Chapter 3

The Inverse Theory and Tomography

In the inverse problem one considers the observations as the basis to obtain the values of some parameters that are related to them through a model. Tomography is a subset of the inverse problem in which measurements are the integration of the model parameters along ray paths. Tomography is a common practice in geophysics using earthquakes or acoustic signals as a source of observations. In this chapter we will describe the mathematical background for tomography and the technique used for solving the inverse problem. Since the parameters describing the atmosphere are time-dependent, the implementation of a Kalman filter will also be presented.

3.1 The inverse problem

3.1.1 Formulation

A definition of the Inverse Problem can be found in [41]: "to use the actual results of some measurements of the observable parameters to infer the actual values of the model parameters". In most cases, however, the set of measurements over-determines some model parameters and under-determine others. This is due to the data not being completely sounding the physical system or to the data being too uncertain (*noisy*) in some regions.

The physical system under study (e.g. ionospheric electron density or wet refractivity) can be described using different model parameters. The abstract space containing as points all possible modeling of the system is the *model space*. Each point is named a *model* and is represented by $\mathbf{m}$, in turn represented by a particular set of values for the model parameters, once we define a particular coordinate system. This is called a *parametrization* of the system.

On the other hand, we have the *data space*. This space is defined as the space of all conceivable instrumental response, and a particular realization will be noted as $\mathbf{d}$. This is the *data set*.

The model and the data sets can in general be related by

$$\mathcal{F}(\mathbf{d}, \mathbf{m}) = 0. \tag{3.1}$$

It is desirable to be able to separate the model parameters from the data set and then form a set of explicit equations:

$$\mathbf{d} = \mathbf{g}(\mathbf{m}); \tag{3.2}$$

if the relationship between model parameters is also linear, then we can write the equation in matrix notation as

$$\mathbf{d} = \mathbf{G}\mathbf{m}, \tag{3.3}$$

where $\mathbf{G}$ is called the data kernel. This latter expression is the most common and, in particular, leads to the formulation of the discrete linear inverse problem.

Tomography is a subset of inverse theory, in which the data kernel is formed by integrating the model parameters along ray paths. We can hence write ([42])

$$d_i = \int_{r_i} m[\vec{r}]d\vec{r}, \tag{3.4}$$

where r_i is the ray path i; the continuous representation of the inverse theory, however, has severe limitations: by considering zero-thickness rays crossing the physical system, we may find that these rays do not cross each other, hence leading to indetermination of the system. To overcome this, rays must have some thickness (being tubes instead of lines) or the model parameters have some smoothness, not letting the solution vary within a certain region. This is accomplished by discretizing the system under study and thus, rewriting the above expression in the form:

$$d_i = \sum_j G_{ij} m_j, \tag{3.5}$$

which is alike Equation 3.3. This will form the linear system to be solved for: if we can construct the data kernel coefficients G_{ij}, then Equation 3.3 is solved by inverting the matrix $\mathbf{G}$. To this end, we can express the state of our physical system ($\mathcal{N}$) at a given time and location as a linear combination of a set of basis functions Φ,

$$\mathcal{N}(\vec{r}, t) = \sum_j \Phi_j(\vec{r}) m_j(t), \tag{3.6}$$

where m_j are the coefficients of the linear combination. The observations in tomography are integrated measurements along ray paths and therefore we can write:

$$d_i(t_k) = \int_{r_i} \mathcal{N}(\vec{r}, t_k) = \sum_j \int_{r_i(t_k)} \Phi_j \; m_j(t_k) ds \tag{3.7}$$

and then we can define the data kernel coefficients as

$$G_{ij} = \int_{r_i(t_k)} \Phi_j ds, \tag{3.8}$$

that is, the integral of the basis functions along ray paths.

3.1.1.1 Set of basis functions

The choice of the basis functions defines the number of unknowns in the linear system. The most straightforward representation of $\mathcal{N}$ is the spatial discretization: we consider $\mathcal{N}$ to be a density and then discretize the three-dimensional space in finite volume pixels (henceforth called *voxels*), in which the density remains constant. This is the direct application of discrete theory to the tomography problem: if one has no a priori knowledge of the state or does not want to apply any model, the discretization is the most general approach.

Another approach is to use empirically determined models: given a set of measurements of the actual distribution of $\mathcal{N}$ (not only integral measurements) in different situations, one can form a set of orthogonal functions using an orthonormalization process. These are called Empirical Orthogonal Functions (EOFs) and contain information on known features of the field. This incorporation of a priori knowledge in the set of basis functions is one great advantage in using EOFs. However, this can be the main drawback: unexpected behaviour (the most interesting) is badly modeled. Other sets of basis functions such as spherical harmonics, wavelets or considering the solution to be a linear combination of piecewise exponential functions are also possible.

The evaluation of the integral of the basis functions (G_{ij}) is the second consideration before choosing the set of basis functions. In the voxel case, the coefficient G_{ij} is the length of the ray i across voxel j (see Figure 3.1). In other cases, there is no such straight physical interpretation. The integral may be found starting from the value of $\mathbf{G_i}$ computed using voxels and then performing a change of base.

It is worth mentioning that we can obtain an explicit and linear system if we assume that the rays are not bended as a consequence of the solution field. Otherwise, we should use Equation 3.1 and another approach to solve the system.

As a consequence from this discussion, it is apparent that the voxel-based tomography introduces no physics knowledge of the atmosphere behaviour other than some continuity and smoothness of the fields in space and time. This, of course, gives a robust system when observed conditions are ill-modeled. On the other hand, one should not discard the use of some modeling when it is founded on solid theory and, in fact, the voxel-based tomography is an intermediate step towards the final ingestion of GPS observables into Numerical Weather

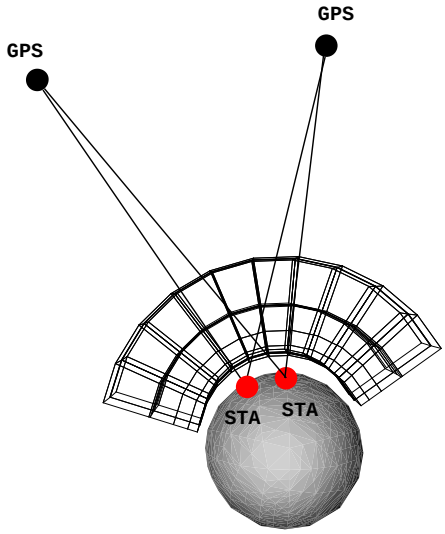

Figure 3.1: Discretization of the space using voxels (not to scale). In the present implementation, it is considered that ray bending is negligible.

Prediction Models. Basically, the NWPM are sets of equations governing physical relationships which are solved based on measurements, correctly relating information coming from different sources. In fact, the ingestion of GPS data in ionospheric models was attempted in [43].

3.1.1.2 Construction of the linear system

Without loose of generality, we will now consider the voxel discretization of the space as the set of basis functions. Mathematically they are described as:

$$\Phi_j^{voxel}(\vec{r}) = \begin{cases} 1 & \text{if } \vec{r} \in V_{voxel} \\ 0 & \text{otherwise} \end{cases} \tag{3.9}$$

Each measurement will be expressed as:

$$d_i = G_{i,1}m_1 + G_{i,2}m_2 + ... + G_{i,N-1}m_{N-1} + G_{i,N}m_N \tag{3.10}$$

and form a linear model in which $\mathbf{m}$ is the vector of N unknowns and $\mathbf{d}$ is the vector with M observations, being $\mathbf{G}$ the MxN matrix that linearly relates them. The usual notation in linear algebra is to take $\mathbf{x}$ as the vector of unknowns, $\mathbf{y}$ as the vector of observations and $\mathbf{A}$ as the matrix. In order to separate the discussion of the modelization itself from that of the solving process, we will use the notation

$$\mathbf{y} = \mathbf{A}\mathbf{x} \tag{3.11}$$

in the following subsections to ease the recognition of the linear system.

3.1.1.3 Coping with the indetermination of the system: constraints

It has been stated that the inverse problem usually suffers from an over-determination of some of the model parameters, while others are under-determined. This is translated in that the matrix of Equation 3.11 does not have a natural inverse because of the lack of data. This can be surmounted by adding information to the system (see Section 3.3 for a probabilistic approach to the lack of information). We can introduce additional equations and include them as constraints. These constraints impose smoothness in the solution: a voxel has to be similar to its neighbours. In that case, a voxel not hit by any ray will extract the information from its vicinity, while an over-determined voxel will fix its value through the data and contribute to its neighbours determination. These equations can be written as a new set:

$$\mathbf{l} = \mathbf{B}\mathbf{x} \tag{3.12}$$

which can be stacked to the system of Equation 3.11 and form a complete system as:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{l} \end{bmatrix} = \begin{bmatrix} \mathbf{A} \\ \mathbf{B} \end{bmatrix} \mathbf{x} \implies \hat{\mathbf{y}} = \mathbf{S}\mathbf{x}. \tag{3.13}$$

An approach in the spectral domain to the smoothing concept can also be implemented in the frame of the least squares approximation, as was discussed in [44]. Finally, it should be noted that the entropy can also be used as a method to smooth the solution but it has the main drawback of acting globally on the solution, while the addition of equations of constraints acts at a local scale, compensating only where there is not enough data available.

3.1.2 Solving process

3.1.2.1 The least squares approximation

The linear system $\hat{\mathbf{y}} = \mathbf{S}\mathbf{x}$ can be solved by making use of the length of the error between the predicted measurements that an estimation $\mathbf{x}$ implies: $\hat{\mathbf{y}}^{\mathbf{pred}} = \mathbf{S}\mathbf{x}^{\mathbf{est}}$. That is, minimizing the norm of the error defined as

$$\mathbf{e} = \hat{\mathbf{y}} - \hat{\mathbf{y}}^{\mathbf{pred}}. \tag{3.14}$$

The choice of the norm to use depends on how accurate our data are and what statistics the data obey. In general, the n-norm (L_n) is defined as

$$\|\mathbf{e}\|_n = \left[\sum_i |e_i|^n\right]^{1/n}. \tag{3.15}$$

As shown in [42], higher norms give the largest weight to the largest component of the error vector. Therefore, if we expect our data to be very accurate, a large error in the solution

has a high importance and has to be strongly reduced; one may use high order norms. On the other hand, if we expect a large scatter in the data, large errors have the same impact as small errors; in such case, a lower order norm should be used. The L_2 norm is the most widely used because, while being a compromise, it implies a Gaussian distribution in the data: a probabilistic analysis can be followed [41] to prove that in the case of uncorrelated Gaussian data, the minimization of L_2 or *least squares solution* is exactly the same as the *maximum likelihood solution.* The assumption of Gaussian distribution is the most common approach in many engineering problems. The total error (E) is now:

$$\begin{aligned} E &= \mathbf{e}^T \mathbf{e} \\ &= (\hat{\mathbf{y}} - \mathbf{S}\mathbf{x})^T (\mathbf{C}_{\hat{y}})^{-1} (\hat{\mathbf{y}} - \mathbf{S}\mathbf{x}) = \chi^2 \end{aligned} \tag{3.16}$$

which in the last equality is expressed as a functional to be minimized ($\mathbf{C}$ is the covariance matrix). The minimum can be found by an iterative method or by applying the closed expression for the least-squares solution:

$$\mathbf{x}^{est} = \left[\mathbf{S}^T (\mathbf{C}_{\hat{y}})^{-1} \mathbf{S}\right]^{-1} \mathbf{S}^T (\mathbf{C}_{\hat{y}})^{-1} \hat{\mathbf{y}} \tag{3.17}$$

$$\mathbf{x}^{est} = \left[\mathbf{A}^T (\mathbf{C}_y)^{-1} \mathbf{A} + \lambda \mathbf{B}^T \mathbf{B}\right]^{-1} \mathbf{A}^T (\mathbf{C}_y)^{-1} \mathbf{y} \tag{3.18}$$

where we have considered that the equations of constraint are homogeneous ($\mathbf{l} = 0$). Note that the matrix to be inverted, $\mathbf{S}^T (\mathbf{C}_{\hat{y}})^{-1} \mathbf{S}$, is an NxN matrix and $\mathbf{S}^T (\mathbf{C}_{\hat{y}})^{-1} \hat{\mathbf{y}}$ is an N-size vector. This allows the application of different techniques to find the inverse. The term $\left[\mathbf{S}^T (\mathbf{C}_{\hat{y}})^{-1} \mathbf{S}\right]^{-1} \mathbf{S}^T (\mathbf{C}_{\hat{y}})^{-1}$ is usually known as *generalized inverse* of $\mathbf{S}$ in the least squares solution of an inverse problem (in fact the term *generalized inverse* has a much broader definition, see [45], Chapter 7, for details). In Section 3.1.2.3 we shall summarize the technique known as Singular Value Decomposition to compute the generalized inverse.

3.1.2.2 The Constraints in the Spectral Domain: Minimizing a Functional

In [44] we introduced the concept of limiting the spatial frequency spectra of the solution by including, in the functional to be minimized, the filtered expression of the autocorrelation of the solution. This is to write:

$$G(x, \lambda) = (\mathbf{y} - \mathbf{A}\mathbf{x})^T (\mathbf{C}_y)^{-1} (\mathbf{y} - \mathbf{A}\mathbf{x}) + \lambda \mathcal{Q}_x^h(\vec{\kappa}) \tag{3.19}$$

where, in the continuous form, the smoothing portion of the functional $\lambda \mathcal{Q}_x^h(\vec{\kappa})$ restricts the spatial frequencies of the power spectrum, because its inclusion in the functional will yield as a solution the one that minimizes the difference between the autocorrelation of the solution

and the power spectrum filtered by a low-pass rectangular filter in all three dimensions. That is, we want to minimize the portion of the spatial spectrum of the solution that falls out of the low pass filter represented by $B_{\vec{\kappa}}(\vec{\omega})$, where $\vec{\omega}$ represents the 3D spatial frequency. Therefore we write:

$$\mathcal{Q}_x^h(\vec{\kappa}) = \int S_{xx}(\vec{\omega})\left[1 - B_{\vec{\kappa}}(\vec{\omega})\right] d^3\omega \tag{3.20}$$

which can be rewritten as:

$$\mathcal{Q}_x^h(\vec{\kappa}) = \mathcal{F}^{-1}\left(S_{xx}(\vec{\omega})\left[1 - B_{\vec{\kappa}}(\vec{\omega})\right]\right) \|_{\vec{\Delta}r=0} \tag{3.21}$$

$$= \mathcal{F}^{-1}\left(S_{xx}(\vec{\omega})\right) * \mathcal{F}^{-1}\left(1 - B_{\vec{\kappa}}(\vec{\omega})\right) \|_{\vec{\Delta}r=0} \tag{3.22}$$

$$= \left(R_{xx}(\vec{r}) * \left[\delta^3(\vec{r}) - b_{\vec{\kappa}}(\vec{r})\right]\right) \|_{\vec{\Delta}r=0} \tag{3.23}$$

where $*$ indicates convolution, $b_{\vec{\kappa}}(\vec{r})$ is the inverse Fourier transform of the 3D low-pass rectangular defined by $\vec{\kappa}$, and thus,

$$\mathcal{Q}_x^h(\vec{\kappa}) = \int R_{xx}(\vec{r})\left[\delta^3(\vec{r}) - b_{\vec{\kappa}}(\vec{r})\right] d^3r, \tag{3.24}$$

This can also be written as additional linear equations. In the generalization of the problem and its software implementation we have followed the linear equations form of the constraints.

3.1.2.3 Singular Value Decomposition

In order to obtain the inverse matrix shown in Equation 3.17 we use the Singular Value Decomposition (SVD), as is described, e.g., in [42], [6], and [44], and the numerical implementation described in [46]. We give here only a short review of the SVD.

The SVD is based on the following theorem (we follow [46] because we use the implementation described there; an equivalent formulation is found in [42] and [45]):

An MxN matrix $\mathbf{S}$ with M≥N can be written as:

$$\mathbf{S} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T \tag{3.25}$$

where $\mathbf{U}$ is an MxN column-orthogonal matrix, $\mathbf{\Lambda}$ is an NxN diagonal matrix of non-negative values (the *singular values*) and $\mathbf{V}$ is an NxN orthogonal matrix. The singular values are uniquely determined by $\mathbf{S}$, although $\mathbf{U}$ and $\mathbf{V}$ are not.

Given a singular value decomposition $\mathbf{S} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T$ then the eigenvalue decomposition of $\mathbf{S}^T\mathbf{S}$, is given by:

$$\mathbf{S}^T\mathbf{S} = \mathbf{V}\left(\mathbf{\Lambda}^T\mathbf{\Lambda}\right)\mathbf{V}^T \tag{3.26}$$

where $\mathbf{\Lambda}^T\mathbf{\Lambda}$ is a diagonal matrix with the eigenvalues of $\mathbf{S}^T\mathbf{S}$.

The generalized inverse (see [42], [45]) is formed using the non-zero singular values:

$$\mathbf{S}^{-g} = \mathbf{V}\mathbf{\Lambda}^{-1}\mathbf{U}^T \tag{3.27}$$

where in $\mathbf{\Lambda^{-1}}$ the corresponding values of zero singular values are set to zero and not to infinity.

Following Equation 3.17, we need to invert a symmetric matrix and we will therefore be using the eigenvalues of the $\mathbf{S}^T\left(\mathbf{C}_{\hat{y}}\right)^{-1}\mathbf{S}$. In other words,

$$\left(\mathbf{S}^T\left(\mathbf{C}_{\hat{y}}\right)^{-1}\mathbf{S}\right)^{-g} = \mathbf{V_s}\mathbf{\Lambda_s}^{-1}\mathbf{V_s}^T \tag{3.28}$$

where $\mathbf{V_s}$ and $\mathbf{\Lambda_s}$ are the matrixes of eigenvectors and eigenvalues of the matrix $\mathbf{S}^T\left(\mathbf{C}_{\hat{y}}\right)^{-1}\mathbf{S}$.

The zero eigenvalues values correspond to model parameters (voxels) not fixed by the data. If we want a solution to be well defined, then there should not be any zero eigenvalues. Furthermore, small eigenvalues, not only the non-zero values due to computational round-off errors, actually produce a large variance in the solution, since the covariance matrix of the solution is [42]

$$\mathbf{C}_x = \sigma_d^2\mathbf{V_s}\mathbf{\Lambda_s}^{-1}\mathbf{V_s}^T \tag{3.29}$$

$$\sigma_d^2 = \frac{(\hat{\mathbf{y}} - \mathbf{S}\mathbf{x}^{est})^T\left(\mathbf{C}_{\hat{y}}\right)^{-1}(\hat{\mathbf{y}} - \mathbf{S}\mathbf{x}^{est})}{N_{obs} - N_{unk}}, \tag{3.30}$$

where σ_d^2 is the parameters a posteriori variance scaling factor needed to adjust the covariances to consistent scales, since they will be combined with the transition covariance matrixes of the Kalman filter. In Equation 3.17 the solution is not affected by the scaling factors present in $\mathbf{C}_{\hat{y}}$ ([47]).

As discussed in [6] and [48], there can be an amplification of the input noise into the solution by a factor given by the smallest non-zero eigenvalue of $\mathbf{S}^T\left(\mathbf{C}_{\hat{y}}\right)^{-1}\mathbf{S}$. We can then consider that $\sigma_x \sim \sigma_y/\nu$ where ν is the smallest non-zero singular value of $\mathbf{S}$. This sets a cut-off value for the values in $\mathbf{\Lambda_s}$. Introducing smoothing constraints increases the value of the zero singular values, due to the inherent enlargement of the effective size of those voxels where there are not sufficient independent observations. We therefore must set the weight of the constraints (λ in Equation 3.17) to the minimum value that makes all the singular values be above a cut-off level. This is the trade-off between variance and resolution [42]: the higher the constraints, the more coarse is the resolution but the smaller the variance is.

The value of ν is, in the context of tomography, the minimum length that a ray has to cross to provide information above the noise level. This is an *effective* length, since N rays crossing the same voxels will carry information of the same region, but, when combining them, the noise will be reduced by a factor of $\sqrt{N}$. Thus the minimum length is $\nu/\sqrt{N}$

3.2 Kalman filter

3.2.1 Forward filter

The dynamics of the solution are accounted for in the implementation of a Kalman filtering. We follow the form described in [47] and [49].

Given the solution to our system at time t,

$$\hat{\mathbf{y}}_t = \mathbf{S}_t \mathbf{x}_t, \tag{3.31}$$

we can predict the solution at time $t+1$ through a state transition matrix F_t

$$\mathbf{x}_{t+1} = \mathbf{F}_t \mathbf{x}_t + \mathbf{w}_t, \tag{3.32}$$

where $\mathbf{w}_t$ is the vector of the random perturbations in the transition from one state to the following. Let us assume that the noise (in the measurements and in the transitions) has zero mean and that measurements and dynamics of the system are uncorrelated, as well as the perturbations at any epoch and them with previous states. We can now form the Kalman filter by sequentially predicting the state at epoch $t+1$ and then update it with measurements. The sequence is as follows (same notation as in [49] but following the implementation of [47], as we shall discuss later): first we can predict the new solution and its covariance matrix by

$$\hat{\mathbf{x}}_{t+1}^{t} = \mathbf{F}_t \hat{\mathbf{x}}_t^t, \tag{3.33}$$

$$\mathbf{C}_{t+1}^{t} = \mathbf{F}_t \mathbf{C}_t^t \mathbf{F}_t^T + \mathbf{W}_t \tag{3.34}$$

and then this information is added to the set of equations including measurements and the complete system is solved through a least-squares procedure:

$$\hat{\mathbf{x}}_{t+1}^{t+1} = \left[\mathbf{S}_{t+1}^T \left(\mathbf{C}_{\hat{y}}^{t+1}\right)^{-1} \mathbf{S}_{t+1} + \left(\mathbf{C}_{t+1}^{t}\right)^{-1}\right]^{-1} \left[\mathbf{S}_{t+1}^T \left(\mathbf{C}_{\hat{y}}^{t+1}\right)^{-1} \hat{y}_{t+1} + \left(\mathbf{C}_{t+1}^{t}\right)^{-1} \mathbf{F}_t x_t\right], \tag{3.35}$$

$$\mathbf{C}_{t+1}^{t+1} = \left[\mathbf{S}_{t+1}^T \left(\mathbf{C}_{\hat{y}}^{t+1}\right)^{-1} \mathbf{S}_{t+1} + \left(\mathbf{C}_{t+1}^{t}\right)^{-1}\right]^{-1}. \tag{3.36}$$

This formulation is chosen because, as it is discussed in [47], it has no round-off problems that may lead to non-convergence of the filter. The drawback is that it requires the matrix to be inverted to have the dimension of the parameters space. Another possible formulation inverts the matrix in the observables space. If we were to perform an update at each measurement epoch, then the matrix might have a much smaller size in the latter formulation. However, this is not the case given that we apply equations of constraints at each voxel and treat them as observations; therefore, the observables space dimension is always greater than the parameters space dimension.

Finally, we consider our system to behave as a random walk stochastic process. This is characterized by the following expressions of the state transition matrix and the covariance of the perturbations:

$$\mathbf{F}_t = \mathbf{I}, \tag{3.37}$$

$$\mathbf{W}_t = |t_{t+1} - t_t|\delta^{\mathbf{2}} \tag{3.38}$$

where $\delta^{\mathbf{2}}$ is a diagonal matrix that represents the power spectrum density of the process: each element represents how much we allow a variable to change per unit of time. This value has to be consistent with the magnitude of the solution covariance matrix.

3.2.2 Smoothing process

The sequence so far described refers to the forward Kalman filter. At the end of the process, when all the observations have been added, the estimation of the non-stochastic parameters contains all the information but, on the other hand, only the last estimation of the stochastic parameters contains it. Therefore, we need to run a *smoothing* or *backward filter*. We have implemented the smoothing solution of [49]. This algorithm runs the Kalman filter forward and backward and then takes the weighted mean at each epoch considering the full covariance matrixes. First, the forward filter is run and the solutions stored. When running backwards, the solution is found for time $t+1$ and then the prediction for time t is formed. Then, this prediction is combined with the solution at time t of the forward filter as follows:

$$\mathbf{B} = \mathbf{C}_+ \left(\mathbf{C}_- + \mathbf{C}_+\right)^{-1} \tag{3.39}$$

$$\hat{\mathbf{x}}_t^s = \mathbf{x}_+ + \mathbf{B}\left(\mathbf{x}_- - \mathbf{x}_+\right) \tag{3.40}$$

$$\mathbf{C}_t^s = \mathbf{C}_+ - \mathbf{B}\mathbf{C}_+ \tag{3.41}$$

where the subscript $-$ stands for the backward filter ($\mathbf{C}_-$ is the covariance matrix of the prediction at time t in the backward filter and $\mathbf{x}_-$ is the predicted solution) and $+$ for the forward filter ($\mathbf{C}_+$ is the covariance matrix of the solution at time t including measurements, and $\mathbf{x}_+$ is the solution itself).

3.3 A probabilistic view of the inverse problem

We have stated before that the key question in solving the inverse problem is how to fill the information gap due to the data being sparse and correlated. In this section we want to provide a qualitative description of the inverse problem in order to understand the different applications and why the different experiments analyzed needed particular approaches.

Probability theory offers a valuable tool for an intuitive interpretation of the state of information. In fact, as [41] postulates, the state of information on a parameter space $\mathbf{X}$ can be in general be described by the probability over $\mathbf{X}$. We will consider in this discussion that the probability density function (PDF) *represents* the state of information about a random variable. As an example one can think of having a Dirac's delta function as PDF when we have Total Knowledge about the value of a variable and a uniform, constant PDF when we don't know anything about its value (State of Ignorance).

Further, given the PDF $f(\mathbf{x}, \mathbf{y})$, representing the state of information on two parameters $\mathbf{x}$ and $\mathbf{y}$, the marginal probability functions for each parameter contain all the information on that parameter, only loosing the correlation with the other. Given these interpretations of PDF's we will here give a qualitative discussion on the impact of the constraints in the solution of an inverse problem. This discussion emerges from [41] and [42] and we will follow the notation of the former.

Let us consider that we have our set of data $\mathbf{d}$ and that we have some knowledge about their real value, represented by the PDF $\rho_D(\mathbf{d})$ (which is not a delta function due to uncertainties and noise in the instrumentation). We can also put some constraints or apriori information on the parameters $\mathbf{m}$ and we will represent this information with the PDF $\rho_M(\mathbf{m})$; if we don't know anything about $\mathbf{m}$, then $\rho_M(\mathbf{m})$ will be the State of Ignorance. Now, by combining these two informations, we must be able to solve the inverse problem. This combination is described by the joint PDF:

$$\rho(\mathbf{d}, \mathbf{m}) = \rho_D(\mathbf{d})\rho_M(\mathbf{m}) \tag{3.42}$$

since, by definition, both informations have been obtained independently. Now, we need some theory that relates observations with parameters. This theory may also have some uncertainty and therefore, we will represent it by a PDF, $\Theta(\mathbf{d}, \mathbf{m})$, that contains the correlations between both variables. If the theory is exact, this PDF represents a surface in the $(\mathbf{d}, \mathbf{m})$ space in which our solution to the inverse problem has to exist. Now, when all these states of information are combined, we obtain the *a posteriori* information $\sigma(\mathbf{d}, \mathbf{m})$ given by (as was postulated in [50]):

$$\sigma(\mathbf{d}, \mathbf{m}) = \frac{\rho(\mathbf{d}, \mathbf{m})\Theta(\mathbf{d}, \mathbf{m})}{\mu(\mathbf{d}, \mathbf{m})} \tag{3.43}$$

where $\mu(\mathbf{d}, \mathbf{m})$ represents the State of Ignorance or non-informative probability density. In the inverse problem, we consider that the theoretical relationship between parameters and observations can be described by a conditional probability density of $\mathbf{d}$ given $\mathbf{m}$:

$$\Theta(\mathbf{d}, \mathbf{m}) = \Theta(\mathbf{d}|\mathbf{m})\mu_M(\mathbf{m}) \tag{3.44}$$

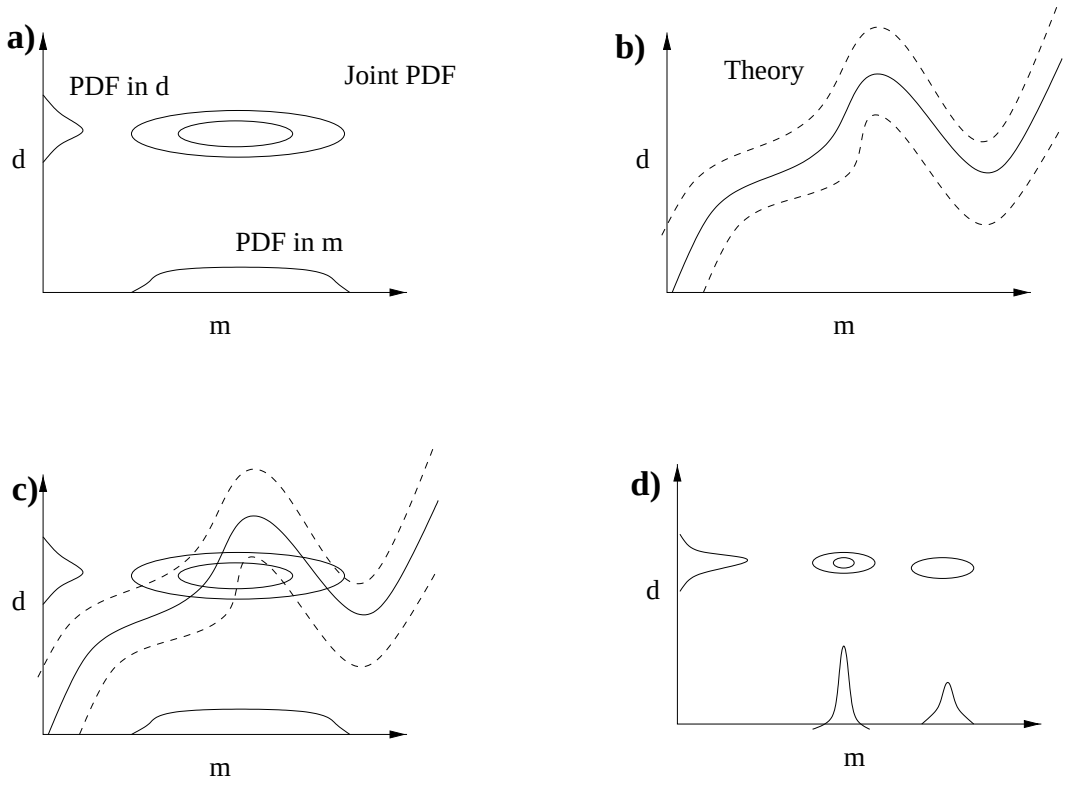

Figure 3.2: Representation of the inverse problem solution through PDF and states of information. In the a) we show the PDF of the data, the model parameters and the joint PDF. In b) the theory relates data with parameters. In c) we combine all the information and in d) the solution is depicted. Note that the marginal PDF of $\mathbf{m}$ shows that there exist more than one state compatible with the data and the theory.

where $\mu_M(\mathbf{m})$ is the non-informative probability density for model parameters, and the solution of the inverse problem is the information on $\mathbf{m}$ that we have in the a posteriori information; that is, the solution is the *marginal* PDF of the a posteriori PDF ([41]):

$$\sigma_M(\mathbf{m}) = \int_D \sigma(\mathbf{d}, \mathbf{m})\, d\mathbf{d} \tag{3.45}$$

The present discussion allows us to have a pictorial representation of different situations, thus providing an intuitive interpretation. In Figure 3.2 we show one of such interpretations: the data have a narrow distribution, while the a priori information on the parameters is relaxed, only limiting the solutions to a certain domain. The joint probability function defines the region of compatibility of both states of information. The theory relates both variables; the solution of the inverse problem is the region in the $(\mathbf{d}, \mathbf{m})$ space that makes the states of $\mathbf{d}$ and $\mathbf{m}$ *through* the theory compatible.

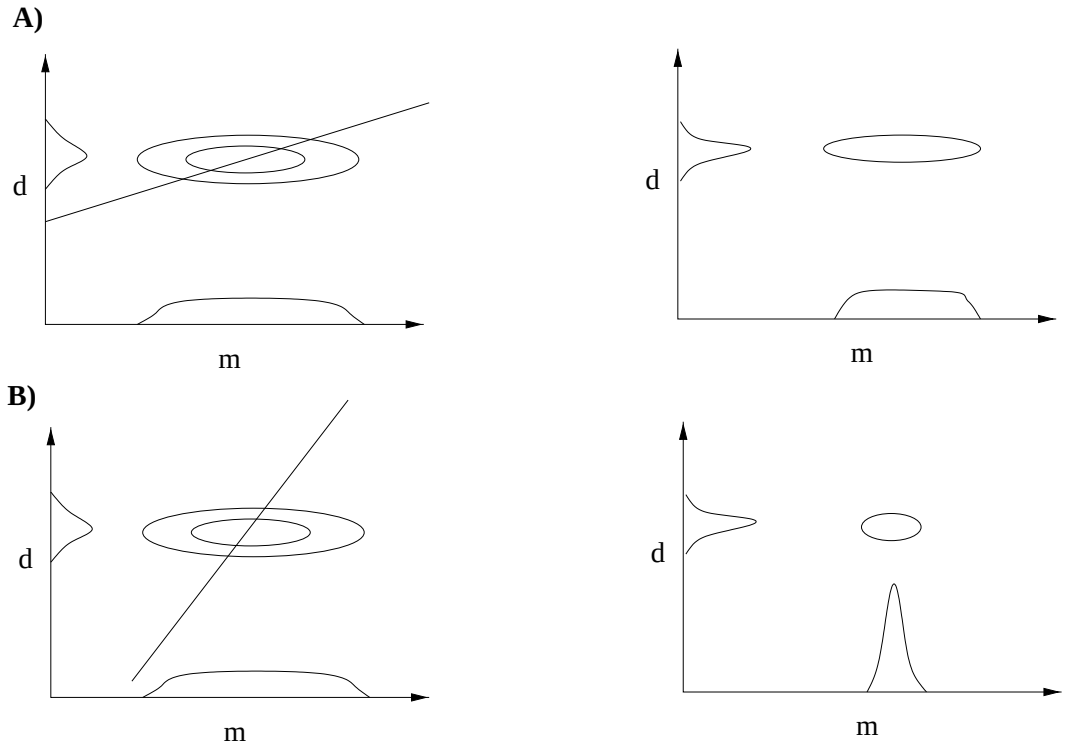

Figure 3.3: Representations of two different situations: A) the eigenvalues are small and the linear functions has a small slope; the solution has no distinct peak in the distribution; B) the eigenvalues are bigger and the solution has a spike-like shape.

3.3.1 Particularization to Tomography

As we have seen, our approach to tomography considers a linear system relating data with parameters, and the uncertainty of such theory is only expressed as the existence of a discretization error, which we will quantify later. Therefore, $\Theta(\mathbf{d}, \mathbf{m})$ will be a linear function that can be viewed in the 2D representation as a piecewise linear function where the slopes are given by the eigenvalues (as a rule-of-thumb): the smaller they are, the more flat is the straight line and the wider the possible solutions for $\mathbf{m}$ (see Figure 3.3).

Data are considered to be uncorrelated and with gaussian distributions. The a priori information is considered as linear equations that are treated as additional observations, also with an uncorrelated gaussian distribution. The solution is obtained by minimizing the quadratic error, which corresponds to the maximum likelihood solution, i.e. the point $\mathbf{m}_{ML}$ that maximizes $\sigma_M(\mathbf{m})$.

We will see in the chapters devoted to ionosphere and troposphere that the weight of the constraints need to be tuned for each case. These cases have geometries (ray distributions across the grid) that actually change the $\Theta(\mathbf{d}, \mathbf{m})$ function, changing its slope and the number of crossings with the joint PDF $\rho(\mathbf{d}, \mathbf{m})$, and hence, the shape of the marginal a posteriori

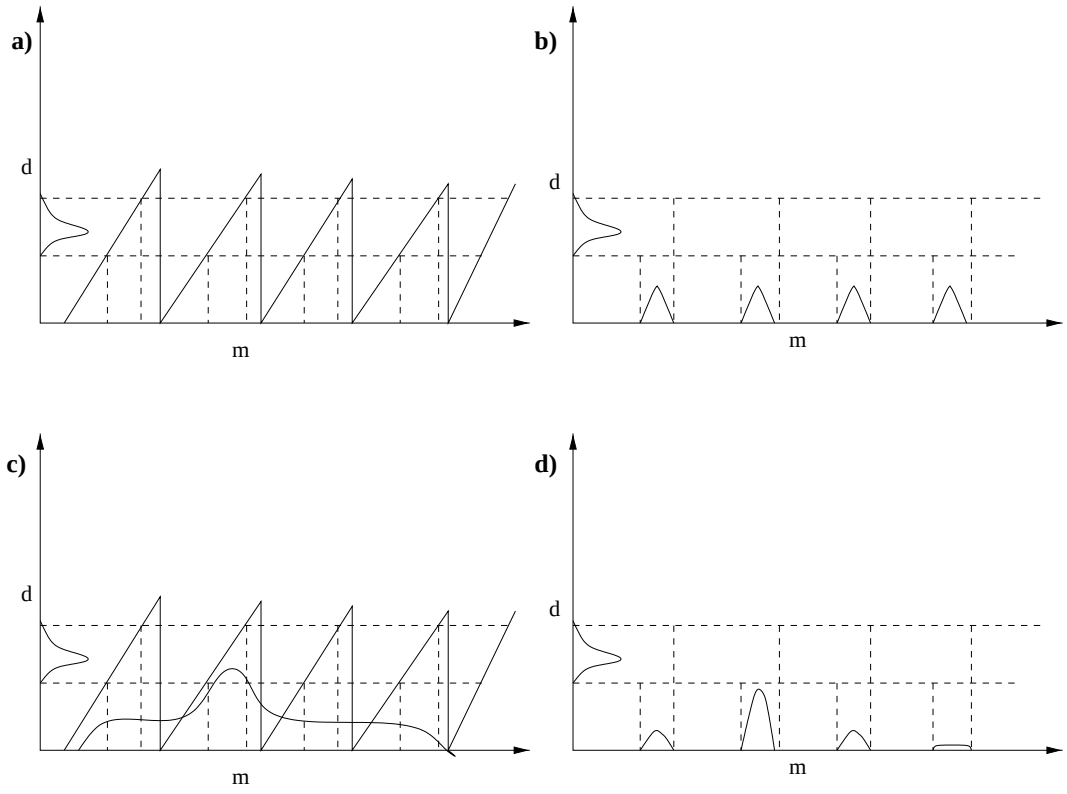

Figure 3.4: Representation of the saw (linear) function. The linear function relates the data with the model parameters but the existence of zero eigenvalues produces the data to be consistent with different sets of parameters (a and b). Some a priori information is needed in order to select one (c and d)

PDF in $\mathbf{m}$ (note that our goal is to obtain a narrow spike-like σ_M function, so that an accurate determination of the maximum likelihood point is possible).

Let us now consider a case where, being $\Theta(\mathbf{d}, \mathbf{m})$ a linear function, it does not uniquely characterize the system. This can be represented using a saw function as shown in Figure 3.4. When applying the data, there will be many possible parametrizations of the model compatible with them. We thus have to apply some a priori information. Because this information is included as additional equations, it can be viewed either as changing the slope and location of $\Theta(\mathbf{d}, \mathbf{m})$ and the data, or as modifying the shape of $\rho_M(\mathbf{m})$. The latter may be more appropriate. Still considering the saw function, we can then increase the PDF around those $\mathbf{m}$ points that comply with a smoothing solution or with a boundary condition (in the tropospheric case it will be shown that it is needed to impose a zero-value condition for the wet refractivity at high altitudes). This produces the solution to be moved closer to that area. Depending on how well the data complies with the a priori information, the latter will require to be weighted more.

The probabilistic approach is not substantially different from the one described in previous

sections but provides a conceptual representation that can be useful when understanding the addition of different constraints to the solution.

3.4 The GIST/LOTTOS Software package

The complete tomographic process has been implemented in a combined software package called Global Ionospheric Stochastic Tomography/LOcal Tropospheric TOmography Software (GIST/LOTTOS) which handles both the application to the ionosphere and to the troposphere. Most of the modules are common to both processes though the preprocessing stage is different. The software is object-oriented based, coded in Fortran 90 and the interface with the user is done under through c-shell scripts. In addition, LOTTOS has some simulation capabilities which makes use of modules contained in the GIPSY-OASIS II software package ([33]). Results presented here have been obtained using GIST/LOTTOS. In [51] there is a detailed explanation of the structure and the different objects defined. The software has been designed to be portable and it can now run under different computer architectures (Solaris-sparc, Linux-alpha, Linux-i386 and HP9000).

The flow chart of the software is presented in Figure 3.5. There are 6 main modules:

- **Data Preprocessor**:
  - **GIST**: The DP consist of a program called **rnx2table** that is able to read files in RINEX format ([52]) and splits the data into phase connected arcs, detecting cycle slips and aligning the phase to the pseudorange yielding the carrier-phase smoothed pseudorange. The output is a table containing time-tagged ionospheric delays with the corresponding cartesian coordinates of the receiver and the transmitter in an Earth Centered Inertial reference frame.
  - **LOTTOS**: The DP module consists of a set of routines that read the GOA II output and combine them to form slant wet delays (described in Section 5.1.1).
- **Gridder**: computes the grid limits.
- **Tomographer**: this module reads the data file in the tabular form and generates the $\mathbf{A}$ matrix described above.
- **Prep_Kalman**: computes the matrixes $\mathbf{A}^T (\mathbf{C}_y)^{-1} \mathbf{A}$ for each batch in which the Kalman filter is split.
- **Prep_Constraints**: computes the matrixes $\mathbf{B}^T\mathbf{B}$ containing the equations of constraint.

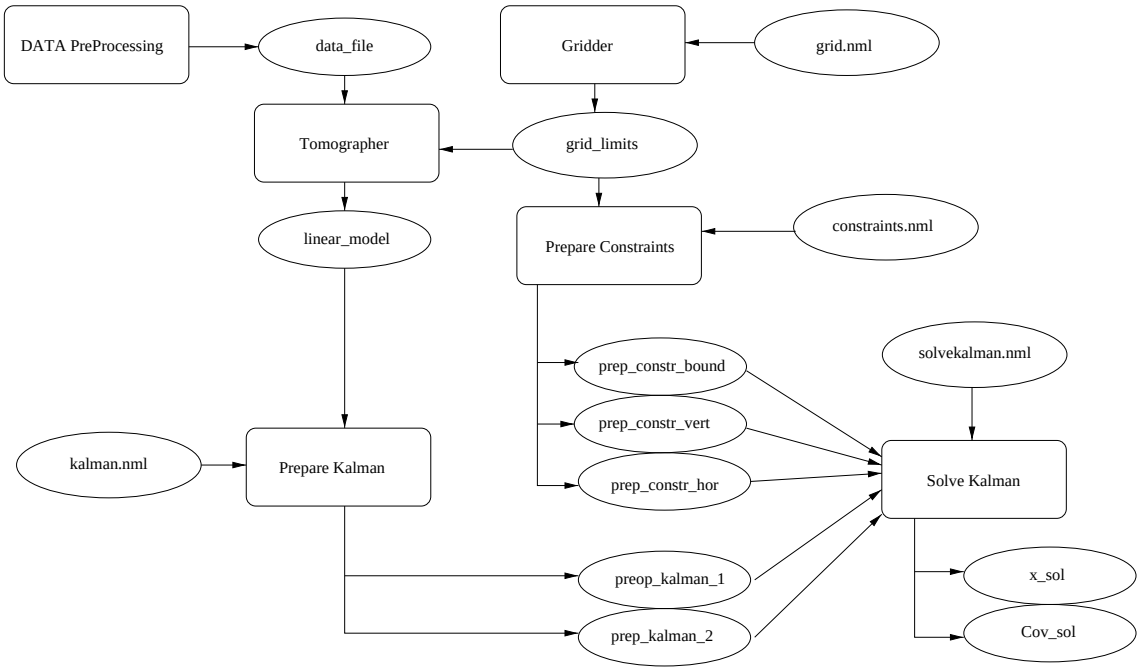

Figure 3.5: Flow chart of the LOTTOS/GIST software package.

- **Solve_Kalman**: Implements the forward Kalman filter and the smoothing process.

There are other submodules to read the different files and some scripts to prepare the namelists to drive the programs. The complete processing is executed under the *make* utility so that different runs with different parameters can be efficiently executed. In terms of the coding itself, the focus of the software development has been on robustness.