upper mantle, or 0.5 per cent over 8 km in the lower mantle. Even if such variations cannot be maintained over distances comparable to the model grid separation, this does not bode well for efforts to find objective constraints on model gradients.

It is a fact of life that inversions without smoothing constraints easily degrade into heavily oscillating tomograms, in which one cannot distinguish the wood from the trees when searching for larger structures that can be understood in terms of dynamical processes. Therefore, the damping values that we actually use in the inversion usually reflect the values that yield the maximum smoothness for the solution while still getting an acceptable data fit, rather than independent constraints on model parameters or their derivatives. But generally, prior variances based on physical limitations for the model are too large to lead to an acceptable posteriori variance. What this is then telling us is that there is no visually pleasing solution if we strive for a resolution equal to that allowed by the parametrization.

Unfortunately, many tomographers choose to analyse the resolving power using the same damping coefficients that led to a lower but acceptable data fit and a lower variance. However, if we lower K to obtain an acceptable model variance with (3.8), the resolution matrix loses much of its physical significance, because we bias our model towards zero. This may lead to meaningless results, e.g. one may damp an ill-resolved parameter to a value equal or close to zero. This parameter will have a very small variance, giving the impression we have 'resolved' it with a very small posteriori error. But the actual error in the estimate is large, being governed by the systematic error introduced by damping – the bias – rather than by random data errors.

The posteriori covariance matrix $\boldsymbol{C}_{\hat{m}}$ can also be computed from the singular value decomposition, using (3.8). Again, it is imperative that we use the full rank N when doing this. Unresolved parameters will then show up with a variance dominated by the physics-based prior uncertainty assigned to them in the Bayesian regularization.

Exercise

Exercise 4.1 Someone suggests you can undo the fact that $\sum_j R_{ij} \neq 1$ by dividing every $\hat{m}_i$ by $\sum_j R_{ij}$ and multiplying the variance by $(\sum_j R_{ij})^2$. Why would this not work?

4.2 Sensitivity tests

To obtain a quick-and-dirty estimate of the influence of damping on the solution we can apply sensitivity tests. For very large tomographic systems, such tests are often the only feasible method for getting some idea of the resolving power without taxing available computing resources beyond reasonable limits.

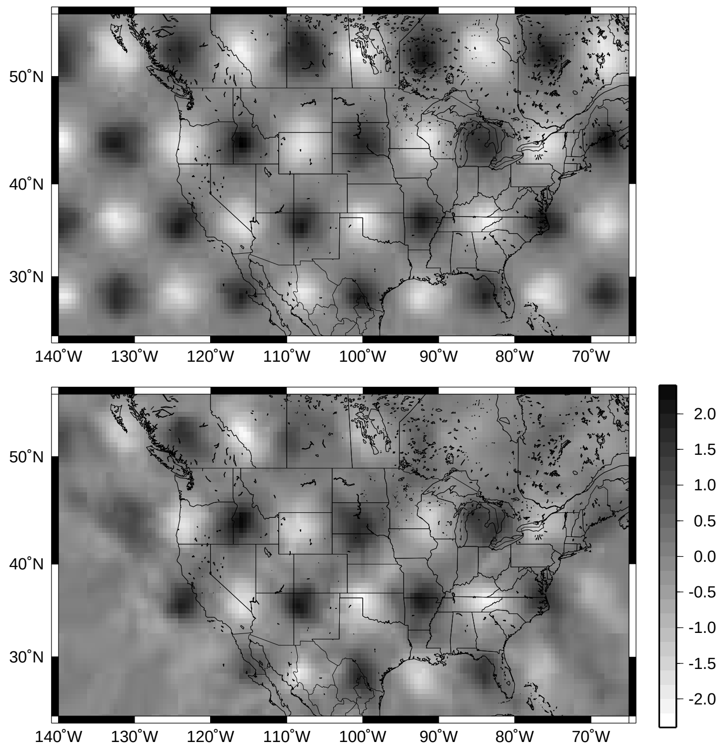


Figure 4.1: Example of a sensitivity test, using a checkerboard of Gaussian spikes with a width of 400 km, at a depth of 600 km beneath North America. Top: input model (V_P in per cent). Bottom: Result from inversion of the synthetic data generated by the input model. Figure courtesy Karin Sigloch.

The idea of a sensitivity test is very simple. Suppose we wish to know how a particular feature of the solution is influenced by the regularization, e.g. a spike-like feature at a certain depth. We design a model with just this feature, $\boldsymbol{m}_\delta$, and compute an artificial ('synthetic') data vector $\boldsymbol{d}_\delta = \boldsymbol{A}\boldsymbol{m}_\delta$. We then invert the data to get a model estimate, using the same damping parameters we used to obtain the original solution: $\hat{\boldsymbol{m}} = \boldsymbol{A}^{-}\boldsymbol{d}_\delta$, and we inspect the result by comparing the test output $\hat{\boldsymbol{m}}$ with the input $\boldsymbol{m}_\delta$. Often, a sharp feature will have spread out horizontally ('smearing'), or vertically ('leaking'). By adding random errors with a given distribution (usually Gaussian) we can study the effect of data errors on the regularized solution. An example of a sensitivity test is shown in Fig. 4.1.

To save effort, we may combine more than one feature in the synthetic model. As long as the smeared solutions do not overlap, this will still give us independent estimates of the effects of regularization. If the input features are point-like, we are testing the regularization effects for a wide spectrum of wavelengths (recall that the Fourier transform of the delta function is equal to 1). In an effort to study the resolving power globally across the model, one often distributes many spikes in a regular pattern, or uses a checkerboard-like synthetic model of alternating positive and negative anomalies. However, the gain in efficiency is somewhat offset by the fact that the regularity of the pattern introduces a dominant wavelength into the model. Leveque et al. [14] warn against a simplistic interpretation of such narrow-band tests. The panacea is to perform the test for a range of patterns covering a wide band of wavelengths. But the interpretation is now effectively in the spectral domain, and usually more difficult to judge than the space domain interpretation of simple spikes.

Note that, by allowing the damping parameters to regularize the solution beyond what is imposed by objective physical constraints, sensitivity tests are no replacement for a proper analysis of resolution and variance in the final model. Clearly, both have their roles in judging the 'preferred' model, but one should always keep in mind that the results of sensitivity tests reflect the influence of the subjective regularization of the tomographer, whereas those of a true resolution analysis are independent of the choice of final model.

I conclude this chapter by discussing various other tests that allow one to judge the reliability of the result or the adequacy of the regularization.

Backprojection of data misfits.

The data misfit vector for the solution $\boldsymbol{m}$ is $\boldsymbol{r} = \boldsymbol{d} - \boldsymbol{Am}$. A histogram of its elements should be inspected to check for anomalies far away from a Gaussian distribution. If there are strong 'tails' this may be an indication that outliers are still present in the data and need to be removed. Once a proper histogram has been obtained with $\chi^2 \approx N$, one should backproject the residual vector to the model space ($\boldsymbol{A}^T\boldsymbol{r}$) and inspect maps of backprojected misfits. These should be uncorrelated in space. If they are not randomly distributed, this may indicate an improper weighting of model parameters (through the prior covariance matrix $\boldsymbol{C}_m$ used in the regularization) or an imbalance in the spatial parametrization of the model.

Monte Carlo tests of error propagation

One may generate a data vector consisting of random noise only and invert this to judge the propagation of data errors. An obvious variant on this is to add the noise to the original data vector and inspect the variability of the resulting models. Both tests can be used to get an estimate of the model covariance by repeating them for many different random outcomes of the data vector. Such 'Monte Carlo' experiments need always be done many times to cover a range of random realizations, but limitations in the compute power often pose restrictions. E.g., Houser et al. [9] generate 100 global models in a global tomography experiment.

If we generate K models $\boldsymbol{m}^{(k)}$ in this way, an unbiased estimate of the posteriori covariance matrix is:

$$\tilde{\boldsymbol{C}}_{\hat{m}} = \frac{1}{K-1}\sum_{k=1}^{K}(\boldsymbol{m}^{(k)} - \bar{\boldsymbol{m}})(\boldsymbol{m}^{(k)} - \bar{\boldsymbol{m}})^T \,, \tag{4.8}$$

where T indicates transpose and where $\bar{\boldsymbol{m}}$ is the average over all K models generated this way. For Monte-Carlo tests, a Gaussian error distribution is acceptable provided one has removed outliers from the data set. Note again that $\tilde{\boldsymbol{C}}_m$ is only a proper statistical estimate of covariance if the Monte Carlo inversions are done with objective damping parameters.

In the absence of more detailed information on the distribution of errors, one usually assumes that these are normally distributed after the removal of outliers. However, for bad data with a very low signal to noise ratio (certainly less than 1) one can also simple scramble the elements of the data vector to obtain a noise vector with a distribution close to the actual distribution.

Cross validation, jackknifing and bootstrapping

Resampling techniques such as cross validation or bootstrapping should only be applied in seismic tomography with extreme care because they require a strictly overdetermined system. They must therefore be applied to the damped system and the damping parameter should, again, reflect true physical information on the range of model values or its derivatives (model smoothness), rather than the subjective damping that was applied to obtain a solution that pleases the eye of the geophysicist. However, if you are lucky enough to have useful prior information, these techniques may help you get robust estimates of model error and resolution without any prior assumptions about the distribution of data errors.

In the classical variant of cross-validation, one leaves out one datum, inverts the data set, then compares how well the omitted datum is predicted. One repeats this for all or a large number of data and computes the root-mean-square misfit of the predictions. This allows one to compare different models, or different damping factors.

Clearly, the method is very expensive to apply. A more efficient variant is to remove a fraction of the data (say 10%) and apply the test to the remaining data, a technique referred to as 'jackknifing'. The larger one chooses the fraction, the faster the computations are performed but the obvious trade-off is that the prediction suffers from the fact that too many of the data may be missing, and the resulting linear system may not be fully representative of the complete data set. Bootstrapping is like jackknifing, but the removed data are replaced by duplicating data randomly selected from the surviving data set (Efron [6]).

Because of the overdetermined nature of the problem, the data fit also contains information about the data errors (if most of the misfit is due to data errors and not to the inadequate nature of the parametrization!), and in fact there are optimum estimators for the damping parameters that can be obtained this way without first analysing the data to determine the standard deviations (Golub et al. [8]).

Hypothesis testing

Occasionally one wishes to test whether a particular model arising from a hypothesis, say $\boldsymbol{m}_h$, satisfies the data. Simply calculating the misfit $\boldsymbol{r}_h = \boldsymbol{d} - \boldsymbol{A}\boldsymbol{m}_h$ is in this case not sufficient, because it could be that, even if χ^2 for this particular model is too high, a small adjustment of $\boldsymbol{m}_h$ brings χ^2 within an acceptable range without invalidating the hypothesis itself. Deal and Nolet [2] introduced a test to see whether $\boldsymbol{m}_h$ is within the subspace of models that satisfy the data to an acceptable precision. There are various ways to do this. The simplest is to simply subtract $\boldsymbol{A}\boldsymbol{m}_h$ from the data and invert for the minimum norm model change to be added to $\boldsymbol{m}_h$ that yields an acceptable χ^2:

$$\begin{aligned} \boldsymbol{A}\delta\boldsymbol{m} &= \boldsymbol{d} - \boldsymbol{A}\boldsymbol{m}_h \\ \hat{\boldsymbol{m}} &= \delta\hat{\boldsymbol{m}} + \boldsymbol{m}_h \,. \end{aligned}$$

For example, Deal et al. [3, 4] invert for a regular tomographic model containing subduction zones in the west Pacific. To this they fit an analytical model $\boldsymbol{m}_h$ of slab temperature anomalies ΔT, using velocity perturbations $\delta V_P = (\partial V_P/\partial T)\Delta T$. The procedure is equivalent to adding components of the nullspace of $\boldsymbol{A}$ to the original solution $\hat{\boldsymbol{m}}$, and the operator that does this is named the ‘nullspace shuttle’.

Chapter 5

The exercises

5.1 The building blocks

We assume you have access to either a Unix or Linux terminal or a Mac. The environment we use is that of a scientists who herself does some programming, so instead of using a mouse on the graphical interface we type our commands in windows called 'Terminals'. There are different Linux flavours, but all allow you to open one or more Terminals. Try a right-click of the mouse in the middle of nowhere if you do not see a Terminal icon. On a Mac, if the Dock does not yet show a Terminal icon, go to Applications and drag the terminal icon (named Terminal.app) onto the Dock so it is more easily available. Check if an X-terminal is also available (X11.app) since this is needed for plotting (it will start up automatically when needed by SAC or GMT)[1]. I have no experience with Unix emulators on Windows systems – in principle you may be able to use Windows but I cannot help you out. For these experiments you will use Fortran programs and GMT plotting scripts. And, to read or edit them, you need an editor.

5.1.1 Text editors

/ To do the experiments, any text editor is suitable provided it does not by itself introduce non-ASCII characters into your text. If you use one of those, make sure to save your file as pure ASCII text. A very simple editor is `nano`, the open-source successor of an earlier version named `pico` (Fig 5.1). For example, if you wish to read an ascii file named `graf.out`, you simply type:

```
nano graf.out
```

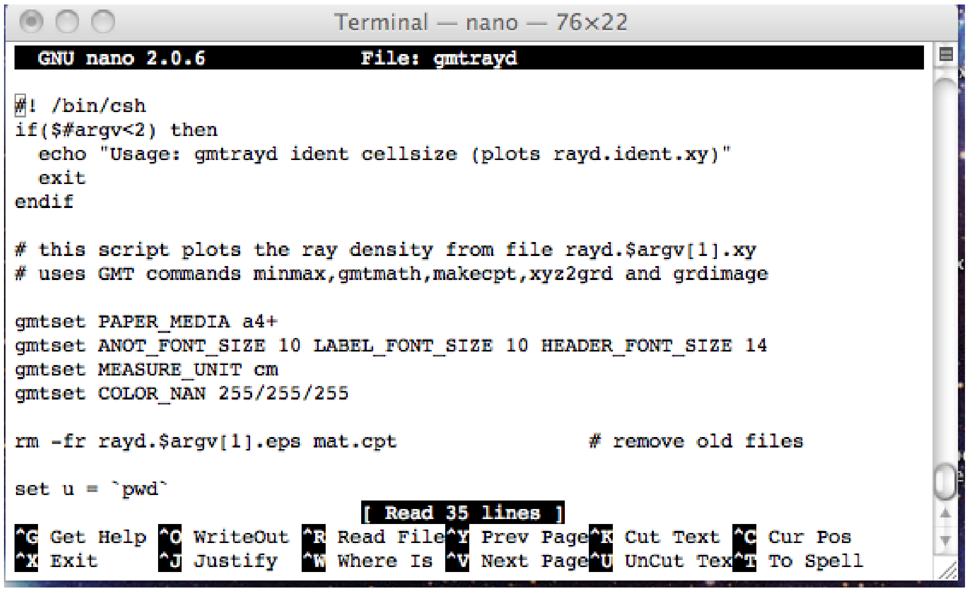
and scroll though the text using Control-V and Control-Y (Figure 5.1). You can use the cursor to move through the text, use the 'delete' key and type in new text. If you exit nano with Control-X you'll be asked if you wish to save changes.

Pico is more intuitive but also more cumbersome than `vi` (for *visual*) that most Unix users prefer to use, since it is available on every Unix system and since it allows you to make changes throughout the whole document with a few keystrokes, rather than endless mouse movements. `emacs`, available on many Unix systems, has many of the features of `vi` but a somewhat steeper learning curve. The following simple introduction to `vi` was modified from `http://www.unb.ca/csd/documentation/UNIX/tips/editors.html`:

The vi editor was designed to run on a wide range of terminals. The key Ctrl-F (hold the control key while pressing F) will scroll the data one page towards the end, Ctrl-B will scroll one page back towards the start.

The commands i, I, a, A, o, or O will put you in *insert* mode. While in insert mode whatever you type becomes part of your document. You use the "Escape" or "ESC" key to get out of insert mode and back to the ordinary command mode. The "i" command inserts before the current cursor position, "a" after the cursor, "I" at the start of the line in which the cursor is located, "A" at the end of the line, "o" inserts a new line after the current line, and "O" inserts a new line before the current line.

[1] As of Mac OS 10.5 Leopard, X11 is installed by default. If you have an older system, insert the installation disk, double-click 'optional Installs' and follow instructions. Once you have clicked on the hard disk icon to locate your installation, click on 'Applications', check the box 'X11', and click Upgrade. When installed, use Software Upgrade under the Apple menu to get the latest version.



```
  GNU nano 2.0.6              File: gmtrayd

#! /bin/csh
if($#argv<2) then
  echo "Usage: gmtrayd ident cellsize (plots rayd.ident.xy)"
  exit
endif

# this script plots the ray density from file rayd.$argv[1].xy
# uses GMT commands minmax,gmtmath,makecpt,xyz2grd and grdimage

gmtset PAPER_MEDIA a4+
gmtset ANOT_FONT_SIZE 10 LABEL_FONT_SIZE 10 HEADER_FONT_SIZE 14
gmtset MEASURE_UNIT cm
gmtset COLOR_NAN 255/255/255

rm -fr rayd.$argv[1].eps mat.cpt                # remove old files

set u = `pwd`
                         [ Read 35 lines ]
^G Get Help  ^O WriteOut  ^R Read File^Y Prev Page^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is ^V Next Page^U UnCut Tex^T To Spell
```

Figure 5.1: A screenshot of the nano or pico editor. The commands (with ˆ for the Control key) are at the bottom of the screen, below the edited text.

Other common commands:

- vi gmt1 edits file gmt1 (if gmt1 does not exist it will be created)
- Typing ":" will put you in vi's "last line" mode. In this mode you can enter syntactically complex editor commands on the last line of the display. The last line commands include things like reading in files to make them part of your document, writing out the current edit buffer to the original file or a different named file, entering find or substitute commands.
- :q Tells vi to quit. You can code :wq to write the file out then quit. Coding :q! will quit the edit without saving the changed data.
- the escape key exits you from *insert* mode. For example, if you want to quit after inserting text, you need to escape first otherwise the string ":q" would simply be added to your text. Use any of the i,I,a,A,o,O,R or C commands to go back into insert mode.
- w Move cursor to start of next word.
- $ Position cursor at end of line.
- 0 Position cursor at start of line.
- 25 Position cursor at start of line 25. G will go to end.
- /Tomo Position cursor at the next occurrence of the string "Tomo"
- x Delete character at cursor location.
- rc Replace character at cursor location with "c".
- cw Replace whatever follows cursor in the current word by what typed following the "cw" until the "escape" key is pressed.
- R Replace text at the current cursor location. The cursor will move as replacement text is typed until "escape".
- C Change from the current cursor location to end of line,
- dd Delete current line. Deleted line placed in paste buffer.
- dw Delete current word from cursor. Text to paste buffer.
- 12d Delete 12 lines from cursor moving to paste buffer.

- p Paste contents of paste buffer at current cursor location. Note that if complete lines are in paste buffer then complete lines will be pasted after current line, otherwise, the text will be pasted into the current line. Note that any command which puts text into the paste buffer will replace the text which was previously there.
- P Paste lines from paste buffer before current line.
- :s/Robert/Michael/ substitutes for the first occurrence of ”Robert” on the current line the string ”Michael”
- :s/XX/Ctrl-VCrl-M/ replaces ”XX” with a newline metacharacter (hold control key and V,M resp.)
- :1,$ s/vi/gea/g Tells vi to start at line 1 and continue through to the last line of the buffer and substitute for any occurrance of the string ”vi” the string ”gea”. The trailing ”g” which terminates the replacement string tells vi to keep looking for all occurrances of the search string on each line. Otherwise vi will only replace the first occurrance on each line.
- :w filename Tells vi to write out the current edit buffer to the named file or to the original file if filename is omitted.
- :set ic tells vi that you want case to be ignored when doing a search. Using /alpha/ will match ”alpha” or ”AlPhA” or any other variant. :se noic tells vi that you care about case again.
- :set nu Tells vi to number the lines down the left side. :set nonu Tells vi to turn off numbering.
- :23,35 m 41 Move lines 23 through 35 after line 41. A dot (.) indicates the current line.
- :54 t 54 Copy line 54 after itself.
- :500,$ d Delete from line 500 to end of document.
- :g/Mary/ s/Johnson/Wood/ change ”Johnson” to ”Wood” throughout the file, but only on lines containing ”Mary”.

5.1.2 GMT

The geographical mapping tools (GMT) are widely used in geophysical research. GMT is a collection of Unix commands that, when used in sequence, write a Postscript plotting file. They come in very handy when plotting of data with or on geographical maps is needed, but can be used as well for the plotting of non-geographical information, as we shall do. Man pages exist for every command, e.g.: `man pscoast` gives the syntax of the Unix command `pscoast`. The output of each command needs to be piped into the postscript file using a >. Sometimes one command is enough to make the file. For example:

```
pscoast -JM10 -R0/360/-50/60 -W1p -P > map.ps
gv map.ps
```

creates a Postscript file `map.ps` of width 10 cm in Mercator projection (option -JM10), with longitudes 0-360 and latitudes 50S-60N (option -R0/360/-50/60), a pen width of 1 (-W1p) and in portrait mode (-P). Try it. On my MacBook I plot postscript files with `gv` which is shorthand for `ghostview`. Other Unix computers have `ghostscript` or `gs` to plot Postscript files. There are a number of preferences to be set with GMT. In the example above, I assumed the option -JM10 to give a plot width in cm, but you may set this to inches. To find out what the defaults are on your system, type: `gmtdefaults -L`, and to learn more about the GMT settings: `man gmtdefaults`. You can change the defaults with the command `gmtset`.

To avoid repeated typing, one could put the commands `pscoast` and `gv` in a file (e.g. `runcoast` and make that executable using the Unix command `chmod a+x runcoast`. Then simply typing `runcoast` will do the job. The use of short and simple scripts is recommended as they save lots of work and avoid errors due to mistyping.

To make full use of Unix, a scripting language such as the C shell or the Bourne shell enhances the use of executable scripts by making them more flexible. Different shells have slightly different syntax, but in my experience it is enough to know a little bit of one dialect to be able to read them all. As you get used to them, you may feel tempted to write longer scripts as well, but as the complexity of the script increases they become less readable and again prone to errors! That job is best left to computer science students. Though I do not intend to give a Unix course, nor to replace the GMT Tutorial, it may be instructive to describe a short GMT script `gmtrays` that we shall use later to help you get going:

```
#! /bin/csh
gmtset PAPER_MEDIA letter+
gmtset ANOT_FONT_SIZE 10 LABEL_FONT_SIZE 10 HEADER_FONT_SIZE 14
gmtset MEASURE_UNIT cm
rm -fr rays.ps
set R = -R0/180/0/300
minmax -M rays.xy
psxy rays.xy -M  $R -JX10/-15 -W1p0/0/0 -P -Ba100:X:/a100:Z:eWnS > rays.ps
gv rays.ps
```

Though the `#`-sign is normally used in Unix scripts to indicate comments, the `#!` indicates by convention which Unix shell is used (in this case the C-shell) for the script. After the change in defaults settings with `gmtset`, we remove the old Postscript file `rays.ps` (if it exists) with the Unix `rm` command. We set the limits by storing the option `-R0/180/0/300` by storing it in a variable `R`. Since this script usually plots rays between two boreholes 180 m apart that reach to a depth of 300 m, we set to limits for the horizontal axis to 0-180 and vertical to 0-300.

The command `minmax` is a GMT command that tells you what the actual limits in file `rays.xy` are, so that you have all information to change the script if needed. The command `psxy` plots the file `rays.xy`. The option `-JX10/-15` indicates that no geographical projection should be used (X) that the plot should be 10 cm wide (10) and 15 cm high but with depth increasing downwards (-15). The only other option worth discussion in some more detail is the `-B` option which dictates the appearance of the axes. Here both the horizontal and vertical axis are annotated every 100 m (a100), and are labeled with a simple letter (X and Z respectively). Only the west (W) and South (S) axis have a label, the East (e) and North (n) are given in lower case to tell GMT only to plot tick marks (this terminology obviously comes from the plotting of maps!). In the `psxy` command the variable `R` is preceded by a dollar sign to indicate its *value* must be substituted; this is the way Unix treats variables.

We shall look at one more script (`gmthist`):

```
#! /bin/csh
if($#argv < 2) then
  echo Usage: gmthist filename interval
  echo where interval is the bin width for the histogram
  if($#argv == 1) then
    echo Limits in this file are
    minmax $argv[1]
  endif
  exit
endif
gmtset PAPER_MEDIA letter+
gmtset ANOT_FONT_SIZE 10 LABEL_FONT_SIZE 10 HEADER_FONT_SIZE 14
gmtset MEASURE_UNIT cm
rm -fr hist.ps
echo Limits in file
minmax $argv[1]
pshistogram $argv[1] -JX10 -W$argv[2] -G128 -L1 -P -Z1 -B:sec:/a10f1:Percent:eWn
S > hist.ps
gv hist.ps
```

This script plots histograms and needs to be called with a filename and a bin width for the histogram. For that reason, we first test if it is called with enough arguments. If it is less than the required two arguments (i.e. filename and bin width), a message is returned to the user with instructions how to use the script (the `echo` command). If only the filename is given, `minmax` is used to inform the user what the properties of the file are, so that an informed decision can be made for the bin width when it is called again.

To learn more about `pshistogram` or other commands and their options, use the Unix `man` command. Refer to the GMT tutorial or manual for more general information. If you intend to use GMT in your own research, take an afternoon to work through the tutorial.

5.1.3 Fortran

We generally use the Fortran language for scientific programming. Popular tools such as Matlab or Excel are not designed for computing on parallel computers, and tomographic computations are usually massive enough to make the use of parallel computing desirable or even necessary (though not in these experiments). Despite the introduction of many other programming languages, Fortran stood the test of time and is one of the major programming tools for scientists. Subroutines for parallel computing (e.g. MPI routines - message passing interface) are available in Fortran and C. To digest/plot numerical output from the Fortran programs, it is perfectly fine to use Matlab or Excel.

For these experiments you do not need to program yourself, since the software has been pre-written. We use the software as a laboratory instrument. But sometimes you my wish to tweak even a lab instrument, and if you know a little Fortran, feel free to modify software if you want to explore something in addition to what is foreseen in this manual. In that case copy the program to a separate directory so that the original version remains untouched. After making changes, the following command compiles your source code into an executable file named `program1` - assuming your program is called program1 and uses separate subroutines stored in subs1.f:

```
gfortran program1.f subs1.f -o program1
```

which creates an executable named program1 (the -o flag stands for object code).

5.2 Ray tracing and Fermat's Principle

We shall first attempt to trace rays through a model, and investigate the change in travel times when we perturb the ray from its minimum travel time path.

5.2.1 Graph theory in a homogeneous model

Rays in a homogeneous model should be straight lines. But when searching the shortest path following a set of model nodes, the rays must follow nodal lines. It is as if one wishes the streets of New York, which form a regular network of NS-oriented avenues perpendicular to EW-oriented streets, along a diagonal.

Graph theory (We do not have time to explain the details, but see 'A Breviary of seismic tomography', chapter 3) is designed to find the shortest path along nodes in a very efficient way. Not surprisingly, graph theory is the workhorse for GPS receivers in cars that point you the way. How one connects models nodes defines how the rays can change direction: they are forced to follow the 'streets' open to them. This introduces an error – rays that theoretically should be straight lines, for example, have to turn corners if there is not direct 'street' available. It is a price we pay for the stability In this experiment we study how the angle discretization imposed by graph theory affects the tracing of the straight rays. As a by-product we can test the validity of Fermat's Principle: the graph-theoretical rays deviate from straight lines, but the travel time error should be of second order only.

For this experiment you need:

- the Fortran program `graf`
- the homogeneous model in file `modelzero`
- the GMT script `gmtray` and the model in gmt format `modelzero.xy`
- the GMT script `gmttimes`
- an editor such as `vi`

First copy the files `modelzero` and `modelzero.xy` from directory 'models' to your own directory, e.g.:

```
cp ../models/modelzero .
```

The two periods indicate 'parent directory', the directlry directly above the current one. Note that the last period (meaning the 'current directry') must be typed.

Inspect the file `modelzero` with the editor. All our model files will have this structure. The numbers on the first line (20, 30 and 10.) represent the width and depth of the model in number of cells, and the width of each cell in meters. This is a borehole experiment, with 300 m deep boreholes forming the left and right edge of the model. The left borehole has sensors at 5,15,25,...,295 m deep. You can place a source at any one of these depths in the right borehole, `graf` then computes the rays to all the sensors from this source.

Below that, are a list of numbers – on close inspection a list of 30 lines with 20 $\times$ the number '50'. By convention, the number x represents a perturbation of $x - 50\%$ on a background model slowness of 1/5000 s/m (v=5000 m/s). The numbers need not be separated by spaces or comma's, since the Fortran code has a statement:

```
      do j=1,nz
        read(1,fmt='(50i2)') (kslwns(i,j),i=1,nx)
      enddo
```

which means that the program read `nz` lines of (at most) 50 numbers in 'i2' (integer of length 2) format. In fact, it only reads 20 (`nx`) numbers and stores these in row i of array `kslwns`.

Try first running the Fortran code by typing `graf` in respnse to the Unix prompt, and give the input you are being asked for. On my screen I got, for example:

```
 Give model file (e.g. modelzero):
modelzero
 Opened: modelzero
 dimensions: 20 30, cell size:  10.
 slowness model read, n= 600
 Minimum slowness:  0.000199999995 (v=  5000. m/s)
 Maximum slowness:  0.000199999995 (v=  5000. m/s)
 Give depth of source (in rightmost borehole):
45
```

If you now type `ls`, the Unix command to list the contents of the directory, you see that `graf` has created new files: an output file `out.graf`, and two gmt plotting files, `modelzero.rays.xy` and `modelzero.tt.xy`. We shall inspect the ray geometry first, typing

`gmtrays modelzero`

If it complains the file with the model may be missing – you can copy the GMT plotting file for homogeneous model to your own directory with:

`cp ../models/modelzero.xy .`

Answer the following questions:

1. Which rays are straight, and why do you think these are straight and others not?
2. Inspect the travel time error for these rays by opening file `graf.out` in an editor. The graph-theoretical time is in a column headed 'tgraf', whereas the travel time for a homogeneous model with a velocity of 5000 m/s is in column 'tzero'. Since our model has indeed a background velocity of 5000 m/s, this is the correct travel time. What do you observe?
3. Inspect the errors for other rays. What is the largest relative error?
4. Why is the average error not close to zero?
5. Assume the dominant frequency of the waves is 250 Hz, and that you can identify the onset of the arrival with a precision of a quarter of the period. Is the graph-theoretical error acceptable in view of the precision of the time observations?
6. Run `gmttimes` to plot the travel time field `tt.xy`. Does it show what you would expect?

Confirm your answers by trying a few more depths.

5.2.2 A magma chamber model

Next, we experiment with a model for a magma chamber, in file `modelmagma`. You can plot this model by copying the gmt plotfile to `modelmagma.xy`, the file that is expected by the GMT script `gmtmodel`. Or:

```
cp ../models/modelmagma.xy .
gmtmodel modelmagma
```

Can you predict what will happen to rays that used to go straight to the region that now has up to 40% higher slowness?

Run `graf` on this model and plot the results with `gmtrays`. Do the rays behave as expected?

5.3 Designing an experiment, computing the matrix

Although we shall 'imitate' the field situation on the computer, many steps we take in this experiment are fairly realistic for a real set-up of a cross-borehole experiment. The only shortcut we take is that we do not fire the shots and measure the travel times from seismograms. Instead we take a shortcut and obtain our data by *computing* them from the model . This is of course cheating because in reality we do not know the model – if we did there would be no need to do the experiment! But once we have the data, we completely igore the knowledge of the model, and try instead to invert for it. But that is for the next section. In this part, we shall try to design an experiment.

The space in between the two boreholes is divided up into cells. The number of cells in the grid with unknown slowness anomaly is equal to the number of columns in the matrix. The number of source-geophone pairs is the number of data and therefore the number of rows. The larger the matrix, the more timeconsuming the inversion will be. Obviously, many small cells allow us to see more detail than few bigger ones. But very small cells may be difficult or impossible to resolve, and make the extra computing time unnecessary.

A similar trade-off exists for the design of the source-geophone array. An optimum coverage requires geophones in *both* boreholes, but this is expensive. So we opt for sources at the surface and in the right borehole, and geophones only in the left borehole, leading to an a-symmetric coverage. In practice there is also an economic trade-off between the number of sources and sensors, and their quality – often the cost of mainenance are high and one accepts that a fraction of sources and geophones fails. The result is that the design of the experiment offers a number of choices with trade-offs in terms of costs and quality of results. In the remaining experiments we shall study the image quality one obtains for various choices.

5.3.1 Design of grid and geophone/source configuration

The design of grid and source/sensor configuration is done by means of program `expsetup`, named such because the configuration determines the matrix. Here is an example of a run with `expsetup` as it appears on the screen:

```
% expsetup
% Give size of cells in m (e.g. 10):
%10
% nx, nz= 18 30
% Total number of cells is  540
 Give source/geophone spacing (eg 10):
10
 What percentage of sources fails?
5
 What percentage of individual rays fails?
5
 Give matrix ident:
test1
```

The last entry allows us to identify this particular configuration in subsequent runs that use the output of `expsetup`. This created a model of 180 $\times$ 300 m (i.e. boreholes are 300 m deep and 180 m apart). The failure rates allow one to experiment with realistic borehole tomography configurations: a source failure is more serious than the failure of one seismogram (i.e. because of a spike in electronics) since it eliminates all rays from the source. The source spacing is the same as the geophone spacing. We inspect the output file `expsetup.out`. On my machine the last few lines are:

```
 Nr of working sources: 44 (out of 47)
 Nr of working geophones: 29 (out of 30)
 Number of data= 1260, parameters= 540
```

On other machines, numbers may differ because the failure is determined by a machine-dependent random number generator (which is also why they may differ somewhat from the 5% specified). Now do the following experiment:

1. Do your own run of `expsetup`. Feel free to vary the numbers.
2. One way to plot the ray coverage is to sum each matrix column, i.e. to compute the *ray density* $\rho_i = \sum_{j=1}^{N} A_{ij}$. Explain why ρ_i is the total length of rays in cell i.
3. Plot ρ_i using the script `gmtrayd test1 10` (the arguments are the matrix ident and the cell size). Explain what you see. How homogeneous is the coverage? Can you identify missing sources or geophones?
4. Are there more data than unknowns in your matrix?
5. Do at least one more run with different failure rates for sensors/sources. Be sure to save the results of new runs with a different identification

Hint: typing `cleanup` activates a script that trashes all the plotfiles that may clutter your directory. Typing `cleanup all` also removes the output from the programs that compute matrices and eigenvalues, and obviously needs to be called only if you wish to start from scratch.

5.3.2 The spectrum of eigenvalues

The more (relatively) large eigenvalues the matrix has, the more constraints the data impose on the model. It is thus important to inspect the 'spectrum' of eigenvalues. To do so, first compute the eigenvalues and eigenvectors:

```
% svdtomo
 Give matrix ident:
test1
 This matrix has  1260 rows and  540 columns.
 progress svdcmp:
Householder:  xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
LH transforms:yyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyy
RH transforms:zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz
Diagonalize:  .........................................................
```

The progress of the program is followed on screen. All eigenvalues are listed in the file `svdtomo.out`, but it is faster to inspect them graphically:

```
gmteigenvalues test1
```

In this plot, the largest eigenvalue is scaled to 1. Since the standard deviation of the resulting model parameters scales inversly with λ, the smallest eigenvalues have a contribution to the error that is out of proportion. To investigate this, inspect the tail end of the eigenvalue distribution (in `svdtomo.out`) for the configurations designed in the previous task, and compare. Select the configuration that seems to have the most eigenvalues above some useful threshold (i.e. 0.2 times the maximum) for the experiments in the next section.

5.3.3 The eigenvectors

The larger an eigenvalue, the more precisely the contribution of the corresponding eigenvector is constrained by the model. It is thus of interest to inspect what the eigenvectors 'look like' in different parts of the eigenvalue spectrum. You can create plottable files using program `plotvector`. Here is an example:

```
% plotvector
 Give eigenvector file ident:
test1
 m,n,nx,nz,h= 1260 540 18 30  10.


Eigenvalues:
 1  354.793983
 2  261.907887
 3  234.647438
 4  214.070391
 5  209.264056
 100  78.1680816
 200  57.8295911
 300  38.5710624
 400  13.5071891
 500  2.45769526
Give eigenvector nr (stop on 0): 1
Give eigenvector nr (stop on 0): 2
Give eigenvector nr (stop on 0): 3
Give eigenvector nr (stop on 0): 500
Give eigenvector nr (stop on 0): 0
```

Which results in files named eigv001.xy, eigv002.xy etc. Typing `gmteigv 001` will plot the first eigenvector, etc. Do the following:

1. Create a selection of eigenvector files in this way, be sure to include 1,2,3 and 500 and a few others,
2. Inspect the plots of vectors belonging to large eigenvalues. How would you describe them? Can you explain some of their characteristics in a general way?
3. Similarly, how do the eigenvectors for small eigenvalues (or zero!) differ from the first few? Can you give a qualitative explanation for what you observe?

5.4 The inverse problem

5.4.1 Generating data

We don't have to go out into the field to get useful data: since we can create models, and we know the linear relationship between the data vector d and the model m is linear: $d = Am$, we can use the matrix A to generate artificial data for that model. This is sometimes called 'solving the forward problem'. This strategy has the added advantage that we can compare the result of an inversion with the model we used to create d. Program `predictdata` generates such (error-free) data for you configuration *for a specified model of heterogeneity*. Here we shall first test if we can image the magma model. Using the configuration in test1, I get:

```
% predictdata
 Give matrix ident:
```

```
test1
 nr of data= 1260, nr of parameters= 540
 mdim, ndim= 1600 1500
 nx,nz,h= 18 30  10.
 a has been read
 Give model file name (e.g. modelzero):
modelmagma
 model has nnx,nnz= 18 30
 model has been read
```

This creates a file with synthetic 'data' (delay times with respect to a homogeneous model with a background velocity of 5 km/s): `delays0.test1.modelmagma`. A histogram of the data can be produced with GMT: `gmthist delays0.test1.modelmagma 0.0002` (the 0.0002 is the interval width for the histogram).

Do the following experiment:

1. Run predictdata on the magma chamber model
2. Inspect the histogram for the range of (still exact) delay times generated by the slow magma anomaly.
3. Next, make the experiment more realistic by adding observational errors to the delay time. This is done by running program `adderr`, which adds errors with a Gaussian distribution to the original delays and creates a new data file. For example:

   ```
   % adderr
    Give matrix ident:
   test1
    Give standard deviation for data errors (eg 0.0001):
   0.0001
    Give integer for random seed (eg 29456):
   25637
    Give filename for new data:
   delays1
    Total nr of data= 1261
   ```

   The random seed number can be used to generate different sequences of random errors, useful if you wish to generate models with different realizations of the random errors.
4. inspect the histogram of the new data. As always, ask yourself questions and try to explain what you see.

5.4.2 Solving for a model

This is done using program `svdsolve`. Though the program allows both for a sharp and a tapered cut-off of the eigenvalues (equations 4.7 and 4.18, respectively), for the experiment we shall limit ourselves to the sharp cut-off case. When you start the program, it lists some of the eigenvalues, as well as the (rms) average standard deviation for the model parameters if the cut-off occurs at that level:

```
 svdsolve
 Give matrix ident:
test1
 Matrix has  1234 rows and  540 columns
 Now opening eigv.test1
 Give data file name (e.g. delays0.xxxx):
```

```
delays1
 Data standard deviation:   9.99999975E-05
 Now computing A^T d
 Now computing V^T A^T d

 RMS model error as function of cut-off:
    i         w    sigrms
    1    353.61 0.0000000
    2    259.80 0.0000000
    3    235.88 0.0000000
```

The sigma's are for the absolute slowness variations (with respect to the background slowness of 0.0002 s/m). More important is to monitor the value of χ^2 when you do inversions for different cut-off levels K. For example:

```
 Tapered [0] or sharp [1] eigenvalue cut-off?
1
 Give rank k (or 0 to stop):
100
 Give ident for this solution:
s100
 Model GMT file is in sol.s100.xy
 Error GMT file is in sig.s100.xy
 Data fit GMT file is in dif.s100.xy

Chi^2=    2391.9, relative Chi^2=    1.9383


 Give rank k (or 0 to stop):
400
 Give ident for this solution:
s400
 Model GMT file is in sol.s400.xy
 Error GMT file is in sig.s400.xy
 Data fit GMT file is in dif.s400.xy

Chi^2=     806.1, relative Chi^2=    0.6533
```

shows that the optimum relative χ^2 of 1 is somewhere between the cut-off levels of 100 and 400 eigenvalues. The two models so created are in files sol.s100.xy and sol.s400.xy and can be plotted with GMT, eg `gmtmodel sol.s100`.

1. Find the optimum model with χ^2 close to 1
2. Plot this model
3. invert the error free data also with this cut-off level and compare the two solutions,
4. Investigate for what K the model errors start to 'blow up'

5.5 Sensitivity analysis

We do the sensitivity analysis by inverting for checkerboard patterns. To this end, directory MODELS contains checkerboard patterns of different size: model2chess, model4chess and model6chess.

1. Generate error-free data for these models using predictdata

2. Invert the error-free data using the cut-off K that you found with the previous experiment. Which regions of the model are very well resolved? Is there a limit to the wavelengths that can be resolved?

3. Add errors to the data. Which regions show the largest effects of errors?

Bibliography

[1] Boschi, L. Measures of resolution in global body wave tomography. *Geophys. Res. Lett.*, **30**:doi:10.1029/2003GL018222, 2003.

[2] Deal, M.M. and Nolet, G. Nullspace shuttles. *Geophys. J. Int.*, **124**:372–380, 1996.

[3] Deal, M.M. and Nolet, G. Slab temperature and thickness from seismic tomography 2. Izu-Bonin, Japan and Kuril subduction zones. *J. Geophys. Res.*, **104**:28803–28812, 1999.

[4] Deal, M.M., Nolet, G., and van der Hilst, R.D. Slab temperature and thickness from seismic tomography, 1. Method and application to Tonga. *J. Geophys. Res.*, **104**:28789–28802, 1999.

[5] Dziewonski, A.M. Mapping the lower mantle: Determination of lateral heterogeneity in P velocity up to degree and order 6. *J. Geophys. Res.*, **89**:5929–5952, 1984.

[6] Efron, B. Estimating the error rate of a prediction rule: Improvement on cross-validation. *J. Am. Stat. Ass.*, **78**:316–331, 1983.

[7] Franklin, J.N. Well posed stochastic extention of ill-posed linear problems. *J. Math. Anal. Appl.*, **31**:682–716, 1970.

[8] Golub, G.H., Heath, M., and Wahba, G. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, **21**:215–223, 1979.

[9] Houser, C., Masters, G., Shearer, P.M., and Laske, G. Shear and compressional velocity models of the mantle from cluster analysis of long-period waveforms. *Geophys. J. Int.*, subm. 2007.

[10] Jackson, D.D. The use of a priori data to resolve non-uniqueness in linear inversion. *Geophys. J. Roy. astr. Soc.*, **57**:137–157, 1979.

[11] Jeffreys, H. On travel times in seismology. *Bur. Centr. Seism. Trav. S.*, **14**:3–36 (reprinted in The Collected Papers of Sir Harold Jeffreys, Vol. 2, Gordon and Breach 1973), 1936.

[12] Johnsonbaugh, R. *Discrete Mathematics*. Macmillan, New York, 1984.

[13] Kuo, B.-Y., Garnero, E.J., and Lay, T. Tomographic inversion of S-SKS times for shear velocity heterogeneity in D'' : Degree 12 and hybrid models. *J. Geophys. Res.*, **105**:28139–28157, 2000.

[14] Leveque, J.-J., Rivera, L., and Wittlinger, G. On the use of the checker-board test to assess the resolution of tomographic inversions. *Geophys. J. Int.*, **115**:313–318, 1993.

[15] Masters, G., Johnson, S., Laske, G., and Bolton, H. A shear velocity model of the mantle. *Phil. Trans. R. Soc. Lond.*, **A354**:1385–1410, 1996.

[16] Moser, T.J. Shortest path calculations of seismic rays. *Geophys.*, **56**:59–67, 1991.

[17] Moser, T.J., Nolet, G., and Snieder, R. Ray bending revisited. *Bull. Seismol. Soc. Am.*, **82**:259–288, 1992.

[18] Nakanishi, I. and Yamaguchi, K. A numerical experiment on nonlinear image reconstruction from first-arrival times for two-dimensional island arc structure. *J. Phys. Earth*, **34**:195–201, 1986.

[19] Nolet, G. Seismic wave propagation and seismic tomography. In G. Nolet, ed., *Seismic Tomography*, pages 1–23. Reidel, Dordrecht, 1987.

[20] Press, W.H., Teukolsky, S.A., Vettering, W.T., and Flannery, B.P. *Numerical Recipes*. Cambridge University Press, Cambridge, UK, second ed., 1992.

[21] Shearer, P.M. and Earle, P.S. The global short-period wavefield modelled with a Monte Carlo seismic phonon method. *Geophys. J. Int.*, **158**:1103–1117, 2004.

[22] Spakman, W. and Nolet, G. Imaging algorithms, accuracy and resolution in delay-time tomography. In N. V. et al., ed., *Mathematical Geophysics*, pages 155–187. Reidel, Hingham, Mass, 1988.

[23] Tarantola, A. *Inverse problem theory*. Elsevier, Amsterdam, 1987.

[24] Tarantola, A. *Inverse problem theory and methods for model parameter estimation*. SIAM, (available on line from `www.ipgp.jussieu.fr/~tarantola/Files/Professional/Books/index.html`), 2005.

[25] Tikhonov, A.N. Solution of incorrectly formulated problems and the regularization method. *Dokl. Akad. Nauk. SSSR*, **151**:501–504, 1963.

[26] Trampert, J. and Snieder, R. Model estimations biased by truncated expansions: possible artifacts in seismic tomography. *Science*, **271**:1257–1260, 1996.

[27] Vasco, D.W., Johnson, L.R., and Marques, O. Resolution, uncertainty, and whole Earth tomography. *J. Geophys. Res.*, **108**:doi:10.1029/2001JB000412, 2003.

Index