3 Maximum a Posteriori Solution

We will now introduce a fundamental approximation used in most practical data-assimilation methods, namely the definition of Gaussian priors. This approximation simplifies the Bayesian posterior, which allows us to compute the maximum a posteriori (MAP) estimate and sample from the posterior pdf. This chapter will introduce the Gaussian approximation and then discuss the Gauss–Newton method for finding the MAP estimate. This method is the starting point for many of the data-assimilation algorithms discussed in the following chapters.

3.1 Maximum a Posteriori (MAP) Estimate

The MAP solution is the state vector **z** that maximizes the posterior pdf, and can thus be seen as the most probable solution for **z** given the measurements **d**. We define it as

$$\mathbf{z}_{\text{MAP}} = \underset{\mathbf{z}}{\operatorname{argmax}}\big(f(\mathbf{z}|\mathbf{d})\big). \tag{3.1}$$

The variable **z** is called the *control variable* or *control vector* in the inverse modeling and control literature. Since we can write any smooth posterior pdf as

$$f(\mathbf{z}|\mathbf{d}) \propto \exp\big\{-\mathcal{J}(\mathbf{z})\big\}, \tag{3.2}$$

and the logarithm is a monotonically increasing function of its argument, the vector that maximizes the posterior pdf equals the vector that minimizes the *cost function* $\mathcal{J}(\mathbf{z})$. Hence, we can write

$$\mathbf{z}_{\text{MAP}} = \underset{\mathbf{z}}{\operatorname{argmin}}\,\mathcal{J}(\mathbf{z}). \tag{3.3}$$

We can find a function's minimum by setting its gradient equal to zero. So, at the minimum, we have

$$\nabla_z\mathcal{J}(\mathbf{z}_{\text{MAP}}) = 0. \tag{3.4}$$

G. Evensen et al., *Data Assimilation Fundamentals*,
Springer Textbooks in Earth Sciences, Geography and Environment,
https://doi.org/10.1007/978-3-030-96709-3_3

Furthermore, the second derivative of the cost function, the so-called *Hessian*, has information on the cost function's curvature at the minimum. As we will see, the inverse of that Hessian provides a first-order estimate of the posterior covariance.

In most geoscience applications of data assimilation that compute the MAP estimate, one assumes that both the prior and observation errors are Gaussian, leading to a more tractable problem. We will explore such methods in the following sections, followed by the explicit solutions for linear problems and an extensive treatment of iterative methods for nonlinear problems.

3.2 Gaussian Prior and Likelihood

Many popular data-assimilation methods assume that the prior distributions are Gaussian, leading to a simple representation of the data-assimilation problem. Note that the cost function is not quadratic in $\mathbf{z}$ as the measurement operator is still nonlinear. Hence, we introduce the following approximation.

Approximation 4 (Gaussian prior and likelihood) *We assume that the prior distributions of the state vector's components* $\mathbf{z}$ *and observation errors* $\boldsymbol{\epsilon}$ *are both Gaussian distributed.* □

We will in Chap. 9 discuss methods that do not apply Approx. 4. Now, we define

$$f(\mathbf{z}) = \mathcal{N}\big(\mathbf{z}^{\mathrm{f}}, \mathbf{C}_{zz}\big), \tag{3.5}$$

$$f\big(\mathbf{d} \,|\, \mathbf{g}(\mathbf{z})\big) = f(\boldsymbol{\epsilon}) = \mathcal{N}\big(\mathbf{0}, \mathbf{C}_{dd}\big), \tag{3.6}$$

where the superscript f denote "first guess." Thus, $\mathbf{z}^{\mathrm{f}}$ is the "first guess" or *prior* estimate of the state vector, and $\mathbf{C}_{zz}$ is its error covariance. The prior error covariance includes the covariances between all the uncertain variables in the state vector,

$$\mathbf{C}_{zz} = \begin{pmatrix} \mathbf{C}_{x_0x_0} & \mathbf{C}_{x_0\theta} & \mathbf{C}_{x_0u} & \mathbf{C}_{x_0q} \\ \mathbf{C}_{\theta x_0} & \mathbf{C}_{\theta\theta} & \mathbf{C}_{\theta u} & \mathbf{C}_{\theta q} \\ \mathbf{C}_{ux_0} & \mathbf{C}_{u\theta} & \mathbf{C}_{uu} & \mathbf{C}_{uq} \\ \mathbf{C}_{qx_0} & \mathbf{C}_{q\theta} & \mathbf{C}_{qu} & \mathbf{C}_{qq} \end{pmatrix}. \tag{3.7}$$

Note that we formulate $\mathbf{z}$ such that it contains the model state at time zero and the model errors at other times, the so-called *forcing formulation*. In this case, the Gaussian prior assumption is reasonable and often assumed. However, if we reformulate the problem so that $\mathbf{z}$ contains the model solution, there is no model error in $\mathbf{z}$. A Gaussian prior for $\mathbf{z}$ would then force us to assume that the model is linear since only a linear model initialized with a Gaussian initial state would yield a model state that remains Gaussian over a time window. For this reason, we use the *forcing formulation* here. In most data-assimilation problems, we would neglect the covariances between different variables and retain only the covariance matrices on the diagonal. However, for the derivation of the methods below, we do not need to make this assumption.

The introduction of Gaussian priors leads to a posterior pdf formulation that we use in Bayes theorem to find

$$f(\mathbf{z}|\mathbf{d}) \propto \exp\big\{-\mathcal{J}(\mathbf{z})\big\}, \tag{3.8}$$

with the cost function $\mathcal{J}(\mathbf{z})$ defined as

Cost function

$$\mathcal{J}(\mathbf{z}) = \frac{1}{2}\big(\mathbf{z}-\mathbf{z}^{\mathrm{f}}\big)^{\mathrm{T}}\mathbf{C}_{zz}^{-1}\big(\mathbf{z}-\mathbf{z}^{\mathrm{f}}\big) + \frac{1}{2}\big(\mathbf{g}(\mathbf{z})-\mathbf{d}\big)^{\mathrm{T}}\mathbf{C}_{dd}^{-1}\big(\mathbf{g}(\mathbf{z})-\mathbf{d}\big). \tag{3.9}$$

Note that $\mathbf{g}(\mathbf{z})$ is the nonlinear mapping from the state vector, i.e., initial conditions, model errors, and parameters, to the predicted measurements. Thus, we have used the Bayesian formulation from Eq. (2.43). As mentioned, minimizing $\mathcal{J}(\mathbf{z})$ in Eq. (3.9) is equivalent to maximizing the *a posteriori* probability (MAP) solution of the posterior pdf in Eq. (3.8) with Approx. 4 on the Gaussian priors.

To find the MAP solution, we start with the cost function's gradient

$$\nabla_{\mathbf{z}}\mathcal{J}(\mathbf{z}) = \mathbf{C}_{zz}^{-1}\big(\mathbf{z}-\mathbf{z}^{\mathrm{f}}\big) + \nabla_{\mathbf{z}}\mathbf{g}(\mathbf{z})\,\mathbf{C}_{dd}^{-1}\,\big(\mathbf{g}(\mathbf{z})-\mathbf{d}\big), \tag{3.10}$$

and by setting it to zero we define the minimizing solution $\mathbf{z}^{\mathrm{a}}$ from

The gradient set to zero

$$\mathbf{C}_{zz}^{-1}\big(\mathbf{z}^{\mathrm{a}}-\mathbf{z}^{\mathrm{f}}\big) + \nabla_{\mathbf{z}}\mathbf{g}\big(\mathbf{z}^{\mathrm{a}}\big)\,\mathbf{C}_{dd}^{-1}\,\Big(\mathbf{g}\big(\mathbf{z}^{\mathrm{a}}\big)-\mathbf{d}\Big) = 0. \tag{3.11}$$

Here the superscript a denote "analysis." This equation forms the implicit, closed-form solution of our estimation problem that minimizes the cost function in Eq. (3.9). The model sensitivity $\nabla_{\mathbf{z}}\mathbf{g}\big(\mathbf{z}^{\mathrm{a}}\big)$ is the gradient of the predicted measurements to the state vector. The following sections will present iterative methods that solve Eq. (3.11), leading to various 4DVar formulations.

3.3 Iterative Solutions

Even if $\mathbf{g}$ is a linear function of its argument and we can write down the explicit solution to Eq. (3.11), it is not uncommon to solve the problem iteratively. The reason is that the matrices involved can be of very high dimension and impossible to store in a computer. Another more practical reason is to avoid inverting matrices.

An important iterative minimization method is the so-called *Newton method*, which we can derive from a second-order Taylor expansion of the cost function. If we have an estimate of the minimum $\mathbf{z}^i$, we can improve this estimate by minimizing the expression

$$\mathcal{J}\big(\mathbf{z}^i+\delta\mathbf{z}\big) \approx \mathcal{J}\big(\mathbf{z}^i\big) + \delta\mathbf{z}^{\mathrm{T}}\,\nabla_{\mathbf{z}}\mathcal{J}^i + \frac{1}{2}\delta\mathbf{z}^{\mathrm{T}}\,\nabla_{\mathbf{z}}\nabla_{\mathbf{z}}\mathcal{J}^i\,\delta\mathbf{z}, \tag{3.12}$$

for $\delta\mathbf{z}$. Here $\nabla_{\mathbf{z}}\mathcal{J}^i$ denotes the cost function's gradient evaluated at $\mathbf{z}^i$, and $\nabla_{\mathbf{z}}\nabla_{\mathbf{z}}\mathcal{J}^i$ is the cost function's *Hessian* where we use the gradient operator twice. We readily find the solution for $\delta\mathbf{z}$ as

$$\frac{1}{2}\nabla_{\mathbf{z}}\nabla_{\mathbf{z}}\mathcal{J}^i\,\delta\mathbf{z} = -\nabla_{\mathbf{z}}\mathcal{J}^i. \tag{3.13}$$

The solution to this problem leads to a new estimate of the cost function's minimum, $\mathbf{z}^{i+1} = \mathbf{z}^i + \delta\mathbf{z}$, and we repeat the process with a second-order Taylor expansion around $\mathbf{z}^{i+1}$.

As mentioned above, although we could multiply this equation by the inverse of the Hessian to directly find the solution for $\delta\mathbf{z}$, this is often not the way this equation is solved. The reason is that the covariance matrices are often so large that they cannot be stored, not even on the world's most giant supercomputers. Instead, we use operators that return the matrix-vector products with these matrices. A beautiful example of this procedure is the so-called variational methods such as 4DVar, which replaces matrix-vector products with adjoint and forward model integrations. This procedure effectively leads to an iterative solution of the form

$$\mathbf{z}^{i+1} = \mathbf{z}^i - \gamma^i\,\mathbf{B}^i\,\nabla_{\mathbf{z}}\mathcal{J}\big(\mathbf{z}^i\big). \tag{3.14}$$

In Eq. (3.14), i is the iteration index, γ^i is a scalar that determines the so-called step size, and $\mathbf{B}^i$ is a matrix we can choose, typically in operator form, as mentioned above. The simplest choice, $\mathbf{B}^i = \mathbf{I}$, leads to the so-called *steepest descent* method, where the new iterate is directly downhill of the previous iterate. In many geoscience applications of data assimilation, this approach turns out to be a poor choice with a low convergence rate because the cost function often has a very irregular shape in high-dimensional spaces.

As we have seen, in the Newton method, one would like to choose $\mathbf{B}^i$ as the inverse of the Hessian, and $\gamma^i = 1$. The advantage of this choice is that if the Hessian is positive definite, the convergence rate is quadratic, meaning that $|\mathbf{z}^{i+1} - \mathbf{z}^{\mathrm{a}}| = r|\mathbf{z}^i - \mathbf{z}^{\mathrm{a}}|^2$, where $\mathbf{z}^{\mathrm{a}}$ denotes the state that minimizes the cost function and r is a positive constant that depends on details of the Hessian.

Often the Hessian is not available, so it is common to use an approximate Hessian. For instance, the *Gauss–Newton method* discussed below ignores part of the Hessian to ensure that the matrix in front of $\delta\mathbf{z}$ is symmetric positive definite by construction. Other approaches may start with an approximation to the Hessian and make this approximation more accurate at each iteration by using new gradient information. These are so-called *quasi-Newton methods*, and a much-used alternative is the Broyden, Fletcher, Goldfarb, and Shanno (BFGS) method. Because these methods use information from the Hessian, their convergence rate is faster than methods that ignore that information, such as steepest descent, but still not quadratic as in the Newton method. We say their convergence is superlinear.

If the matrix in front of $\delta\mathbf{z}$ is symmetric and positive definite, we can use an extremely efficient method called *conjugate gradient*. It has the advantage that it only requires the computation of one matrix-vector product at each iteration. Furthermore, we do not need to store the matrix. We can often represent it by a code that takes a vector as input and gives the matrix times that vector as output.

The Newton method is used in 3DVar and 4DVar, as we will discuss in Chaps. 4, 5, and 6. Primarily used, however, is the Gauss–Newton method, leading to implementations such as incremental 4DVar, which explores the conjugate-gradient minimization method commonly used in numerical weather and ocean forecasting. Furthermore, this formalism has led to a general methodology that can effectively be solved in ensemble space, resulting in iterative ensemble smoothers used in reservoir-engineering applications, amongst others. We will discuss this method next.

3.4 Gauss–Newton Iterations

A popular choice for finding an iterative solution to the cost function is the so-called Gauss–Newton method (Lawless et al., 2005). The Gauss–Newton method is an approximate Newton method where we approximate the Hessian by ignoring the second-order derivative of the nonlinear measurement operator. Let's take a deeper look at this approximation. We can write the full Hessian of the cost function in Eq. (3.9) as

$$\nabla_{\mathbf{z}}\nabla_{\mathbf{z}}\mathcal{J} = \mathbf{C}_{zz}^{-1} + \nabla_{\mathbf{z}}\mathbf{g}(\mathbf{z})\,\mathbf{C}_{dd}^{-1}\left(\nabla_{\mathbf{z}}\mathbf{g}(\mathbf{z})\right)^{\mathrm{T}} + \nabla_{\mathbf{z}}\nabla_{\mathbf{z}}\mathbf{g}(\mathbf{z})\,\mathbf{C}_{dd}^{-1}\left(\mathbf{g}(\mathbf{z}) - \mathbf{d}\right). \tag{3.15}$$

The Gauss–Newton method ignores the last term, leading to

$$\nabla_{\mathbf{z}}\nabla_{\mathbf{z}}\mathcal{J}(\mathbf{z}) \approx \mathbf{C}_{zz}^{-1} + \nabla_{\mathbf{z}}\mathbf{g}(\mathbf{z})\,\mathbf{C}_{dd}^{-1}\left(\nabla_{\mathbf{z}}\mathbf{g}(\mathbf{z})\right)^{\mathrm{T}}. \tag{3.16}$$

We can now write a Gauss–Newton iteration similar to Eq. (3.14) as

Gauss–Newton iteration

$$\mathbf{z}^{i+1} = \mathbf{z}^i - \gamma^i\left(\mathbf{C}_{zz}^{-1} + \mathbf{G}^{i\,\mathrm{T}}\mathbf{C}_{dd}^{-1}\mathbf{G}^i\right)^{-1}\left(\mathbf{C}_{zz}^{-1}(\mathbf{z}^i - \mathbf{z}^{\mathrm{f}}) + \mathbf{G}^{i\,\mathrm{T}}\mathbf{C}_{dd}^{-1}\left(\mathbf{g}(\mathbf{z}^i) - \mathbf{d}\right)\right). \tag{3.17}$$

Here the increment is a steplength γ times the gradient normalized by $\left(\mathbf{C}_{zz}^{-1} + \mathbf{G}^{i\,\mathrm{T}}\mathbf{C}_{dd}^{-1}\mathbf{G}^i\right)$, the approximate Hessian. In correspondence with Eq. (3.14), we have chosen $\mathbf{B}^i = \left(\mathbf{C}_{zz}^{-1} + \mathbf{G}^{i\,\mathrm{T}}\mathbf{C}_{dd}^{-1}\mathbf{G}^i\right)^{-1}$. Furthermore, we have defined the gradient of $\mathbf{g}(\mathbf{z})$ at iteration i as

$$\mathbf{G}^{i\,\mathrm{T}} = \nabla_{\mathbf{z}}\mathbf{g}\left(\mathbf{z}^i\right). \tag{3.18}$$

We can interpret the operator $\mathbf{G}^i$ as the tangent-linear-model operator at iteration i, which provides the linear relation between the state vector and the observations. Likewise, we can interpret the operator $\mathbf{G}^{i\,\mathrm{T}}$ as the tangent-linear model's adjoint.

3.5 Incremental Form of Gauss–Newton Iterations

As mentioned earlier, the storage of the approximate Hessian would require substantial memory if we use the direct Gauss–Newton method for high-dimensional problems, and the Hessian's inversion can be rather expensive. We will present two solutions to solve this problem. In Chap 7, we will use Eq. (3.17) to develop the ensemble-random-maximum-likelihood (EnRML) method, which is commonly used in the petroleum industry.

An alternative is to write Eq. (3.17) with $\gamma^i = 1$ as

$$\left(\mathbf{C}_{zz}^{-1} + \mathbf{G}^{i\,\mathrm{T}}\mathbf{C}_{dd}^{-1}\mathbf{G}^i\right)\left(\mathbf{z}^{i+1} - \mathbf{z}^i\right) = -\left(\mathbf{C}_{zz}^{-1}(\mathbf{z}^i - \mathbf{z}^{\mathrm{f}}) + \mathbf{G}^{i\,\mathrm{T}}\mathbf{C}_{dd}^{-1}\left(\mathbf{g}(\mathbf{z}^i) - \mathbf{d}\right)\right). \quad (3.19)$$

When we define, as before,

$$\delta\mathbf{z} = \mathbf{z}^{i+1} - \mathbf{z}^i, \quad (3.20)$$

this equation also arises as the minimum of the following *quadratic* cost function for $\delta\mathbf{z}$

$$\begin{aligned}\mathcal{J}(\delta\mathbf{z}) = &\frac{1}{2}\left(\delta\mathbf{z} + \mathbf{z}^i - \mathbf{z}^f\right)^{\mathrm{T}}\mathbf{C}_{zz}^{-1}\left(\delta\mathbf{z} + \mathbf{z}^i - \mathbf{z}^f\right)\\ &+ \frac{1}{2}\left(\mathbf{G}^i\delta\mathbf{z} + \mathbf{g}(\mathbf{z}^i) - \mathbf{d}\right)^{\mathrm{T}}\mathbf{C}_{dd}^{-1}\left(\mathbf{G}^i\delta\mathbf{z} + \mathbf{g}(\mathbf{z}^i) - \mathbf{d}\right).\end{aligned} \quad (3.21)$$

This cost function linearizes the model and observation operators around the model trajectory for each Gauss–Newton iteration starting from the initial condition $\mathbf{z}^i$. Because $\delta\mathbf{z}$ is small, we can approximate $\mathbf{g}(\mathbf{z}^i + \delta\mathbf{z}) \approx \mathbf{g}(\mathbf{z}^i) + \mathbf{G}^i\delta\mathbf{z}$ in which $\mathbf{G}^i$ is the transpose of the gradient of $\mathbf{g}(\mathbf{z}^i)$ from Eq. (3.18). For convenience, we define the innovation vector

$$\boldsymbol{\eta}^i = \mathbf{d} - \mathbf{g}(\mathbf{z}^i), \quad (3.22)$$

and the residual

$$\boldsymbol{\xi}^i = \mathbf{z}^{\mathrm{f}} - \mathbf{z}^i. \quad (3.23)$$

With $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$, we can now write the cost function in Eq. (3.21) for the increments $\delta\mathbf{z}$ as

Quadratic cost function for the increments

$$\mathcal{J}(\delta\mathbf{z}) = \frac{1}{2}\left(\delta\mathbf{z} - \boldsymbol{\xi}^i\right)^{\mathrm{T}}\mathbf{C}_{zz}^{-1}\left(\delta\mathbf{z} - \boldsymbol{\xi}^i\right) + \frac{1}{2}\left(\mathbf{G}^i\delta\mathbf{z} - \boldsymbol{\eta}^i\right)^{\mathrm{T}}\mathbf{C}_{dd}^{-1}\left(\mathbf{G}^i\delta\mathbf{z} - \boldsymbol{\eta}^i\right). \quad (3.24)$$

The solution for the increments becomes, from Eq. (3.19),

$$\left(\mathbf{C}_{zz}^{-1} + \mathbf{G}^{i\,\mathrm{T}}\mathbf{C}_{dd}^{-1}\mathbf{G}^i\right)\delta\mathbf{z} = \mathbf{C}_{zz}^{-1}\boldsymbol{\xi}^i + \mathbf{G}^{i\,\mathrm{T}}\mathbf{C}_{dd}^{-1}\boldsymbol{\eta}^i. \quad (3.25)$$

We can solve this linear set of equations iteratively, and we usually implement the approximate Hessian as a set of operations working on the vector $\delta\mathbf{z}$. Quasi-Newton methods like BFGS and conjugate gradient are highly efficient for minimizing this cost function.

Thus, the incremental form of the Gauss–Newton method corresponds to an iterative scheme where we find the minimum of a quadratic cost function for $\delta\mathbf{z}$ in each iteration. After that, we update $\mathbf{z}^{i+1} = \mathbf{z}^i + \delta\mathbf{z}$ from (3.20), integrate the nonlinear model with the updated state vector, and recompute the variables $\boldsymbol{\eta}^i$ and $\boldsymbol{\xi}^i$ from Eqs. (3.22) and (3.23) before we solve the quadratic minimization problem again.

Gauss–Newton has a special status among minimization methods. It turns non-quadratic minimization problems into a sequence of quadratic minimization problems. We can solve each of these quadratic problems iteratively, leading to one iteration within another. We will explore this approach in the methods discussed in the following chapters.