Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control. James C. Spall

ISBN: 0-471-33052-3

CHAPTER 13

MODEL SELECTION AND STATISTICAL INFORMATION

All models are wrong; some are useful.

—George E. P. Box, Professor of Statistics and Professor of Industrial Engineering, University of Wisconsin

Mathematical modeling is a fundamental topic underlying many aspects of stochastic search and optimization. Up to now, however, we have largely suppressed important aspects of modeling. In fact, we will continue to suppress important aspects of modeling, as the subject is huge and this book is not!

The focus in this text has been on the configuration, theory, and mechanics of the procedures for determining the best values for some quantities when the *loss function and parameter definitions have been taken as a given.* The definition of the loss function is fundamentally connected to assumptions about the mathematical structure representing the system. Even when minimal assumptions about the model are made, *some* information must be available to even state the problem and define the parameters θ.

At some level, modeling is devoted to the broadest of issues in mathematical analysis: How can one best use data to learn about a system? Because many of the issues relevant to such a question are far afield of our focus on search algorithms, we address only a sliver of the available results. The limited—but important—results here on model selection and the Fisher information matrix connect to the basic focus of this book in at least three ways. First, choosing the model is a stochastic search and optimization process itself ("find the *best* model"), although it is a process of a different type than that for estimating θ. Second, the definitions of θ and $L(\theta)$ are intimately connected to the choice of model. Third, the Fisher information matrix, as studied here, provides a measure of accuracy for the estimate for θ and provides a mechanism for choosing inputs to enhance this accuracy.

Section 13.1 treats the fundamental tradeoff between the bias and variance in choosing a model form. This provides a structure for balancing the need to have a relatively simple model that is easy to interpret and the need to have a model sufficiently rich to capture all relevant linear or nonlinear effects. The bias–variance tradeoff in selecting a model is related to determining the

value for p in the search and optimization and for determining the analytical form of $L(\boldsymbol{\theta})$ and/or $\boldsymbol{g}(\boldsymbol{\theta})$. Section 13.2 focuses on cross-validation, one of the most popular and flexible means of realizing an optimal tradeoff between bias and variance. Cross-validation provides a mechanism for choosing between candidate model forms.

Section 13.3 discusses applications and the computation of the Fisher information matrix. Among other uses, the information matrix can be applied to construct uncertainty bounds (e.g., confidence intervals) for the estimates of $\boldsymbol{\theta}$. (Chapter 17 discusses an application of this matrix in the problem of optimal experimental design.) Section 13.3 also presents a Monte Carlo method for approximating the information matrix in complex estimation problems where an analytical derivation may be difficult or impossible. Finally, Section 13.4 offers some concluding remarks.

13.1 BIAS–VARIANCE TRADEOFF

> *Pluralitas non est ponenda sine necessitate. (Plurality should not be assumed without necessity.)*
>
> —The "Occam's razor" principle, William of Occam, English philosopher, 1285–1349 (approx.).

13.1.1 Bias and Variance as Contributors to Model Prediction Error

The bias–variance tradeoff is a fundamental principle in comparing the quality of different mathematical models. In most of this book, we have taken the dimension p and the form of loss function $L(\boldsymbol{\theta})$ and/or root-finding function $\boldsymbol{g}(\boldsymbol{\theta})$ as a given (although $L(\boldsymbol{\theta})$ and/or $\boldsymbol{g}(\boldsymbol{\theta})$ may not be directly available, corresponding to cases with noisy function measurements). We then described methods by which $\boldsymbol{\theta}$ can be estimated. This estimation will continue to be the focus of the book. A closely related issue, however, is determining the mathematical *form* (and associated dimension of $\boldsymbol{\theta}$) of the functions $L(\boldsymbol{\theta})$ or $\boldsymbol{g}(\boldsymbol{\theta})$ prior to estimating $\boldsymbol{\theta}$. In the linear or nonlinear regression context (Chapters 3, 5, 11, and 17), this is essentially tantamount to determining the form of an underlying model describing the input–output relationship of the system of interest. Parameter estimation (à la nonlinear regression) is also critical for simulation modeling (Chapters 14 and 15) to determine the internal coefficients of the simulation *prior* to application of the simulation for analyzing the system under study (the latter application is the focus of Chapters 14 and 15). A typical question might be: How many terms should be included in a polynomial approximation to an unknown function of interest?

We encountered in Section 5.2 a taste of the bias–variance tradeoff in the discussion of over- and under-fitting for neural networks. The bias–variance tradeoff also connects to the message in the *Occam's razor* principle, which

states, in essence, that one should seek simpler models over more complex models. Of course, to achieve optimal predictive power from a mathematical model, it is also necessary to include sufficient richness in the model to capture the essential characteristics of the process. Hence, one should not use a model that is *too* simple. ("Everything should be made as simple as possible, but not simpler."—Albert Einstein.) The bias–variance tradeoff provides a formal structure for interpreting the Occam's razor principle. The formal structure, however, does not lead directly to implementable algorithms for realizing an optimal tradeoff between the bias and the variance. That will await the next section, which considers into the cross-validation method for model selection.

Let z represent the scalar output for some system based on an input vector $\boldsymbol{x}$. Suppose, as in previous chapters, that we model this input–output process with a regression function $h(\boldsymbol{\theta}, \boldsymbol{x})$ and a noise term. In particular, the *model* for the actual output z is the right-hand side of

$$z \overset{\text{model}}{=} h(\boldsymbol{\theta}, \boldsymbol{x}) + v,$$

where v is a noise term that may or may not have mean zero. The model on the right-hand side above does not generally correspond to the actual mechanism for generating the *true* output z.

We are emphasizing this distinction by writing $\overset{\text{model}}{=}$ in place of the usual equal sign, indicating that true equality only holds in the idealized case where the model is a precise description of the process. Elsewhere in the literature and in this book, the standard equal sign is used instead of the modified form above, where the equality should be interpreted to mean "the left-hand side is *modeled* to equal the right-hand side" (analogous to an assignment operator in computer programming). While the distinction may seem pedantic, the reader should understand that the equality here does not have the same meaning as an equality such as $10 = 6 + 4$. As we saw in Chapters 3 and 5 (and will see again in the experimental design discussion of Chapter 17), the loss and gradient functions, $L(\boldsymbol{\theta})$ and $\boldsymbol{g}(\boldsymbol{\theta})$, in a regression context are directly related to $h(\boldsymbol{\theta}, \boldsymbol{x})$. So the choice of model directly affects the optimization process.

A natural measure of effectiveness of the regression function (with specific $\boldsymbol{\theta}$) as a predictor of z given the current value of $\boldsymbol{x}$ is the conditional mean-squared error (MSE), $E\left[(h(\boldsymbol{\theta}, \boldsymbol{x}) - z)^2 \,|\, \boldsymbol{x}\right]$, where the expectation is computed with respect to the random variable z.[1] Conditional on $\boldsymbol{x}$ and, for the moment, assuming a fixed $\boldsymbol{\theta}$,

[1] For any function $f(z, \boldsymbol{x})$, the conditional expectation $E[f(z, \boldsymbol{x})|\boldsymbol{x}]$ is the average of $f(z, \boldsymbol{x})$ taken with respect to the conditional probability measure $P(\cdot\,|\boldsymbol{x})$ for the random variable z.

$$
\begin{aligned}
E\left[(h(\boldsymbol{\theta},\boldsymbol{x})-z)^2\,|\,\boldsymbol{x}\right] &= E\left[(h(\boldsymbol{\theta},\boldsymbol{x})-E(z\,|\,\boldsymbol{x})+E(z\,|\,\boldsymbol{x})-z)^2\,|\,\boldsymbol{x}\right] \\
&= E\left[(E(z\,|\,\boldsymbol{x})-z)^2\,|\,\boldsymbol{x}\right]+[h(\boldsymbol{\theta},\boldsymbol{x})-E(z\,|\,\boldsymbol{x})]^2 \\
&\quad +2[h(\boldsymbol{\theta},\boldsymbol{x})-E(z\,|\,\boldsymbol{x})]E[(z-E(z\,|\,\boldsymbol{x}))|\boldsymbol{x}] \\
&= \underbrace{E\left[(z-E(z\,|\,\boldsymbol{x}))^2\,|\,\boldsymbol{x}\right]}_{\substack{\text{process variance}\\ \text{(nonmodel)}}}+\underbrace{[h(\boldsymbol{\theta},\boldsymbol{x})-E(z\,|\,\boldsymbol{x})]^2}_{\text{model error}}, \qquad (13.1)
\end{aligned}
$$

where the last line follows by $E[(z-E(z\,|\,\boldsymbol{x}))\,|\,\boldsymbol{x}]=0$. Expression (13.1) states that the MSE for a regression prediction can be decomposed into a part due to the inherent variability of the process (i.e., $E[(z-E(z\,|\,\boldsymbol{x}))^2\,|\,\boldsymbol{x}]$) and a part due to the error in the model at a specified $\boldsymbol{\theta}$ (i.e., $[h(\boldsymbol{\theta},\boldsymbol{x})-E(z\,|\,\boldsymbol{x})]^2$). The first of these two parts does not depend on the model—it simply reflects the conditional variance of the *true process*. The analyst, presumably, has no control over that. The second part, on the other hand, is directly related to the model. It is for this second part that we are interested in a bias–variance analysis. Moreover, the bias–variance analysis is fundamentally based on *estimated* values of $\boldsymbol{\theta}$, rather than the fixed value above, as we now discuss.

An analysis of the squared error of the model $[h(\boldsymbol{\theta},\boldsymbol{x})-E(z\,|\,\boldsymbol{x})]^2$ appearing as the second term in the last line of (13.1) provides direct insight into the quality of the regression function $h(\boldsymbol{\theta},\boldsymbol{x})$ as a predictor of z given $\boldsymbol{x}$ for a fixed $\boldsymbol{\theta}$. Of course, in practice, $\boldsymbol{\theta}$ is not usually fixed, but is *estimated* from a set of input–output data. That is, although the decomposition in (13.1) is of interest for analyzing the overall prediction error, it does not provide direct insight into the connection of the input–output fitting (training) data and the quality of final regression function. Hence, we must average the squared error over reasonable values of $\boldsymbol{\theta}$, which are the possible values estimated from input–output data.

Let $\{(\boldsymbol{x}_1, z_1), (\boldsymbol{x}_2, z_2),\ldots, (\boldsymbol{x}_n, z_n)\}$ be n input–output data pairs that will be collected to form the estimate of $\boldsymbol{\theta}$, say $\hat{\boldsymbol{\theta}}_n$ (recursive) or $\hat{\boldsymbol{\theta}}^{(n)}$ (batch), based on an appropriate search and optimization algorithm. The data are processed in a sequential or batch manner—as appropriate—to form the estimate of $\boldsymbol{\theta}$. If the data are processed recursively (i.e., one at a time), $\hat{\boldsymbol{\theta}}_n$ is an estimate of $\boldsymbol{\theta}$ after n iterations of an algorithm, notationally corresponding to the usage in previous chapters. If the data are processed en masse, as in the classical batch least-squares solution (Subsection 3.1.2), $\hat{\boldsymbol{\theta}}^{(n)}$ is the estimate. To avoid the cumbersome need to include multiple versions of particular equations and formulas, we also use $\hat{\boldsymbol{\theta}}_n$ when making a *generic* reference to an estimate of $\boldsymbol{\theta}$ (without specifying if it is recursive or batch). It should always be clear from the context if $\hat{\boldsymbol{\theta}}_n$ is being used to denote a recursive estimate or a generic estimate.

Following the guidelines above for a generic θ estimate, an expectation of the form $E[h(\hat{\theta}_n, x)\,|\,x]$ represents an expectation of $h(\cdot)$ conditioned on an input x, where the expectation is with respect to the randomness in the input–output data as they manifest themselves in $\hat{\theta}_n$ (since $\hat{\theta}_n$ is a function of the x_i, z_i pairs). The input x represents some fixed value of interest, perhaps corresponding to some future value that the analyst expects to encounter (x does not generally correspond to any previously observed input x_i). It is possible that a specific x and set of input–output data will yield an $h(\hat{\theta}_n, x)$ providing a good prediction, while the same x and another set of data (a different estimate $\hat{\theta}_n$) yields a poor prediction.

A useful approach to analyzing the inherent model quality is to take the mean of the squared model error $\left[h(\hat{\theta}_n, x) - E(z\,|\,x)\right]^2$ based on averaging with respect to the distribution for the fitting data. This is equivalent to averaging with respect to the resulting distribution for $\hat{\theta}_n$. It can be shown (Exercise 13.1) that for any future x this MSE is

$$E\left\{\left[h(\hat{\theta}_n, x) - E(z\,|\,x)\right]^2 \Big| x\right\}$$

$$= \underbrace{E\left\{\left[h(\hat{\theta}_n, x) - E\left(h(\hat{\theta}_n, x)\,|\,x\right)\right]^2 \Big| x\right\}}_{\text{variance at } x} + \underbrace{\left[E\left(h(\hat{\theta}_n, x)\,|\,x\right) - E(z\,|\,x)\right]^2}_{(\text{bias at } x)^2}. \tag{13.2}$$

An unbiased estimator is one with $E\left(h(\hat{\theta}_n, x)\,|\,x\right) = E(z\,|\,x)$ (implying that the second term on the right-hand side of (13.2) is zero).

As a final *overall* assessment of contributions toward the model MSE, one can average the squared bias and variance in (13.2) over all possible values of x, yielding mean values, say $\overline{\text{bias}^2}$ and $\overline{\text{variance}}$. If x is generated randomly, then, of course, the averaging is with respect to the probability measure for x. If x is chosen deterministically, then the averaging is with respect to plausible future values of x and their expected frequency. In either case, averaging the variance and squared bias terms in (13.2) leads to a global measure of the contributions to the MSE that does not depend on a specific value of x:

$$\begin{aligned} \text{MSE}_{\text{overall}} &= E_x\left[E\left\{\left[h(\hat{\theta}_n, x) - E(z\,|\,x)\right]^2 \Big| x\right\}\right] \\ &= E_x\left[\text{variance at } x + (\text{bias at } x)^2\right] \\ &= \overline{\text{variance}} + \overline{\text{bias}^2}, \end{aligned} \tag{13.3}$$

where $E_x[\cdot]$ denotes the appropriate stochastic or deterministic average over possible values of $\boldsymbol{x}$. (Exercise 13.2 shows the importance of proper averaging with respect to the bias contribution. If one replaces the mean of bias^2 in (13.3) with the square of the mean of the bias, it is possible to obtain nonsensical values of overall MSE.)

13.1.2 Interpretation of the Bias–Variance Tradeoff

The subsection is devoted to interpreting the bias–variance tradeoff as it relates to picking the best model. Although unbiasedness is generally considered a desirable property, (13.2) and (13.3) show that an unbiased estimator can have a large MSE when the variance is large. In particular, a biased estimator may have a lower MSE than an unbiased estimator, even better than unbiased estimators that are best by some criterion (such as an estimator with the lowest variance among unbiased estimators that are linear combinations of the measurements). We begin with a simple example illustrating this point.

Example 13.1—Biased estimator with lower MSE than obvious unbiased estimator. Consider a sequence of n scalar independent, identically distributed (i.i.d.) measurements $\{z_1, z_2, \ldots, z_n\}$ having mean μ and variance $\sigma^2 > 0$. We can write $z_i = \mu + v_i$, where the v_i are mean-zero noises. The aim is to find a good estimator for the unknown μ. This is a simple case for the function $h(\cdot)$ since there is no input $\boldsymbol{x}$ (i.e., $h(\theta, \boldsymbol{x}) = \theta$, where $\theta = \mu$). Let $\hat{\theta}_n$ represent the estimate of μ (so $h(\hat{\theta}_n, \boldsymbol{x}) = \hat{\theta}_n$). The obvious unbiased estimator of μ is the sample mean, say $\hat{\theta}_n = \bar{z}$ (which may be computed in either recursive or batch form). Among all unbiased estimators that are linear combinations of the data, this estimator is minimum variance (and hence minimum MSE since the bias is zero; Wilks, 1962, pp. 279–280; Bickel and Doksum, 1977, p. 143).[2]

An alternative biased estimator is $\hat{\theta}_n = r\bar{z}$, where $0 < r < 1$. The bias and variance of the alternative estimator are

$$\text{bias} = E(r\bar{z}) - \mu = (r-1)\mu,$$

$$\text{variance} = \text{var}(r\bar{z}) = r^2 \frac{\sigma^2}{n}.$$

Expression (13.2) or, equivalently in this case of no dependence on $\boldsymbol{x}$, (13.3) then lead to an MSE of

[2]The properties of the sample mean are slightly different if *both* μ and σ^2 are being estimated (versus only μ). If σ is unknown and if the data are normally distributed, then the sample mean is the minimum MSE estimator of μ among *all* (not just linear) *unbiased* estimators (e.g., Bickel and Doksum, 1977, pp. 123–124). This is a consequence of the notion of completeness and sufficiency for estimators, a topic not considered here.

$$\text{MSE} = r^2 \frac{\sigma^2}{n} + (r-1)^2 \mu^2 .$$

If σ^2/n is sufficiently large relative to μ^2, then for a given r the alternative estimator has a lower MSE than $\bar{z}$ (which has an MSE of σ^2/n). For example, if $\sigma^2/n = 0.1$, $\mu^2 = 0.1$, and $r = 0.5$, then the MSE for $r\bar{z}$ is $0.25(0.1 + 0.1) = 0.05$, while the MSE for $\bar{z}$ is 0.1.

Unfortunately, this result is not particularly useful in practice, since the question of whether $r\bar{z}$ or $\bar{z}$ is a better estimator for a specific value of r can only be answered if one knows μ, the very quantity being estimated! Nevertheless, it does show that a biased estimator can yield a lower MSE than an unbiased estimator (see also Exercise 13.3). ❑

There is generally a tradeoff between the variance and bias contributions to the overall MSE. Regression functions $h(\cdot)$ with high variance tend to have low bias and vice versa. One can see this in Example 13.1 by letting r range from near 0 (high bias/low variance) to near 1 (low bias/high variance). More generally, when $h(\cdot)$ depends on an input $\boldsymbol{x}$, there is a relationship between the complexity of the model and the relative bias and variance. In particular, the following relationships typically hold:

Simple model	$\Leftrightarrow$	**High bias/low variance**
Complex model	$\Leftrightarrow$	**Low bias/high variance**

The bias–variance tradeoff provides a framework for choosing among candidate models. Of course, in practice, many factors other than bias, variance, and the resulting MSE may be relevant. These include cost, development time, historical precedent for particular model forms, desires of organizational leadership, and so on. Nevertheless, all other factors being equal, one would wish to pick the function $h(\cdot)$ with a balanced bias and variance. This balance in bias and variance results in the minimum MSE according to (13.2) or (13.3).

Figure 13.1 presents three plots to illustrate the above relationships on a simple problem with scalar input x. Each plot uses the same two sets of five data points (with both sets being at the same input values $x_1, x_2, \ldots, x_5$). One of the data sets is used to fit the curve and the other independent set is used to test the fit. Each of the three individual figures shows the result of a model fit based on a polynomial model with specified p (the polynomial has order $p - 1$ to allow for the additive constant term in the polynomial).

Figure 13.1(a) shows a case where the model is too complex relative to the data. Here the curve perfectly matches the five data points available for

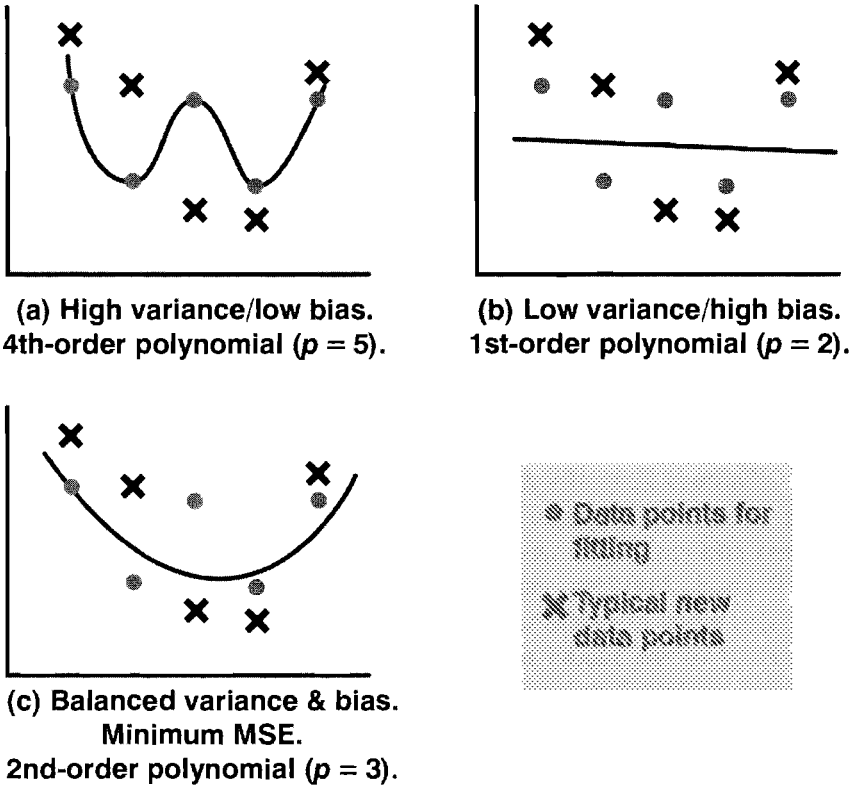

Figure 13.1. Illustration of the bias–variance tradeoff in model selection in a simple problem. Model in part (c), which has minimum MSE relative to the fitting data, is also most consistent with the new data points not used in fitting model.

fitting the model. Hence the variability of $h(\cdot)$ at each x will be identical to that of the data (z) itself. That is, at $n = 5$, $E\{[h(\hat{\theta}_n, x) - E(h(\hat{\theta}_n, x)|x)]^2 | x\} = E[(z - E(z|x))^2 | x]$, for any $x \in \{x_1, x_2, \ldots, x_5\}$ and generic (batch or nonbatch) estimate $\hat{\theta}_n$. Note that the bias of $h(\cdot)$ relative to the fitting data set is zero since $z = z(x) = h(\hat{\theta}_n, x)$ trivially implies that $E(z|x) = E[h(\hat{\theta}_n, x)|x]$. Figure 13.1(b) shows a case with nearly the opposite character. This model is too simple relative to the data. Given the limited flexibility in the curve, there will be relatively little variation in $h(\cdot)$ at each x as new fitting data sets are collected. This provides for a small variance, $E\{[h(\hat{\theta}_n, x) - E(h(\hat{\theta}_n, x)|x)]^2 | x\}$, but a large bias since $h(\cdot)$ will not track z very well due to the rigidity in $h(\cdot)$. Figure 13.1(c) shows a curve that balances the bias–variance tradeoff with a curve of "reasonable" flexibility. A visual examination of the three curves relative to the new testing data (the **x** points) reveals that the balanced $p = 3$ (quadratic) model of Figure 13.1(c) seems to best match the new data.

Unfortunately, the bias–variance tradeoff is largely limited to gaining a *conceptual* understanding for comparing different models. Because the probability distributions for the input–output data, together with the resulting distribution of $\hat{\theta}_n$, are not known (they depend on knowing the unknown true model), the values of the bias and variance will generally be unknown. It is, however, clear from the bias–variance tradeoff that there can be no universal best model form. One model form may provide nicely balanced bias and variance on one class of problems, but be too rigid (high bias) or flexible (high variance) in another example. For a *fixed* model form, the variance contribution to MSE tends to decrease when the sample size used in fitting the model is increased. Intuitively, this follows since the model quality improves from the greater information available for fitting the model. The variance contribution to MSE decreases with a greater amount of data since there is a reduced tendency to fit to the individual data points.

13.1.3 Bias–Variance Analysis for Linear Models

For the important special case of linear models, let us present an explicit form for the bias and variance. Although the bias and variance provide useful insight (see Example 13.2), they are generally not computable in practice because they depend on quantities that are unknown. Suppose that the *true process* generates data according to $z = f(\boldsymbol{x}) + \eta$, where $f(\boldsymbol{x})$ is an unknown (possibly nonlinear) function and η is an independent error having mean zero and variance σ^2. Suppose that the classical linear regression model (Section 3.1) is used to describe the process,

$$z_k = \boldsymbol{h}_k^T \theta + v_k ,$$

where $\boldsymbol{h}_k$ is the design vector (dependent on input $\boldsymbol{x} = \boldsymbol{x}_k$) and v_k is a noise term having common variance across k. Suppose further that classical (batch) least-squares (Subsection 3.1.2) is used to estimate θ (producing $\hat{\theta}^{(n)}$). Then, the prediction for a *future* output $z = z(\boldsymbol{x})$ based on n input–output pairs $\boldsymbol{h}_k$, z_k being used for estimating θ is given by

$$\begin{aligned} \hat{z}(\boldsymbol{x}) &= \boldsymbol{h}(\boldsymbol{x})^T \hat{\theta}^{(n)} \\ &= \boldsymbol{h}(\boldsymbol{x})^T (\boldsymbol{H}_n^T \boldsymbol{H}_n)^{-1} \boldsymbol{H}_n^T \boldsymbol{Z}_n , \end{aligned}$$

where $\boldsymbol{h}(\boldsymbol{x})$ is a $p \times 1$ vector dependent on the input $\boldsymbol{x}$, $\boldsymbol{Z}_n = [z_1, z_2, \ldots, z_n]^T$, and $\boldsymbol{H}_n$ is the $n \times p$ concatenated matrix of $\boldsymbol{h}_k^T$ row vectors.

In computing the average bias and variance according to (13.3), suppose that there are m future $\boldsymbol{x}$ values of interest, $\chi_1, \chi_2, \ldots, \chi_m$. We are interested in the bias and variance of $\hat{z}(\boldsymbol{x})$ averaged over these m input values (this is the averaging that is being used in concert with (13.3)). Given the estimate of θ

based on the original n input measurements $x_1, x_2, \ldots, x_n$ (as reflected in $\boldsymbol{H}_n$) together with the corresponding output measurements $\boldsymbol{Z}_n$, the stacked value of predictions for the m new inputs is $[\hat{z}(\chi_1), \hat{z}(\chi_2), \ldots, \hat{z}(\chi_m)]^T$, where the $\hat{z}(\cdot)$ function is given above. The corresponding matrix of the m new inputs as reflected in $\boldsymbol{\mathcal{k}}(\cdot)$ is $\boldsymbol{\mathcal{H}}_m \equiv [\boldsymbol{\mathcal{k}}(\chi_1), \boldsymbol{\mathcal{k}}(\chi_2), \ldots, \boldsymbol{\mathcal{k}}(\chi_m)]^T$.

Then, the vector of predictions for the m new inputs is

$$\begin{bmatrix} \hat{z}(\chi_1) \\ \hat{z}(\chi_2) \\ \vdots \\ \hat{z}(\chi_m) \end{bmatrix} = \boldsymbol{\mathcal{H}}_m \hat{\theta}^{(n)} = \boldsymbol{\mathcal{H}}_m (\boldsymbol{H}_n^T \boldsymbol{H}_n)^{-1} \boldsymbol{H}_n^T \boldsymbol{Z}_n \equiv \boldsymbol{S}_{m|n} \boldsymbol{Z}_n .$$

From this prediction function, the average bias-squared and variance for the m predictions based on the n data for estimating θ, now follow:

$$\overline{\text{bias}^2} = \frac{1}{m} \sum_{k=1}^{m} \left\{ \boldsymbol{s}_k^T \begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{bmatrix} - f(\chi_k) \right\}^2 , \tag{13.4}$$

$$\overline{\text{variance}} = \frac{\sigma^2}{m} \text{trace}(\boldsymbol{S}_{m|n} \boldsymbol{S}_{m|n}^T) , \tag{13.5}$$

where $\boldsymbol{s}_k^T$ is the kth row in $\boldsymbol{S}_{m|n}$ (Exercise 13.4). In practice, of course, the $f(\cdot)$ values and (probably) σ^2 will be unknown. Expressions (13.4) and (13.5) represent the $\overline{\text{bias}^2}$ and $\overline{\text{variance}}$ terms that appear in (13.3).

Although (13.4) and (13.5) cannot typically be used *directly* in practice to evaluate candidate models, they can be used to provide valuable insight. Example 13.2 considers a specific $f(\cdot)$ and σ^2 as a means of illustrating the bias–variance tradeoff. This example compares the bias and variance contributions to MSE for several candidate models in the curvilinear form (Subsection 3.1.1) when the true data-generating process is nonlinear. Example 13.3 considers the effect of increasing sample size on the MSE.

Example 13.2—Bias–variance tradeoff for curvilinear models. Consider a problem where the true process is $f(x) = (x + x^2)^{1.1}$, with x a scalar input. Suppose that the additive independent noise η for the true process has $\sigma^2 = 100$. Using the standard batch least-squares estimate, let us compare the bias and variance for three candidate curvilinear models of the form $\hat{z}(x) = \boldsymbol{\mathcal{k}}(x)^T \hat{\theta}^{(n)}$:

Model 1 (linear): $p = 1$; $\boldsymbol{\mathcal{k}}(x) = x$,

$$\textbf{Model 2 (quadratic):}\ p = 2;\ \mathcal{A}(x) = [x, x^2]^T,$$

$$\textbf{Model 3 (cubic):}\ p = 3;\ \mathcal{A}(x) = [x, x^2,\ x^3]^T.$$

(Unlike Figure 13.1, there is no additive constant in the models; hence, the value of p corresponds directly to the polynomial order.) Note that model 2 with $\hat{\boldsymbol{\theta}}^{(n)} \approx [1, 1]^T$ is almost the correct model (differing in form by only the exponent 1.1). Hence, it might be expected that model 2 will provide the optimal bias–variance tradeoff. Let us see if this is true.

In computing the average bias and variance according to (13.3), suppose that $m = n$ and that the future x values are the same as the n values of x_k that were used in estimating $\boldsymbol{\theta}$. In particular, suppose that $x_k = \chi_k = k$, $k = 1, 2, \ldots, m$, and that $m = n = 10$ (i.e., $\mathcal{H}_m = \boldsymbol{H}_n$). For example, in the case of $p = 2$,

$$\mathcal{H}_m = \boldsymbol{H}_n = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 1 & 4 & 9 & 16 & 25 & 36 & 49 & 64 & 81 & 100 \end{bmatrix}^T.$$

Table 13.1 presents the bias–variance analysis.

As shown in Table 13.1, model 2 is the preferred model (the lowest MSE), although model 3 has an MSE that is similar (due to the value in using a cubic term to capture some of the nonpolynomial effects caused by the exponent 1.1). Note the overall pattern of decreasing bias and increasing variance as the model complexity increases, consistent with Figure 13.1. ❑

Example 13.3—Effect of sample size. Consistent with the discussion of the preceding subsection, let us show how an increasing sample size (n) can reduce MSE via a reduction in the variance contribution. With the exception of the change in n, the setting is identical to Example 13.2. Suppose that $n = 20$, with the additional 10 inputs being identical to the initial 10 inputs, $x_k = x_{k+10} = k$, $k = 1, 2, \ldots, 10$. Then, using (13.5), $\overline{\text{variance}}$ for models 1, 2, and 3 is 5.0, 10.0, and 15.0, respectively. Because of the doubling of the sample size for fitting, these

Table 13.1. Average bias and variance and overall MSE for candidate curvilinear models.

	$\overline{\text{bias}^2}$	$\overline{\text{variance}}$	Overall MSE
Model 1 (linear)	510.6	10.0	520.6
Model 2 (quadratic)	0.53	20.0	20.53
Model 3 (cubic)	0.005	30.0	30.005

variances are half the magnitude of the variances in Table 13.1. The bias is unchanged. Hence, as expected, the overall MSE is reduced with an increase in the amount of fitting data. ❑

Given the conceptual insight of the bias–variance tradeoff, we are now in a position to consider practical means of optimizing this tradeoff. The next section addresses this issue.

13.2 MODEL SELECTION: CROSS-VALIDATION

There are a great number of methods for approximately addressing the bias–variance tradeoff in a manner that is feasible for implementation. These methods are variations on the theme of balancing low- and high-order requirements to produce a model that (implicitly at least) balances the bias and variance in a manner similar to Figure 13.1. Some of the best-known methods include the information criterion (AIC) (Akaike, 1974), the principle of minimum description length (Rissanen 1978; Wei, 1992), bootstrap model selection (Efron and Tibshirani, 1997), Bayesian model selection (Akaike, 1977; Schwarz, 1978; George and McCulloch, 1997), V-C dimension (Vapnik and Chervonenkis, 1971; Cherkassky and Mulier, 1998, Chap. 4), the Fisher information criterion (Wei, 1992), and cross-validation (Allen, 1974; Stone, 1974; Geisser, 1975). These methods rely on approaches such as maximum likelihood, the Fisher information matrix (see Section 13.3), information theory, regression, computer-based resampling, risk minimization, and sample fitting.

The basic principle in model selection is to minimize, implicitly or explicitly, a criterion of the generic form

$$f_1(\text{fitting error from given data}) + f_2(\text{model complexity}), \tag{13.6}$$

where $f_1(\cdot)$ and $f_2(\cdot)$ are increasing functions of, respectively, some measure of the error in the model predicting the fitting data (the fitting error) and some measure of the number of terms in the model (model complexity). Good general discussions of model selection methods appear in Linhart and Zucchini (1986), Shao (1997), McQuarrie and Tsai (1998), and Ljung (1999, Chap. 16). Applications of some of these methods to neural networks are discussed in Geman et al. (1992). For reasons of wide applicability and popularity, this section focuses on the cross-validation method of model selection. In some applications, however, one of the other methods may be more effective. The reader seriously interested in a broader review of the important subject of model selection is directed to the references above.

The cross-validation approach is perhaps the most straightforward formal model selection method to understand and to implement. It is based on manipulations of the fitting (training) data (i.e., the data assumed available for model estimation.) Cross-validation does not require additional data and/or

detailed prior information or analytical analysis beyond sample model fits. Cross-validation also has the advantage of applying to candidate models of virtually any form, not being restricted to specific classes of candidate models (e.g., linear/curvilinear regression models), as are some of the approaches mentioned above. Also, unlike some other approaches, it does not require that the underlying data be normally distributed. On the other hand, cross-validation is not necessarily the most powerful or discerning method in any specific problem, nor is it the most computationally efficient (since it requires repeated "sample" model fits). Cross-validation is one of the model-selection methods that *implicitly* optimizes the tradeoff criterion (13.6), as there is no direct construction of a performance metric dependent on $f_1(\cdot)$ and $f_2(\cdot)$. (Cross-validation has other applications as well. For example, one other use is helping determine when to stop an algorithm's iteration process; see Amari et al., 1997.)

Let us sketch how cross-validation works in the context of selecting a model. A more formal step-by-step description is given later in this section. Suppose that two or more candidate model forms are to be evaluated. For example, an analyst may have small-, medium-, and large-scale neural networks as candidate model forms and wishes to know which neural network is likely to produce the best predictions. Cross-validation is a commonsense approach based on sequentially partitioning the full data set into fitting and test *subsets*. For each partition, estimates are produced for the candidate model forms from the fitting subset. Then, the performance of each candidate model is measured on the test subset. This procedure is repeated for all partitions of the full data set.

Let n_T denote the size of the test subsets, where, of course, $n_T < n$, with n the size of the full data set. A common strategy—called *leave-one-out*—is to pick $n_T = 1$ and cycle through all n possible combinations of fitting and test subsets (e.g., Stone, 1974; Allen, 1974). This approach produces n model fits from the n possible fitting subsets of size $n - 1$. Each of these model fits generates a prediction error on the one data point left out (i.e., the difference between the outcome of the point left out and the predicted value based on the model fit from the remaining $n - 1$ points). The best model form is the one for which the chosen type of average for these n prediction errors—say, the sample MSE or mean absolute deviation (MAD)—is lowest. (We do not include the qualifier "sample" below, but it should be clear from context that the MSE and other values are not analytically based, but are derived from the specific sample.)

There are often advantages, however, to choosing $n_T > 1$. The advantages include greater efficiency (i.e., fewer model fits) for *some* implementations with $n_T > 1$ (but definitely not all implementations, as we see shortly). There is also some theoretical and empirical evidence that this n_T-fold ($n_T > 1$) approach produces more accurate results than the leave-one-out strategy (Breiman and Spector, 1992; Shao, 1993; Breiman, 1996). In fact, for linear models, the leave-one-out strategy has been shown in Shao (1993) to be biased to picking models with excessive complexity (i.e., p too large). When $n_T > 1$, the test subsets may be chosen deterministically or randomly, with or without replacement. For test

subsets chosen deterministically with replacement (i.e., all possible combinations of test subsets of size n_T are used), there are a potentially huge number of possible fitting/test subset combinations. In particular, the number of combinations is "n choose n_T" $\binom{n}{n_T}$. For example, with $n = 30$ and $n_T = 6$ (as in Example 13.4), cross-validation in this manner would require over 590,000 sample model fits!

One way of mitigating this explosion is to randomly select (usually with replacement) a relatively small number of test subsets of size n_T (e.g., Shao, 1993). Another approach is to choose n_T such that n is divisible by n_T and then choose the test subsets *without replacement* so that all of the data appear once and only once in a test subset. The allocation of the n data to the n/n_T test subsets may be done randomly or deterministically (e.g., Neter et al., 1996, p. 437; Cherkassky and Mulier, 1998). The "once and only once" aspect may be viewed as an extension of the leave-one-out strategy. This allocation reduces the number of modelfits from n in leave-one-out and from $\binom{n}{n_T}$ in deterministic replacement selection to n/n_T (e.g., from 30 and over 590,000, respectively, to only 5 in the illustration above).

Figure 13.2 presents a schematic of the process of partitioning the data into three combinations of fitting and test subsets when the test subsets are chosen deterministically or randomly so that all data serve once (and only once) in a test subset. Hence, the test subsets are disjoint. The deterministic version of this disjoint sampling procedure is used in Examples 13.4 and 13.5. Note that while the test subsets may be independent (given that the raw data are independent), the fitting subsets typically share data as in Figure 13.2. For example, each fitting subset in Figure 13.2 shares half of its data with each of the other fitting subsets. As described in the leave-one-out strategy above, the model form that produces the best performance across the sequence of test subsets with respect to a specified metric is chosen as the best model form. Typically, the performance metric is the MSE (equivalently, the root-mean-squared error, RMS) of the predictions over the multiple test subsets, although other approaches may be used as well (such as MAD in Example 13.5).

After determining the best model form via cross-validation, the *full* data set is used to produce the final estimates for the parameters of this model. Obviously, there are some choices to be made in implementing the cross-validation approach. In particular: the type of partitioning for the data, the metric by which the models will be compared on the sequence of test sets (MSE, MAD, etc.), whether to use random or deterministic sampling for the test subsets, and so on. Guidelines for these choices are discussed, for example, in Shao (1993), Neter et al. (1996, pp. 437–439), Cherkassky and Mulier, (1998, pp. 78–79), and McQuarrie and Tsai (1998, pp. 251–261). In practice, however, an intuitive "feel" is often the primary guide since the more formal guidelines are generally restricted to relatively narrow model classes.

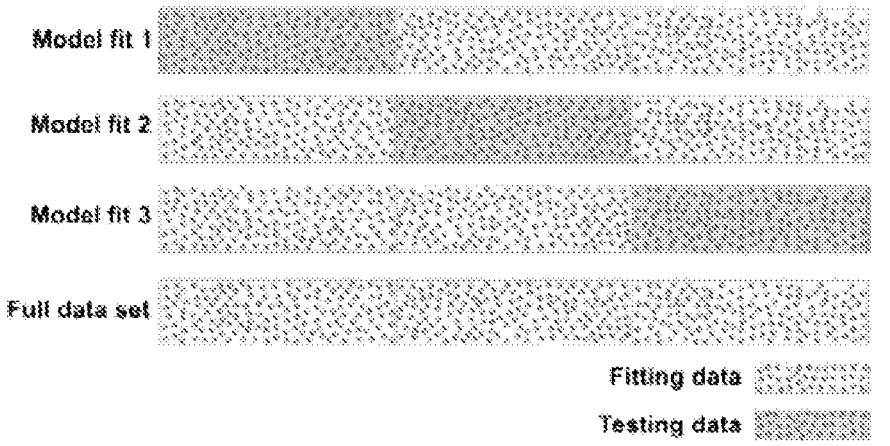

Figure 13.2. Cross-validation with three model fits based on three subsets of fitting data within the full data set. The data are illustrated schematically as random points. The test subsets are disjoint, with the union covering the full data set. The models are compared based on performance over three test subsets. The winning model form is fit with the full data set after the cross-validation is complete.

The steps given below are a formal summary of the procedure outlined above. These steps describe a typical implementation of cross-validation. The performance metric discussed below is MSE, but the overall approach is identical if a different metric is used. Unlike the bias–variance calculations of Section 13.1, the following steps are based solely on the *available* model $h(\cdot)$ and data. It is not necessary to know the unknown: the true data-generating mechanism or the distribution of the noise in the process.

Implementation of Cross-Validation with Disjoint Test Subsets

Step 0 (**Initialization**) Determine n_T and the strategy for choosing the disjoint test subsets; n is assumed divisible by n_T. Set $m = 1$ and $i = 1$, where m is the counter for the candidate model being considered and i is the counter for the test subset being used from the n/n_T possible test subsets.

Step 1 Consider the mth model (the mth candidate form for $h(\cdot)$). For the ith test subset, let the remaining $n - n_T$ elements be the ith fitting subset. Estimate $\boldsymbol{\theta}$ for the mth model from this fitting subset.

Step 2 Based on the estimate for $\boldsymbol{\theta}$ from step 1, compare the predictions of the mth model and the data in the ith test subset. Suppose that the MSE is being used for this comparison. Then let $\text{MSE}_i(m)$ denote the mean of the sum of squared errors over the n_T points in the ith test subset. (If another metric is used, simply replace MSE in the steps below with that metric.)

Step 3 Update i to $i + 1$ and return to step 1. Terminate when all data have been included once and only once in a test subset (corresponding to a terminal value of $i = n/n_T$). Upon termination, let $\overline{\text{MSE}}(m)$ denote the mean of the n/n_T values for $\text{MSE}_i(m)$ (i.e., $\overline{\text{MSE}}(m)$ is the overall MSE for model m across the n/n_T test subsets).

Step 4 Repeat steps 1 to 3 for the next model by updating m to $m + 1$ and resetting i to $i = 1$. Terminate when the cross-validation calculations have been performed for all models.

Step 5 Choose the model corresponding to the lowest $\overline{\text{MSE}}(m)$ as the best model.

One of the limitations in formal model selection in general—including cross-validation—is that statistical tests of whether one model is better than another "...are difficult to construct and one must then rely on a simple comparison between estimated expected discrepancies" (Linhart and Zucchini, 1986, p. 15). For cross-validation, this is associated with step 5 above. Although it is usually relatively straightforward to determine the model with the lowest $\overline{\text{MSE}}(m)$ (or sample mean of other metric if appropriate), it is difficult to know if the lowest value is statistically significantly lower than the value of other candidate models.[3] Might the choice of a particular winning model just be an anomaly of the particular data set?

Unfortunately, there is very little information available to formally determine the P-value (say) associated with the difference in $\overline{\text{MSE}}(m)$ values for the candidate models. This question of statistical significance appears to have no easy answer given that the n/n_T values of $\overline{\text{MSE}}(m)$ will be statistically dependent in a complicated way. That is, the $\overline{\text{MSE}}(m)$ contributions depend—through the parameter estimates and testing subsets—on the *same* full set of data. An additional complication follows from this being a multiple comparisons problem (Chapter 12) since, in general, there will be more than two candidate models.

The dependence and multiple comparisons aspects preclude the use of standard techniques for comparisons such as the matched or unmatched t-tests

[3]The lack of formal statistical justification for choosing one criterion value as being significantly better than another (in the sense of something like a P-value) is not unique to cross-validation. Other approaches, not surprisingly, suffer the same shortcoming, as they, too, depend on nontrivial transformations of the same data set, leading to a comparison based on highly dependent test measures. There are, however, special cases where it is possible to derive (at least approximate) distributions for test statistics associated with the difference in models and/or derive the expected value and variance associated with such differences (see, e.g., McQuarrie and Tsai, 1998, pp. 25–27, for the approximate mean and variance of the difference for AIC in linear regression models). Bayesian methods for model selection eliminate the need to work with test statistics per se, but introduce other complications associated with the choice of prior distribution and the need to carry out numerical integration (e.g., Schwarz, 1978; George and McCulloch, 1997).

(Appendix B). Hence, typical applications of cross-validation simply rely on the outcome in step 5 without further statistical testing and without assigning a *P*-value to the outcome. (*Limited* inference results, however, are available. For instance, Example 13.5 uses an independent data set to confirm the outcome of the cross-validation. McQuarrie and Tsai, 1998, pp. 254, 258–259, describe approximate tests for leave-one-out cross-validation that apply in comparing two linear models with normally distributed noise when one of the candidate models is the true model.)

Two demonstrations of cross-validation are given below. Example 13.4 is for an artificial data set, and Example 13.5 is for the oboe reed data of Section 3.4. Example 13.4 illustrates cross-validation for a relatively low noise level in the measurements, while Example 13.5 involves a larger noise contribution.

Example 13.4—Cross-validation on artificial data set. This example is similar to a problem in Cherkassky and Mulier (1998, pp. 85–88). Although the system and data are artificial, the general model selection process here is identical to what typically occurs with a real system. A simulated set of $n = 30$ data points is produced from the true system

$$z_k = \sin(2\pi x_k) + \eta_k, \tag{13.7}$$

where η_k is i.i.d. $N(0, 0.1^2)$ and the scalar inputs x_k are generated as i.i.d. $U(0, 1)$ random variables. (As needed in Exercise 13.6, the input–output data for this example are in the file **`sinedata`** at the book's Web site.) For a scalar input x, consider candidate regression models of the polynomial form

$$h(x, \theta) = \sum_{i=0}^{p-1} t_{i+1} x^i, \tag{13.8}$$

where $\theta = [t_1, t_2, \ldots, t_p]^T$, as usual. Note that this model is of the curvilinear form (Subsection 3.1.1) and hence specialized techniques for model determination in standard linear regression (as reviewed in Cherkassky and Mulier, 1998, p. 229, or Ljung, 1999, Chap. 16) can be used. However, we will use a generic cross-validation approach, which applies equally in linear and nonlinear models. All model fitting for cross-validation here is carried out with ordinary least-squares estimation (Section 3.1).

We choose $n_T = 6$ points for the $n/n_T = 5$ disjoint testing subsets. Hence, $i = 1, 2, 3, 4$, or 5 in the notation of the five-step procedure for cross-validation above. The five test subsets are chosen as elements 1 to 6, 7 to 12,..., and 25 to 30. Because the input data are generated uniformly, we know that each of these five test subsets is statistically representative of the full data set. For each division of the full data set, we fit the model with the subset of $n - n_T = 30 - 6 = 24$ data points left in the fitting set after removing the points for the testing subset (e.g., elements 7 to 30 are the fitting subset for the first test subset of elements 1

to 6). We then choose the model that provides the best overall fit over the multiple test subsets of data. The measure of fit reported here is the RMS error over all five test subsets, calculated by averaging the MSEs over the test subsets and taking the square root (i.e., $[\overline{\text{MSE}}(m)]^{1/2}$ in the notation of the cross-validation implementation steps above; RMS is reported here to maintain an error measure in the same units as the z_k values).

Let us consider three candidate polynomial models: a linear (affine) model ($p = 2$, including the additive constant), a third-order polynomial ($p = 4$), and a tenth-order polynomial ($p = 11$). Hence, the model counter $m = 1$, 2, or 3 in the notation of the five-step procedure above. Table 13.2 shows the RMS errors across the five test subsets for the three candidate models.[4]

Table 13.2. RMS errors in cross-validation for the three candidate models. RMS errors derived from MSE over all test subsets.

Linear model ($p = 2$)	Third-order polynomial ($p = 4$)	Tenth-order polynomial ($p = 11$)
0.610	0.129	3.78

As described in the five-step procedure above, we choose as the best model the one with the lowest overall RMS error (equivalent to the lowest MSE, of course). Hence, the third-order (cubic) polynomial yields the best fit according to the cross-validation principle. The linear function exhibited excessive bias with the inadequate flexibility of a straight line, while the tenth-order polynomial tended to fit too closely to the 24 data points of each fitting subset, causing some very large individual errors on the testing subsets (and hence contributing to the large overall RMS). The cubic polynomial did a nice job of balancing the bias and the variance. Figure 13.3 shows the 30 data points in **sinedata**, the true sine function, and the cubic polynomial with coefficients estimated from the full set of 30 points. Visually, the cubic polynomial provides a nice fit to the true sine function. ❑

[4]The RMS error for the $p = 11$ model in the table is only an approximation. The matrices corresponding to $(\boldsymbol{H}_{n-n_T}^T \boldsymbol{H}_{n-n_T})^{-1}$ in the basic least-squares formula in Subsection 3.1.2 are ill-conditioned, leading to some numerical instability (an example of multicollinearity in regression). A MATLAB-based calculation of the RMS error is 3.90 (versus the MS EXCEL result of 3.78 in the table). Despite the instability, it is clear that the RMS error for $p = 11$ is significantly larger than the error for $p = 4$.

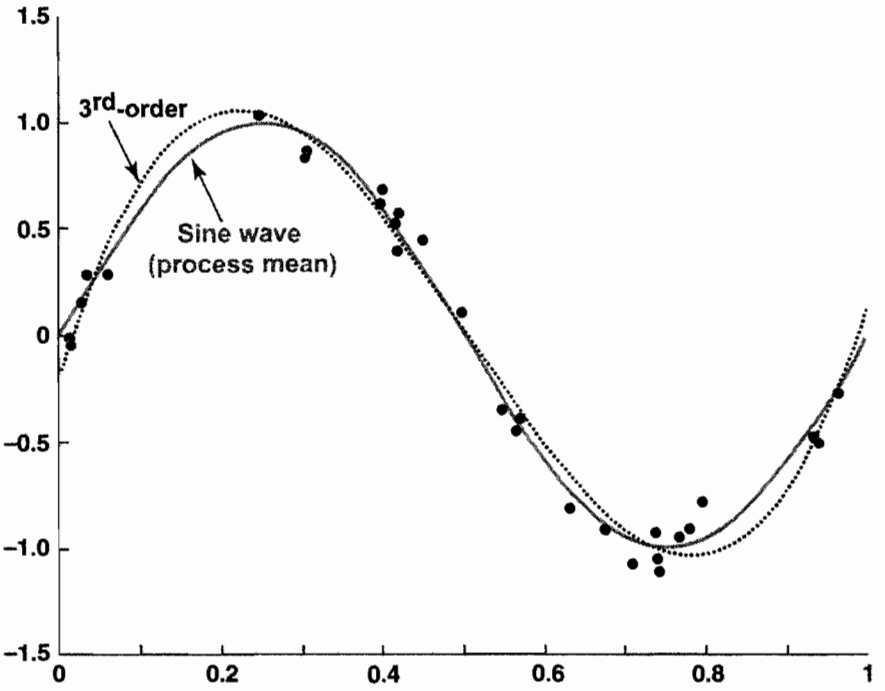

Figure 13.3. True mean and fitted model in a low-noise environment. The 30 data points are from the true process (13.7). The third-order polynomial model is chosen from cross-validation. The sine function corresponds to the mean of the data-generating process.

Example 13.5—Cross-validation with oboe reed data. Section 3.4 described the problem of predicting the final quality of an oboe reed from indicators that are available early in the reed-making process. These indicators represent the input variables $\boldsymbol{x}$. Based on the data set **`reeddata-fit`** (at the book's Web site), let us use cross-validation to compare two candidate models. The first model is the standard linear model given in Section 3.4, with $\boldsymbol{x}$ composed of all six input variables T, A, E, V, S, F:

$$z = \theta_{\text{const}} + \theta_T T + \theta_A A + \theta_E E + \theta_V V + \theta_S S + \theta_F F + v. \tag{13.9}$$

This model has $p = 7$ (including the additive constant). The second model is a simplified form with only first element of $\boldsymbol{x}$, "top close" (T), as an input:

$$z = \theta_{\text{const}} + \theta_T T + v, \tag{13.10}$$

implying that $p = 2$. As with Example 13.4, all model fitting here (for cross-validation and final estimation) is carried out with ordinary least-squares estimation.

The cross-validation process uses the 160 input–output measurements in **reeddata-fit**. As in Figure 13.2, we sequentially divide the full data set into pairs of fitting and disjoint test subsets, with each fitting subset containing 120 measurements and each test subset containing the remaining 40 measurements. These four pairs of fitting/testing subsets are chosen based on picking the four test subsets as disjoint measurements 1 to 40, 41 to 80, 81 to 120, and 121 to 160. The fitting data for each of the pairs are the remaining 120 elements (e.g., measurements 41 to 160 in the first pair). Relative to the five steps of cross-validation above, the index i runs over 1, 2, 3, and 4. Because the data exhibit trends in $\boldsymbol{x}$ due to certain similarities in the raw cane for measurements collected near the same time, it may have been useful to randomize the order of the measurements before constructing the disjoint test subsets. This randomization was not done in the study here.

Table 13.3 shows the RMS and MAD errors over the four test subsets for the two candidate models. The RMS estimates are the square roots of the mean of the four sample MSEs taken over the test subsets (analogous to Example 13.4). The MAD values are the sample means of the prediction errors $|\hat{z}_k - z_k| = |\boldsymbol{x}_k^T \hat{\theta} - z_k|$, where $\boldsymbol{x}_k$ denotes the kth input vector (or input scalar for model (13.10)), z_k is the corresponding output, and $\hat{\theta}$ denotes one of the four estimates for θ formed from one of the fitting subsets containing 120 measurements. The MAD estimate is the mean over the four test subsets.

From Table 13.3, both the MAD and RMS errors suggest that the full linear model provides superior predictions. This is not surprising given that the reduced model omits the critical "first blow" (F) input variable. Note, however, that we have not provided a formal statistical justification for this choice. As discussed following the generic cross-validation steps above, the individual MAD or MSE contributions for the four subsets of test data are dependent in a complicated way, implying that there is no known method for calculating P-values or providing other formal justification for the choice of model (13.9) over model (13.10). In particular, the four MAD or MSE contributions depend—through the parameter estimates and testing subsets—on the *same* data in **reeddata-fit**.

For this problem, however, we have the luxury of a *separate* (independent) test set of data, **reeddata-test** (at the book's Web site). This is not typical in applications of cross-validation, where it is usually assumed that

Table 13.3. RMS and MAD errors from cross-validation for the full- and reduced-order linear models.

	Full linear model (13.9)	Reduced linear model (13.10)
RMS	0.327	0.386
MAD	0.266	0.306

all available data are used in the cross-validation. (Further, we have the luxury of only two candidate models, thus avoiding the need to appeal to multiple comparisons methods, as in Chapter 12.)

This separate test set can be used to provide an independent assessment of the models based on parameter estimates using the full set of data in **reeddata-fit**. In this way, we can create statistically independent predictions (conditional on the parameter estimates from **reeddata-fit**) and conduct a matched-pairs t-test. This provides an independent assessment of the value of cross-validation in this relatively high-noise problem (higher noise than the previous sine example, Example 13.4). Carrying out this test on the 80 measurements in **reeddata-test** gives a t-statistic of 1.68, yielding a one-sided P-value of 0.049. This provides substantial—but not overwhelming—support for the superiority of model (13.9) over (13.10), as predicted by cross-validation. ❑

13.3 THE INFORMATION MATRIX: APPLICATIONS AND RESAMPLING-BASED COMPUTATION

13.3.1 Introduction

The Fisher information matrix plays a central role in the practice and theory of estimation. This matrix provides a summary of the amount of information in the data relative to the quantities of interest. Some of the specific applications of the information matrix include confidence region calculation for parameter estimates, input determination in experimental design, performance-bound determination in an adaptive system (such as a control system), model selection via some of the methods *other* than cross-validation (as mentioned at the beginning of Section 13.2), and uncertainty-bound calculation for predictions (such as with a neural network). The information matrix has several connections to the theme of this book: (i) It is the limiting covariance matrix in the asymptotic distribution of root-finding stochastic approximation when the optimal gain sequence is used (e.g., Sections 4.4 and 7.8); (ii) it provides the basis for the important *D-optimal* criterion that will be seen in the optimal design discussion of Chapter 17; and (iii) it can be computed using Monte Carlo resampling together with a technique originally developed for search and optimization (the efficient technique for estimating the Hessian matrix discussed in Section 7.8).

Subsection 13.3.2 provides some formal background on the information matrix. Subsection 13.3.3 discusses two key properties that closely connect the information matrix to the covariance matrix of general parameter estimates. This connection provides the prime rationale for applications of the information matrix in the areas of uncertainty regions for parameter estimation, experimental design, and predictive inference, as summarized in Subsection 13.3.4. Finally, Subsection 13.3.5 describes the above-mentioned resampling-based approach to approximating the matrix. The definitions and key facts here are a prerequisite to most of Chapter 17.

13.3.2 Fisher Information Matrix: Definition and Two Equivalent Forms

Consider a sequence of random vectors $\{z_1, z_2, \ldots, z_n\}$. For example, z_k may be modeled to represent the output in a multivariate version of our traditional model of the process, $z_k = \boldsymbol{h}(\theta, x_k) + v_k$ (i.e., z_k, $\boldsymbol{h}(\cdot)$, and v_k may be vectors). More generally, it may simply be assumed that the z_k are random vectors without such a model form. The Fisher information matrix can be defined once there is *some* additional structure. If inputs x_k are relevant to the problem, let us assume that they are chosen deterministically. Further, let us assume that the *general form* for the joint probability density or probability mass (or hybrid density/mass) function for the stacked vector of random output data $Z_n = [z_1^T, z_2^T, \ldots, z_n^T]^T$ is known, but that this function depends on an unknown vector θ and, if relevant, on the inputs x_k. So, for example, it might be assumed that the z_k are i.i.d. normal with mean $\mu = \mu(\theta)$ and covariance matrix $\Sigma = \Sigma(\theta)$ dependent on the unknown parameters θ (there are no inputs x_k in this i.i.d. case). Once θ is specified, the distribution for the outputs z_k is known precisely.

To define the information matrix, we need to define the *likelihood function* based on the probability density/mass function for the outputs. The likelihood function is identical to the probability density/mass function with the exception that there is a reversal of the conditioning for the arguments. The probability density/mass function is a representation of the relative frequency with which a collection of outcomes Z_n will be observed conditioned on θ. The likelihood function, on the other hand, is the probability density/mass function considering θ as a variable *conditioned* on the data Z_n.

Let the probability density/mass function for Z_n be $p_Z(\zeta|\theta)$, where ζ (zeta) is a dummy vector representing the possible outcomes for the elements in Z_n (in $p_Z(\zeta|\theta)$, the index n on Z_n is being suppressed for notational convenience). The corresponding likelihood function, say $\ell(\theta|\zeta)$, satisfies

$$\ell(\theta|\zeta) \equiv p_Z(\zeta|\theta). \tag{13.11}$$

One important application of the likelihood interpretation of the probability density/mass function is in maximum likelihood estimation (as seen in the production function example of Section 4.2). In maximum likelihood, the estimate for θ is the value maximizing the likelihood function (13.11) based on an observed collection of data $\zeta = Z_n$ (roughly speaking, the value of θ "most likely" given the observed data). In cases where the z_k are independent, one usually maximizes $\log \ell(\theta|\zeta)$ (rather than $\ell(\theta|\zeta)$) because the criterion then simplifies to an additive form (why?).[5] Taking the logarithm, of course, does not alter the maximum likelihood estimate.

[5] The basic likelihood function is a product over $k = 1, 2, \ldots, n$ of the density function for each data point because of the independence. Taking the logarithm converts this product to a sum.

With the definition of the likelihood function in (13.11), we are now in a position to define the Fisher information matrix. For convenience in the derivatives and integrals below, we will generally suppress the arguments in $\ell(\cdot)$ since the interpretation should be clear from the context. When ℓ appears as part of an integrand, then $\ell = \ell(\boldsymbol{\theta}|\boldsymbol{\zeta})$, where $\boldsymbol{\zeta}$ is the dummy vector of integration. When ℓ appears in an expectation, then $\ell = \ell(\boldsymbol{\theta}|\boldsymbol{Z}_n)$ is the *random variable* dependent on $\boldsymbol{Z}_n$. All expectations below are with respect to the data $\boldsymbol{Z}_n$ (i.e., all randomness manifests itself in the output data; unlike some of the discussion in Sections 13.1 and 13.2, the inputs $\boldsymbol{x}_i$ are not random). For example,

$$E\left(\frac{\partial \log \ell}{\partial \boldsymbol{\theta}}\middle|\boldsymbol{\theta}\right) = E\left(\frac{\partial \log \ell(\boldsymbol{\theta}\,|\,\boldsymbol{Z}_n)}{\partial \boldsymbol{\theta}}\middle|\boldsymbol{\theta}\right)$$
$$= \int \frac{\partial \log \ell(\boldsymbol{\theta}|\boldsymbol{\zeta})}{\partial \boldsymbol{\theta}} p_{\boldsymbol{Z}}(\boldsymbol{\zeta}|\boldsymbol{\theta})\, d\boldsymbol{\zeta} = \int \frac{\partial \log \ell}{\partial \boldsymbol{\theta}} \ell\, d\boldsymbol{\zeta},$$

where the integrals are over the domain for $\boldsymbol{Z}_n$. The conditioning on $\boldsymbol{\theta}$ here and elsewhere emphasizes that in some cases the value of $\boldsymbol{\theta}$ for which the information matrix is being computed will represent a random quantity (such as the estimate based on the n measurements).

With the above notational convention, the $p \times p$ Fisher information matrix $\boldsymbol{F}_n(\boldsymbol{\theta})$ for a differentiable log-likelihood function is given by

$$\boldsymbol{F}_n(\boldsymbol{\theta}) \equiv E\left(\frac{\partial \log \ell}{\partial \boldsymbol{\theta}} \cdot \frac{\partial \log \ell}{\partial \boldsymbol{\theta}^T}\middle|\boldsymbol{\theta}\right)$$
$$= \int \frac{\partial \log \ell}{\partial \boldsymbol{\theta}} \cdot \frac{\partial \log \ell}{\partial \boldsymbol{\theta}^T} \ell\, d\boldsymbol{\zeta}. \tag{13.12}$$

(Following the notational convention of Section A.1 in Appendix A, note that the argument in the expectation of (13.12) involves the product of $p \times 1$ and $1 \times p$ vectors.)

In the common case where the underlying data are assumed independent, the magnitude of $\boldsymbol{F}_n(\boldsymbol{\theta})$ grows at a rate proportional to n since $\log \ell$ represents a sum of n random terms. To see this, recall that when the z_k are independent, $\ell(\boldsymbol{\theta}|\boldsymbol{\zeta}) = p_z(\boldsymbol{\zeta}|\boldsymbol{\theta})$ is a product of the n density functions for each of the z_k. The log operator then converts this product into a sum of n terms.

The bounded quantity $\boldsymbol{F}_n(\boldsymbol{\theta})/n$ is employed as an average information matrix over all measurements. When the data depend on some inputs $\boldsymbol{x}_k$, then $\boldsymbol{F}_n(\boldsymbol{\theta})$ also depends on these (deterministic) inputs (i.e., $\boldsymbol{F}_n(\boldsymbol{\theta}) = \boldsymbol{F}_n(\boldsymbol{\theta}|\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$). For notational convenience—and since many applications depend on cases (such as i.i.d. data) where there are no varying inputs—we suppress this dependence and write $\boldsymbol{F}_n(\boldsymbol{\theta})$ for the information matrix. In

experimental design, however, this dependence on the $\boldsymbol{x}_i$ is critical, as we see in Subsection 13.3.4 and Chapter 17.

Let us now derive some important relationships that are used in establishing properties of $\boldsymbol{F}_n(\boldsymbol{\theta})$. Assume that the likelihood function is regular in the sense that standard conditions such as in Wilks (1962, pp. 408–411 and 418–419) or Bickel and Doksum (1977, pp. 126–127) hold. One of these conditions is that the set $\{\boldsymbol{\zeta}: \ell(\boldsymbol{\theta}|\boldsymbol{\zeta}) > 0\}$ does not depend on $\boldsymbol{\theta}$. A fundamental implication of the regularity for the likelihood is that the necessary interchanges of differentiation and integration below are valid (see Theorem A.3 in Appendix A). Let us begin by noting that

$$\frac{\partial \log \ell}{\partial \boldsymbol{\theta}} = \frac{1}{\ell} \frac{\partial \ell}{\partial \boldsymbol{\theta}} \quad \text{if } \ell = \ell(\boldsymbol{\theta}|\boldsymbol{\zeta}) \neq 0. \tag{13.13}$$

A key contributor to the important properties of $\boldsymbol{F}_n(\boldsymbol{\theta})$ is the fact the random vector $\partial \log \ell / \partial \boldsymbol{\theta}$ has mean zero. This follows from (13.13) and the above-mentioned interchange of derivative and integral:

$$E\left(\frac{\partial \log \ell}{\partial \boldsymbol{\theta}} \middle| \boldsymbol{\theta} \right) = \int \frac{\partial \ell}{\partial \boldsymbol{\theta}} \, d\boldsymbol{\zeta} = \frac{\partial}{\partial \boldsymbol{\theta}} \int \ell \, d\boldsymbol{\zeta} = \frac{\partial}{\partial \boldsymbol{\theta}} \int p_{\boldsymbol{Z}}(\boldsymbol{\zeta}|\boldsymbol{\theta}) \, d\boldsymbol{\zeta} = \frac{\partial(\text{constant})}{\partial \boldsymbol{\theta}} = \boldsymbol{0}, \tag{13.14}$$

where the constant in the last derivative expression of (13.14) is unity. One interesting consequence of the mean-zero result in (13.14), as applied to (13.12), is that $\boldsymbol{F}_n(\boldsymbol{\theta})$ is the *covariance matrix* of the random vector $\partial \log \ell / \partial \boldsymbol{\theta}$. Except for relatively simple problems, however, the form in (13.12) is generally not useful in the practical calculation of the information matrix. Computing the expectation of the indicated vector product of multivariate nonlinear functions is usually a hopeless task.

Fortunately, there is an expression equivalent to (13.12) that is more amenable to computation. Suppose that $\log \ell$ is twice differentiable in $\boldsymbol{\theta}$. That is, the Hessian matrix

$$\boldsymbol{H}_{\log \ell}(\boldsymbol{\theta}; \boldsymbol{\zeta}) \equiv \frac{\partial^2 \log \ell(\boldsymbol{\theta}|\boldsymbol{\zeta})}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^T}$$

exists. (The subscript $\log \ell$ is included on $\boldsymbol{H}$ to distinguish this Hessian matrix from the generic Hessian of the loss function $L = L(\boldsymbol{\theta})$, denoted $\boldsymbol{H}(\boldsymbol{\theta}) = \partial^2 L / \partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^T$, which appeared a number of times in Chapters 4–7.) Let us now derive the Hessian-based form. Differentiating (13.14), of course, yields zero. Hence,

$$\begin{aligned}\frac{\partial}{\partial\boldsymbol{\theta}}E\left(\frac{\partial\log\ell}{\partial\boldsymbol{\theta}^T}\middle|\boldsymbol{\theta}\right) &= \frac{\partial}{\partial\boldsymbol{\theta}}\int\frac{\partial\log\ell}{\partial\boldsymbol{\theta}^T}\,\ell\,d\zeta \\ &= \int\frac{\partial\log^2\ell}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^T}\,\ell\,d\zeta+\int\frac{\partial\log\ell}{\partial\boldsymbol{\theta}}\cdot\frac{\partial\ell}{\partial\boldsymbol{\theta}^T}\,d\zeta \\ &= \int\boldsymbol{H}_{\log\ell}\,\ell\,d\zeta+\boldsymbol{F}_n(\boldsymbol{\theta}) \\ &= \mathbf{0},\end{aligned}$$

where the second line (once again) uses an interchange of derivative and integral and the third line follows by (13.13) applied to the definition of $\boldsymbol{F}_n(\boldsymbol{\theta})$ in (13.12). Solving for $\boldsymbol{F}_n(\boldsymbol{\theta})$, the information matrix is related to the Hessian matrix of $\log\ell$ through

$$\boldsymbol{F}_n(\boldsymbol{\theta}) = -E\left[\boldsymbol{H}_{\log\ell}(\boldsymbol{\theta};\boldsymbol{Z}_n)\middle|\boldsymbol{\theta}\right]. \tag{13.15}$$

The form in (13.15) is usually more amenable to practical calculation than the product-based form in (13.12).

Let us present two examples of the computation of $\boldsymbol{F}_n(\boldsymbol{\theta})$. The first example pertains to a case where the scalar outcomes z_i are discrete. The second example is a signal-plus-noise problem.

Example 13.6—Information number with a Poisson distribution. Suppose that i.i.d. scalar measurements z_i are collected where the z_i can take on integer values $\{0, 1, 2,\ldots\}$. These data are modeled as coming from a Poisson distribution, a common assumption for the distribution of the number of events that occur in a given period of time (as heavily used, say, in queuing theory for transportation and communications systems). The joint probability mass function for the n measurements is

$$p_{\boldsymbol{Z}}(\boldsymbol{Z}_n\,|\,\theta) = \prod_{i=1}^{n}\frac{e^{-\theta}\theta^{z_i}}{z_i!},\ \theta>0,$$

where, for convenience, we are evaluating the mass function at the random measurements $\{z_i\}$ instead of the "dummy" vector $\boldsymbol{\zeta}$. It is known that $E(z_i) = \mathrm{var}(z_i) = \theta$ (Bickel and Doksum, 1977, p. 456). From the mass function,

$$\log\ell(\theta\,|\,\boldsymbol{Z}_n) = -n\theta+\sum_{i=1}^{n}[z_i\log\theta-\log(z_i!)],$$

$$\frac{\partial\log\ell}{\partial\theta} = -n+\frac{1}{\theta}\sum_{i=1}^{n}z_i\,.$$

So, by (13.12) and (13.14),

$$F_n(\theta) = \text{var}\left(\frac{\partial \log \ell}{\partial \theta}\right) = \text{var}\left(\frac{1}{\theta}\sum_{i=1}^{n} z_i\right) = \frac{1}{\theta^2} n\theta = \frac{n}{\theta}.$$

The same result applies, of course, if the second-derivative-based form in (13.15) is used:

$$F_n(\theta) = -E\left(\frac{\partial^2 \log \ell}{\partial \theta^2}\right) = E\left(\frac{1}{\theta^2}\sum_{i=1}^{n} z_i\right) = \frac{1}{\theta^2} n\theta = \frac{n}{\theta}.$$

While there is little difference in the effort required to derive the product-based or second-derivative-based forms for the information number in this simple example, we will see in the next example a more typical case where the second-derivative (Hessian)-based form is easier to compute. ❑

Example 13.7—Information matrix in a signal-plus-noise problem. Suppose independent, nonidentically distributed scalar data z_i are collected with $z_i \sim N(\mu, \sigma^2 + q_i)$ where $\theta = [\mu, \sigma^2]^T$ and the q_i are known parameters. Such a framework arises when each z_i represents an independent measurement of a signal (with distribution $N(\mu, \sigma^2)$) that is obscured by independent, nonidentically distributed noise (having distribution $N(0, q_i)$). The nonidentically distributed noise arises in settings where some measurements are of better quality than other measurements (due perhaps to the varying orientation of the measurement apparatus in a series of signal measurements). Examples of application for this statistical model include parameter estimation for the state-space model in Kalman filtering (Shumway et al., 1981; Sun, 1982) and dose response analysis (Hui and Berger, 1983). The probability density for the sequence of measurements is given by

$$p_Z(\boldsymbol{Z}_n \mid \theta) = \left\{\prod_{i=1}^{n}\left[2\pi(\sigma^2 + q_i)\right]\right\}^{-1/2} \exp\left[-\frac{1}{2}\sum_{i=1}^{n}\frac{(z_i - \mu)^2}{\sigma^2 + q_i}\right],$$

where, as in Example 13.6, the random measurements $\{z_i\}$ are replacing the dummy variables (elements of ζ) in the density function.

The density above leads to the log-likelihood function

$$\log \ell(\theta \mid \boldsymbol{Z}_n) = -\frac{n\log(2\pi)}{2} - \frac{1}{2}\sum_{i=1}^{n}\log(\sigma^2 + q_i) - \frac{1}{2}\sum_{i=1}^{n}\frac{(z_i - \mu)^2}{\sigma^2 + q_i}.$$

The derivatives with respect to the two elements of θ are

$$\frac{\partial \log \ell}{\partial \mu} = \sum_{i=1}^{n} \frac{z_i - \mu}{\sigma^2 + q_i},$$

$$\frac{\partial \log \ell}{\partial (\sigma^2)} = -\frac{1}{2} \sum_{i=1}^{n} \frac{1}{\sigma^2 + q_i} + \frac{1}{2} \sum_{i=1}^{n} \frac{(z_i - \mu)^2}{(\sigma^2 + q_i)^2}.$$

The above lead to the following second derivatives appearing in the 2×2 Hessian matrix:

$$\frac{\partial^2 \log \ell}{\partial \mu^2} = -\sum_{i=1}^{n} \frac{1}{\sigma^2 + q_i},$$

$$\frac{\partial^2 \log \ell}{\partial \mu \partial (\sigma^2)} = \frac{\partial^2 \log \ell}{\partial (\sigma^2) \partial \mu} = -\sum_{i=1}^{n} \frac{z_i - \mu}{(\sigma^2 + q_i)^2},$$

$$\frac{\partial^2 \log \ell}{\partial (\sigma^2)^2} = \frac{1}{2} \sum_{i=1}^{n} \frac{1}{(\sigma^2 + q_i)^2} - \sum_{i=1}^{n} \frac{(z_i - \mu)^2}{(\sigma^2 + q_i)^3}.$$

Hence,

$$\boldsymbol{F}_n(\boldsymbol{\theta}) = -E\left[\boldsymbol{H}_{\log \ell}(\boldsymbol{\theta}; \boldsymbol{Z}_n) \middle| \boldsymbol{\theta}\right] = \begin{bmatrix} \sum_{i=1}^{n} (\sigma^2 + q_i)^{-1} & 0 \\ 0 & \frac{1}{2} \sum_{i=1}^{n} (\sigma^2 + q_i)^{-2} \end{bmatrix}. \quad (13.16)$$

Of course, the above result could have been obtained using the equivalent vector-product definition of the information matrix in (13.12). As suggested in the discussion following (13.14), however, the vector-product form is usually more cumbersome than the Hessian-based form used here. In this example, the vector-product definition requires some messy bookkeeping in computing the (2, 2) matrix component corresponding to the mean of the product of the $\partial \log \ell / \partial (\sigma^2)$ terms. (Equivalence for a simplified version of this problem is considered in Exercise 13.10.) ❑

Despite the advantages of the Hessian-based form, the analytical calculation of $\boldsymbol{F}_n(\boldsymbol{\theta})$ is often difficult or impossible in many nonlinear problems. Obtaining the required first or second derivatives of $\log \ell$ may be a formidable task in some applications, and computing the required expectation of the generally nonlinear multivariate function is often impossible in problems of practical interest. The resampling approach described in Subsection 13.3.5 is oriented to such cases.

13.3.3 Two Key Properties of the Information Matrix: Connections to the Covariance Matrix of Parameter Estimates

The above discussion focused on the definition of the information matrix and the equivalence of two representations for the matrix (the gradient-product form and the Hessian-based form). We now discuss two of the most important analytical properties of the matrix. The primary rationale for $\boldsymbol{F}_n(\boldsymbol{\theta})$ as a measure of information about $\boldsymbol{\theta}$ within the data $\boldsymbol{Z}_n$ comes from its connection to the covariance matrix for the estimate of $\boldsymbol{\theta}$ constructed from $\boldsymbol{Z}_n$. As in Sections 13.1 and 13.2, let $\hat{\boldsymbol{\theta}}_n$ denote a generic estimate of $\boldsymbol{\theta}$ based on the n data points in $\boldsymbol{Z}_n$. That is, $\hat{\boldsymbol{\theta}}_n$ may be a batch estimate or a recursive estimate where the data pairs, $\boldsymbol{x}_i$, z_i, are processed one at a time.

The first of the key properties makes the connection to the covariance matrix of an estimate via an asymptotic normality result. For some common forms of estimates $\hat{\boldsymbol{\theta}}_n$ (e.g., maximum likelihood and Bayesian maximum a posteriori), it is known that, under modest conditions,

$$\sqrt{n}(\hat{\boldsymbol{\theta}}_n - \boldsymbol{\theta}^*) \xrightarrow{\text{dist}} N(\boldsymbol{0}, \bar{\boldsymbol{F}}^{-1}) \tag{13.17}$$

where $\xrightarrow{\text{dist}}$ denotes convergence in distribution (Appendix C, Subsection C.2.4) and

$$\bar{\boldsymbol{F}} \equiv \lim_{n\to\infty} \frac{\boldsymbol{F}_n(\boldsymbol{\theta}^*)}{n}$$

provided that the indicated limit exists and is invertible (e.g., Hoadley, 1971; Rao, 1973, pp. 415–417). Hence, in practice, for n reasonably large, $\hat{\boldsymbol{\theta}}_n$ is approximately $N(\boldsymbol{\theta}, \boldsymbol{F}_n(\boldsymbol{\theta})^{-1})$ distributed when $\boldsymbol{\theta}$ is chosen close to the unknown $\boldsymbol{\theta}^*$. Because $\hat{\boldsymbol{\theta}}_n$ is generally convergent to $\boldsymbol{\theta}^*$ in some stochastic sense under the conditions in which (13.17) holds, $\boldsymbol{\theta}$ is usually chosen to be $\hat{\boldsymbol{\theta}}_n$ for the evaluation of $\boldsymbol{F}_n(\boldsymbol{\theta})$.

Let us comment on the special case where $\hat{\boldsymbol{\theta}}_n$ is a recursive estimate, particularly the gradient-based stochastic approximation (SA) algorithm discussed in Chapter 5. Recall that SA includes popular algorithms such as least-mean-squares (LMS) (Sections 3.2, 3.3, and 5.1) and neural network backpropagation (Section 5.2) as special cases. From Section 5.1, the stochastic gradient interpretation of root-finding SA is

$$\hat{\boldsymbol{\theta}}_{k+1} = \hat{\boldsymbol{\theta}}_k - a_k \boldsymbol{Y}_k(\hat{\boldsymbol{\theta}}_k), \quad k = 0, 1, \ldots, n-1,$$

where $\boldsymbol{Y}_k(\hat{\boldsymbol{\theta}}_k)$ is an unbiased measurement of the gradient $\boldsymbol{g}(\boldsymbol{\theta}) = \partial L/\partial \boldsymbol{\theta}$ evaluated at $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}_k$. Then, as shown, for example, in Kushner and Yang (1995)

and Kushner and Yin (1997, pp. 332–333), (13.17) holds for the SA recursion above if one of the following hold: (i) The optimal matrix gain sequence $\boldsymbol{a}_k = \boldsymbol{H}(\boldsymbol{\theta}^*)^{-1}/(k+1)$ is used where $\boldsymbol{H}(\boldsymbol{\theta}) = \partial^2 L/\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^T$ (Section 4.4); (ii) an adaptive gain of the form $\boldsymbol{a}_k = \boldsymbol{H}_k^{-1}/(k+1)$ is used where $\boldsymbol{H}_k \to \boldsymbol{H}(\boldsymbol{\theta}^*)$ a.s. as $k \to \infty$ (Sections 4.5 and 7.8); or (iii) iterate averaging is used based on a scalar gain $a_k = a/(k+1)^\alpha$ for $\alpha < 1$ (Subsection 4.5.3). In practice, only options (ii) and (iii) are generally feasible since one will rarely know $\boldsymbol{H}(\boldsymbol{\theta}^*)$.

Returning to the general estimation context (not restricted to the recursive form in the preceding paragraph), let us present the second key property of the information matrix. This property applies in finite samples. If $\hat{\boldsymbol{\theta}}_n$ is *any unbiased* estimator of $\boldsymbol{\theta}$ (not just one for which (13.17) holds),

$$\text{cov}(\hat{\boldsymbol{\theta}}_n) \geq \boldsymbol{F}_n(\boldsymbol{\theta}^*)^{-1} \text{ for all } n, \tag{13.18}$$

where the inequality is in the matrix sense (Appendix A: Section A.2). There is also an expression analogous to (13.18) for biased estimators, but it is not especially useful in practice because it requires knowledge of the gradient of the bias with respect to $\boldsymbol{\theta}$ (Rao, 1973, pp. 323–327; Bickel and Doksum, 1977, pp. 127–128). Expression (13.18) is often referred to as the *Cramér–Rao inequality*, but priority of discovery is now given to French mathematician M. Frêchet (Bickel and Doksum, 1977, p. 142).

Expressions (13.17) and (13.18) point to the close connection between the inverse Fisher information matrix and the covariance matrix of the estimator. While (13.17) is an asymptotic result, (13.18) applies for all sample sizes subject to the unbiasedness requirement. It is also clear why the name *information matrix* is used for $\boldsymbol{F}_n(\boldsymbol{\theta})$: A larger $\boldsymbol{F}_n(\boldsymbol{\theta})$ (in the matrix sense) is associated with a smaller covariance matrix (i.e., more information), while a smaller $\boldsymbol{F}_n(\boldsymbol{\theta})$ is associated with a larger covariance matrix (i.e., less information). In particular, suppose that $\tilde{\boldsymbol{F}} \geq \tilde{\tilde{\boldsymbol{F}}} > \boldsymbol{0}$ for two information matrices $\tilde{\boldsymbol{F}}$ and $\tilde{\tilde{\boldsymbol{F}}}$ (e.g., perhaps $\tilde{\boldsymbol{F}}$ and $\tilde{\tilde{\boldsymbol{F}}}$ are associated with different sample sizes). Then by Appendix A (Section A.2), $\tilde{\boldsymbol{F}}^{-1} \leq \tilde{\tilde{\boldsymbol{F}}}^{-1}$, which by the covariance interpretation in (13.17) and (13.18) suggests less uncertainty in the parameter estimate for $\boldsymbol{\theta}$ under the scenario producing $\tilde{\boldsymbol{F}}$.

13.3.4 Selected Applications

This section is composed of several short discussions of applications of the information matrix. Some areas not discussed, where the information matrix has also been prominently used, include model selection via some of the techniques *other* than cross-validation (Section 13.2) and the determination of *noninformative* prior distributions for Bayesian analysis. Noninformative priors allow the use of the famous Bayes' rule in inference and estimation, but are intended to be "neutral" (i.e., noninformative) relative to the data.

Uncertainty Bounds and Hypothesis Tests

This application is based on the fundamental asymptotic normality result (13.17). The asymptotic normality provides an approximate distribution for $\hat{\boldsymbol{\theta}}_n$ given that n is sufficiently large. This distribution, in turn, can be used to characterize the uncertainty in $\hat{\boldsymbol{\theta}}_n$ and test hypotheses about specific values of $\boldsymbol{\theta}$.

For example, in a hypothesis testing context, suppose that we are testing a null hypothesis of $\boldsymbol{\theta} = \bar{\boldsymbol{\theta}}$ against the alternative hypothesis $\boldsymbol{\theta} \neq \bar{\boldsymbol{\theta}}$ for some nominal value $\bar{\boldsymbol{\theta}}$. In testing such a hypothesis, we treat $\bar{\boldsymbol{\theta}}$ as the true value of $\boldsymbol{\theta}$, and test whether the observed $\hat{\boldsymbol{\theta}}_n$ is in or out of some specified acceptance region. Using (13.17), the hypothesis test is based on the assumption that

$$\hat{\boldsymbol{\theta}}_n \overset{\text{approx.}}{\sim} N\left(\bar{\boldsymbol{\theta}}, \boldsymbol{F}_n(\bar{\boldsymbol{\theta}})^{-1}\right)$$

for the finite n of practical interest. Because of the difficulties in interpreting simultaneous intervals for the multiple components of $\boldsymbol{\theta}$, it is common to map the difference $\hat{\boldsymbol{\theta}}_n - \bar{\boldsymbol{\theta}}$ into a scalar via the following inner product form of test statistic:

$$(\hat{\boldsymbol{\theta}}_n - \bar{\boldsymbol{\theta}})^T \boldsymbol{F}_n(\bar{\boldsymbol{\theta}})(\hat{\boldsymbol{\theta}}_n - \bar{\boldsymbol{\theta}}).$$

Based on the approximate normality for $\hat{\boldsymbol{\theta}}_n$, the above test statistic is, under the null hypothesis, approximately chi-squared distributed with p degrees of freedom. (Note that the test statistic represents an approximate sum of p squared $N(0, 1)$ random variables; see Exercise 13.14.).

Hence, $\hat{\boldsymbol{\theta}}_n$ provides evidence to *reject* the null hypothesis that $\boldsymbol{\theta} = \bar{\boldsymbol{\theta}}$ if the P-value associated with the above test statistic is small. This P-value is small if $\hat{\boldsymbol{\theta}}_n - \bar{\boldsymbol{\theta}}$ is large in magnitude, where *large* here is relative to the approximate covariance matrix given by the inverse information matrix (so the normalization in the inner product of the test statistic is the inverse of the inverse information matrix—the matrix itself). The P-value is computed based on the above-mentioned chi-squared distribution.

Choice of Optimal Input Values

Recall that the input and output vectors may be modeled according to $\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{\theta}, \boldsymbol{x}_k) + \boldsymbol{v}_k$. A problem in experimental design for control and other applications is to choose the inputs $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ so as to maximize the useful information in the data $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_n$. Much can (and has!) been said about this problem when the underlying model is linear in $\boldsymbol{\theta}$. In the context of nonlinear models (such as neural networks), significantly less is known.

The field of optimal experimental design is devoted to the question of picking the best inputs. Obviously, for this question to make sense, we must

formally define what is meant by *best*. Intuitively, the aim is to maximize the information in the data with respect to the estimation of $\boldsymbol{\theta}$ under the assumed model form. In this way, we use the resources devoted to data collection in the most efficient way. This aim, however, is still too vague. Fortunately, the information matrix provides a formal means of measuring information in the data relative to $\boldsymbol{\theta}$.

Recall that, for convenience, we suppressed the inputs in writing $\boldsymbol{F}_n(\boldsymbol{\theta})$ above. More completely, the information matrix is $\boldsymbol{F}_n(\boldsymbol{\theta}) = \boldsymbol{F}_n(\boldsymbol{\theta}|\boldsymbol{x}_1, \boldsymbol{x}_2,\ldots, \boldsymbol{x}_n)$. Hence, picking the n inputs to maximize $\boldsymbol{F}_n(\boldsymbol{\theta}|\boldsymbol{x}_1, \boldsymbol{x}_2,\ldots, \boldsymbol{x}_n)$ in some sense is a means of providing the data with the most information about $\boldsymbol{\theta}$. Because $\boldsymbol{F}_n(\boldsymbol{\theta}|\boldsymbol{x}_1, \boldsymbol{x}_2,\ldots, \boldsymbol{x}_n)$ is a *matrix*, there is no unique notion of maximum. For this reason, the most popular criterion in optimal design is the determinant of the information matrix, $\det[\boldsymbol{F}_n(\boldsymbol{\theta}|\boldsymbol{x}_1, \boldsymbol{x}_2,\ldots, \boldsymbol{x}_n)]$. The determinant is a standard measure of the size of a matrix and reduces the information matrix to a scalar criterion that can be uniquely maximized. In the field of experimental design, this is the famous *D-optimal criterion* (*D* for determinant).

There are other criteria based on the information matrix, but the *D*-optimal criterion is the most popular. Chapter 17 considers the subject of optimal experimental design—including *D*-optimality—in much greater detail.

Prediction Intervals for Neural Networks and Other Function Approximators

Related to the problem of putting uncertainty bounds on the estimate of $\boldsymbol{\theta}$ (as discussed above) is the problem of putting some type of probabilistic bounds on the accuracy of a prediction coming from nonlinear models such as neural networks (NNs). Suppose, for convenience, that z_k (and $h(\boldsymbol{\theta}, \boldsymbol{x}_k)$, of course) are scalar (the ideas here also apply in the multivariate case). For discussion purposes, suppose that $h(\boldsymbol{\theta}, \boldsymbol{x})$ represents a NN output based on weight parameters $\boldsymbol{\theta}$. The essential problem here is that one uses a set of fitting data to estimate the NN weights, and then one wants to use a trained NN to make predictions about the outcomes for new input values. However, since there is inevitable error in the weight estimates, the predictions will also be in error to some extent. As discussed in Chryssolouris et al. (1996) and Hwang and Ding (1997), the information matrix is valuable for finding the prediction bounds.

Suppose that the data vector $\boldsymbol{Z}_n$ has a known distributional form with unknown parameters. Commonly, $\boldsymbol{Z}_n$ is assumed multivariate normal, with the mean for each z_k being $h(\boldsymbol{\theta}, \boldsymbol{x}_k)$ (i.e., v_k has mean zero). When $h(\boldsymbol{\theta}, \boldsymbol{x})$ is differentiable in $\boldsymbol{\theta}$ for an $\boldsymbol{x}$ of interest and the asymptotic normality in (13.17) holds, then

$$\sqrt{n}\left[h(\hat{\boldsymbol{\theta}}_n, \boldsymbol{x}) - h(\boldsymbol{\theta}^*, \boldsymbol{x})\right] \xrightarrow{\text{dist}} N\left(\boldsymbol{0}, h'(\boldsymbol{\theta}^*, \boldsymbol{x})^T \bar{\boldsymbol{F}}^{-1} h'(\boldsymbol{\theta}^*, \boldsymbol{x})\right),$$

where $h'(\cdot)$ denotes the gradient of $h(\cdot)$ with respect to $\boldsymbol{\theta}$. Hence, the NN prediction satisfies

$$h(\hat{\boldsymbol{\theta}}_n, \boldsymbol{x}) \overset{\text{approx.}}{\sim} N\left(h(\boldsymbol{\theta}, \boldsymbol{x}),\ h'(\boldsymbol{\theta}, \boldsymbol{x})^T \boldsymbol{F}_n(\boldsymbol{\theta})^{-1} h'(\boldsymbol{\theta}, \boldsymbol{x})\right) \tag{13.19}$$

for $\boldsymbol{\theta}$ close to $\boldsymbol{\theta}^*$ when n is reasonably large. In practice, $\boldsymbol{\theta}$ is often set to $\hat{\boldsymbol{\theta}}_n$ in the mean and variance expressions on the right-hand side of (13.19). Thus, the prediction $h(\hat{\boldsymbol{\theta}}_n, \boldsymbol{x})$ has an uncertainty given by a normal distribution with an approximate variance given by the variance in (13.19) evaluated at $\hat{\boldsymbol{\theta}}_n$. This uncertainty provides some sense of how much $h(\hat{\boldsymbol{\theta}}_n, \boldsymbol{x})$ is likely to differ from $E(z|\boldsymbol{x})$ when $h(\cdot)$ is such that $E(z|\boldsymbol{x}) \approx h(\boldsymbol{\theta}^*, \boldsymbol{x})$.

As opposed to the prediction error $E(z|\boldsymbol{x}) - h(\hat{\boldsymbol{\theta}}_n, \boldsymbol{x})$, the approach above can also be used to form an approximate distribution for the actual *observation* error, namely $z - h(\hat{\boldsymbol{\theta}}_n, \boldsymbol{x})$, where $z = z(\boldsymbol{x})$ is some future measurement. Suppose that the true process and model are both of the form $z_k = h(\boldsymbol{\theta}, \boldsymbol{x}_k) + v_k$, where v_k is i.i.d. $N(0, \sigma^2)$ with a reliable estimate of σ^2 available. Then, from (13.19), $z - h(\hat{\boldsymbol{\theta}}_n, \boldsymbol{x})$ is approximately normally distributed with mean zero and variance σ^2 plus the $O(1/n)$ variance appearing in (13.19).

13.3.5 Resampling-Based Calculation of the Information Matrix[6]

The calculation of $\boldsymbol{F}_n(\boldsymbol{\theta})$ is often difficult or impossible in many nonlinear problems. Obtaining the required first or second derivatives of the log-likelihood function may be a formidable task in some applications, and computing the required expectation of the generally nonlinear multivariate function is often impossible in problems of practical interest. To address this difficulty, this subsection outlines a computer resampling approach to estimating $\boldsymbol{F}_n(\boldsymbol{\theta})$. This approach is useful when analytical methods for computing $\boldsymbol{F}_n(\boldsymbol{\theta})$ are infeasible. The approach makes use of an idea introduced for optimization—the Hessian estimation for stochastic approximation introduced in Section 7.8—even though this problem is not directly one of optimization.

The basis for the technique below is to use computational horsepower in lieu of traditional detailed theoretical analysis to determine $\boldsymbol{F}_n(\boldsymbol{\theta})$. The method here is an example of a Monte Carlo-based method for producing an estimate. Such methods have become very popular as a means of handling problems that were formerly infeasible. Two other notable Monte Carlo techniques are the bootstrap method for determining statistical distributions of estimates (e.g., Efron and Tibshirani, 1986; Lunneborg, 2000) and the Markov chain Monte Carlo method for producing pseudorandom numbers and related quantities, considered in Chapter 16. Part of the appeal of the Monte Carlo method here for estimating $\boldsymbol{F}_n(\boldsymbol{\theta})$ is that it can be implemented with only evaluations of the log-likelihood (typically much easier to obtain than the customary gradient or second derivative information). Alternatively, if the gradient of the log-likelihood is available, that information can be used to enhance performance.

---

[6]This subsection may be omitted at first reading. It provides a Monte Carlo method for estimating the information matrix when it is not possible to obtain an analytical solution.

The essence of the method is to produce a large number of efficient "almost unbiased" estimates of the Hessian matrix of $\log \ell(\cdot)$ and then average the negative of these estimates to obtain an approximation to $\boldsymbol{F}_n(\boldsymbol{\theta})$. This approach is directly motivated by the definition of $\boldsymbol{F}_n(\boldsymbol{\theta})$ as the mean value of the negative Hessian matrix (eqn. (13.15)). To produce these estimates, we generate *pseudodata vectors* in a Monte Carlo manner analogous to the bootstrap method mentioned above. The pseudodata are generated according to a bootstrap resampling scheme treating the chosen $\boldsymbol{\theta}$ as "truth." The pseudodata are generated according to the probability model (13.11). So, for example, if the real data $\boldsymbol{Z}_n = [\boldsymbol{z}_1^T, \boldsymbol{z}_2^T, ..., \boldsymbol{z}_n^T]^T$ are assumed to be jointly normally distributed, $N(\boldsymbol{\mu}(\boldsymbol{\theta}), \boldsymbol{\Sigma}(\boldsymbol{\theta}))$, then the pseudodata are generated by Monte Carlo according to a normal distribution based on a mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ evaluated at the chosen $\boldsymbol{\theta}$.

In particular, let the ith pseudodata vector be $\boldsymbol{Z}_{\text{pseudo}}(i)$, where $\dim(\boldsymbol{Z}_{\text{pseudo}}(i)) = \dim(\boldsymbol{Z}_n)$ (some multiple of n corresponding to the dimension of the $\boldsymbol{z}_i$). This pseudodata vector represents a sample of size n (analogous to the real data $\boldsymbol{Z}_n$). The use of the notation $\boldsymbol{Z}_{\text{pseudo}}$ without the argument (i) is a generic reference to a pseudodata vector. The form of the distribution used to generate $\boldsymbol{Z}_{\text{pseudo}}$ is identical to the form represented by the likelihood function (13.11); the choice of $\boldsymbol{\theta}$ in the distribution depends on the application.

Given the aim to avoid the complex calculations usually needed to obtain second derivative information, the critical part of this conceptually simple scheme is the efficient Hessian estimation. Section 7.8 introduced an efficient scheme for estimating Hessian matrices in the context of optimization. Although there is no optimization here per se, we use the same formula for Hessian estimation. This formula is based on the simultaneous perturbation principle.

The approach given below can work with either $\log \ell(\boldsymbol{\theta} | \boldsymbol{Z}_{\text{pseudo}})$ values (alone) or with the gradient $\boldsymbol{g}(\boldsymbol{\theta} | \boldsymbol{Z}_{\text{pseudo}}) \equiv \partial \log \ell(\boldsymbol{\theta} | \boldsymbol{Z}_{\text{pseudo}}) / \partial \boldsymbol{\theta}$ if that is available. The former usually corresponds to cases where the likelihood function and associated nonlinear process are so complex that no gradients are available. The latter allows for the fact that sometimes the gradient is available in even relatively complex problems and that such information should be used to enhance the estimation process if available. To highlight the fundamental commonality of approach, we let $\boldsymbol{G}(\boldsymbol{\theta} | \boldsymbol{Z}_{\text{pseudo}})$ represent either a gradient *approximation* (based on $\log \ell(\boldsymbol{\theta} | \boldsymbol{Z}_{\text{pseudo}})$ values) or the exact gradient $\boldsymbol{g}(\boldsymbol{\theta} | \boldsymbol{Z}_{\text{pseudo}})$. Because of its efficiency, the simultaneous perturbation gradient approximation is recommended in the case where only $\log \ell(\boldsymbol{\theta} | \boldsymbol{Z}_{\text{pseudo}})$ values are available (see Section 7.8).

We now present the Hessian estimate. Let $\hat{\boldsymbol{H}}_k$ denote the kth estimate of the Hessian $\boldsymbol{H}_{\log \ell}(\cdot)$. The estimate here differs slightly from that in Section 7.8 with the decaying c_k sequence because there is no iterating towards a solution in the optimization sense (hence no need for an explicit c_k sequence). The Hessian estimate is

$$\hat{\boldsymbol{H}}_k = \frac{1}{2}\left\{\frac{\delta \boldsymbol{G}_k}{2}\left[\Delta_{k1}^{-1}, \Delta_{k2}^{-1}, \ldots, \Delta_{kp}^{-1}\right] + \left(\frac{\delta \boldsymbol{G}_k}{2}\left[\Delta_{k1}^{-1}, \Delta_{k2}^{-1}, \ldots, \Delta_{kp}^{-1}\right]\right)^T\right\}, \quad (13.20)$$

where $\delta \boldsymbol{G}_k = \boldsymbol{G}(\boldsymbol{\theta} + \Delta_k \mid \boldsymbol{Z}_{\text{pseudo}}) - \boldsymbol{G}(\boldsymbol{\theta} - \Delta_k \mid \boldsymbol{Z}_{\text{pseudo}})$ and the vector $\Delta_k \equiv [\Delta_{k1}, \Delta_{k2}, \ldots, \Delta_{kp}]^T$ is a mean-zero random perturbation such that the $\{\Delta_{kj}\}$ are "small" symmetrically distributed random variables that for all k, j are independent, identically distributed, uniformly bounded, and satisfy $E(|1/\Delta_{kj}|) < \infty$ uniformly in k, j. The latter condition *excludes* such commonly used Monte Carlo distributions as uniform and Gaussian. Assume that $|\Delta_{kj}| \leq c$ for some small $c > 0$. Note that the user has full control over the choice of the Δ_{kj} distribution. A valid (and simple) choice is the Bernoulli $\pm c$ distribution (it is not known at this time if this is the "best" distribution to choose). To illustrate how the *individual* Hessian estimates may be quite poor, note that $\hat{\boldsymbol{H}}_k$ in (13.20) has (at most) rank two (and may not be positive semidefinite). This low quality, however, does not prevent the information matrix estimate from being accurate. The averaging process eliminates inadequacies in the Hessian estimates.

Given the form for the Hessian estimate in (13.20), it is now relatively straightforward to estimate $\boldsymbol{F}_n(\boldsymbol{\theta})$. From the results in Section 7.8, the Hessian estimate has an $O(c^2)$ bias. That is, $E(\hat{\boldsymbol{H}}_k \mid \boldsymbol{Z}_{\text{pseudo}}) = \boldsymbol{H}_{\log \ell}(\boldsymbol{\theta}; \boldsymbol{Z}_{\text{pseudo}}) + O(c^2)$. Hence, averaging many $\hat{\boldsymbol{H}}_k$ values yields an estimate of

$$E[\boldsymbol{H}_{\log \ell}(\boldsymbol{\theta}; \boldsymbol{Z}_{\text{pseudo}})] = -\boldsymbol{F}_n(\boldsymbol{\theta})$$

to within an $O(c^2)$ bias (the expectation in the left-hand side above is with respect to the pseudodata). The resulting estimate can be made as accurate as desired through reducing c and increasing the number of $\hat{\boldsymbol{H}}_k$ values being averaged. The averaging of the $\hat{\boldsymbol{H}}_k$ values may be done recursively (as in Section 7.8) to avoid having to store many matrices.

Let us now present a step-by-step summary of the above Monte Carlo resampling approach for estimating $\boldsymbol{F}_n(\boldsymbol{\theta})$. A numerical example is given in Spall (1998); this example is an extension of the signal-plus-noise problem in Example 13.7 to the setting where the data (z_i) are multivariate.

Monte Carlo Resampling Method for Estimating $\boldsymbol{F}_n(\boldsymbol{\theta})$

Step 0 **(Initialization)** Determine $\boldsymbol{\theta}$ and the number of pseudodata vectors that will be generated (N). Determine whether log-likelihood $\log \ell(\cdot)$ or gradient information $\boldsymbol{g}(\cdot)$ will be used to form the $\hat{\boldsymbol{H}}_k$ estimates. Pick the small number c in the Bernoulli $\pm c$ distribution used to generate the perturbations Δ_{kj}; $c = 0.0001$ has been effective in the author's experience (non-Bernoulli distributions may also be used subject to the conditions mentioned below (13.20)). Set $i = 1$.

Step 1 **(Generating pseudodata)** Based on $\boldsymbol{\theta}$ given in step 0, generate by Monte Carlo the ith pseudodata vector $\boldsymbol{Z}_{\text{pseudo}}(i)$.

Step 2 **(Hessian estimation)** With the pseudodata vector in step 1, use the Hessian estimation formula in (13.20) to determine one or more $\hat{\boldsymbol{H}}_k$. (Forming an average of more than one $\hat{\boldsymbol{H}}_k$ is useful at each pseudodata vector if the vectors are relatively expensive to generate.)

Step 3 **(Averaging Hessian estimates)** Repeat steps 1 and 2 a large number of times (N). Take the negative of the average of the N Hessian estimates produced in step 2; this is the estimate of $\boldsymbol{F}_n(\boldsymbol{\theta})$. It is usually convenient to use the standard recursive representation of a sample mean to avoid the storage of the N Hessian estimates.

13.4 CONCLUDING REMARKS

It was largely assumed in Chapters 1–12 that there was some given model, as reflected in the choice of the parameters to estimate ($\boldsymbol{\theta}$) and the associated loss or root-finding function. We considered a number of search and optimization methods for estimating $\boldsymbol{\theta}$. This chapter, on the other hand, has looked at some issues that are relevant *before* and *after* the estimation of $\boldsymbol{\theta}$.

In particular, this chapter discussed the bias–variance tradeoff in fitting a model to a set of data, the choice of the "best" structure for a model in light of the bias–variance tradeoff, and the Fisher information matrix as a summary description of the amount of information in a set of data relative to a given model form. Among many other areas for application, these topics arise in constructing simulation models or in building open-loop models for use with control systems. Some of these modeling issues are typically encountered prior to the application of a search and optimization method. In particular, one must select the form of the model before it is possible to estimate the parameters of the model!

The bias–variance tradeoff is enlightening largely for the conceptual understanding it provides. The tradeoff has little direct utility in assessing the adequacy of candidate models because it depends on information that is generally unavailable (such as the true probability distribution of the data). One consequence of the tradeoff is that no single type of model can be universally superior. Models that are very flexible (have many parameters) are valuable in certain complex problems but will hew too closely to the nuances of specific data sets in problems with simple input–output relationships. Application of a complex model to a fundamentally simple problem results in a model with a variance that is too high. On the other hand, a simple model without sufficient flexibility will suffer in problems with complex (highly nonlinear) input–output relationships. In particular, models that are too simple create excess bias, as the details of the input–output relationship are smoothed over.

As a practical means of capturing the essence of the bias–variance tradeoff, we discussed model selection methods, focusing on the popular cross-validation approach. Cross-validation is based on the commonsense idea of

repeatedly partitioning an existing set of data into fitting and test subsets, choosing the best model as the one that provides the best average fit over all test subsets. This method has been used in a wide variety of applications, as it is simple to implement and imposes minimal assumptions on the problem structure. The method's generality, of course, means that in certain problems, other approaches that exploit specific problem structure may work better. Further, a downside of standard cross-validation (as discussed here) is that the complete set of data must be available in advance. Other approaches are better suited to (say) on-line applications where the model selection may be determined adaptively as data arrive.

Cross-validation is ideally suited to comparing candidate simulation models based on data from an actual physical process. Such models are often too complex for the use of methods that require simpler analytical structure (e.g., most of the other methods mentioned at the start of Section 13.2).

We concluded this chapter with a discussion of the Fisher information matrix. This matrix has wide application before *and* after the estimation of $\boldsymbol{\theta}$. Two important applications prior to estimating $\boldsymbol{\theta}$ are in model selection and experimental design. Although cross-validation as discussed in Section 13.2 is one model selection method that does not use the information matrix, some of the other methods that were mentioned *do* rely on the information matrix. In experimental design, the information matrix is used to pick the input ($\boldsymbol{x}$) values to provide the best estimate for $\boldsymbol{\theta}$. Chapter 17 considers the subject of experimental design further. One prominent application for the information matrix after $\boldsymbol{\theta}$ is estimated is approximate confidence region calculation.

While the information matrix is very useful in several applications, we also saw that it can be difficult or impossible to analytically compute in certain problems. To this end, we were able to draw on one of the methods introduced in the context of stochastic search and optimization: the simultaneous perturbation-based estimate of the Hessian matrix. By averaging a large number of these matrix estimates, we are able to generate an accurate estimate of the information matrix in problems for which no analytical solution is available. This is an example of a resampling-based estimate, where Monte Carlo sampling is used to estimate a quantity of physical interest. Another prominent example of estimation via Monte Carlo sampling is Markov chain Monte Carlo, as considered in Chapter 16.

EXERCISES

13.1 Prove the bias–variance decomposition in eqn. (13.2).

13.2 Provide a brief description or sketch of a case where the model provides a poor description of the process, but where a global MSE defined as $\overline{\text{variance}} + (\overline{\text{bias}})^2$ (instead of the form in (13.3)) is zero. This shows the importance of a proper definition of averaging with respect to the inputs $\boldsymbol{x}$.

13.3 **(a)** In the setting of Example 13.1, show that for any $0 < |\mu| < \infty$ and $0 < \sigma < \infty$, there exists an estimator of the form $r\bar{z}$, $0 < r < 1$, that produces a lower MSE than the unbiased estimator $\bar{z}$.

(b) For a fixed μ and σ satisfying the conditions of part (a), let r_n denote the value of r producing the estimator $r\bar{z}$ with the lowest MSE at a sample size n. Show that the difference $1 - r_n$ decays at rate $O(1/n)$ (i.e., $\bar{z}$ is nearly the lowest MSE estimator for large n).

13.4 Prove the bias and variance relationships in (13.4) and (13.5).

13.5 Consider the setting of Example 13.2 except that the scalar inputs $x_k = \chi_k = 2k$ (versus $x_k = \chi_k = k$), $k = 1, 2, \ldots, m$, $m = 10$. Produce a table analogous to Table 13.1 in Example 13.2 and comment on the differences in the results here and the results in Example 13.2.

13.6 With the same 30 data points (**sinedata** at the book's Web site) and cross-validation approach used in Example 13.4, compute the RMS error of a fifth-order polynomial. Show that this model is superior to the linear and tenth-order models, but slightly inferior to the cubic polynomial.

13.7 Consider the cross-validation/oboe reed problem in Example 13.5. Produce an additional column in Table 13.3 showing the RMS and MAD values for the five-input linear model

$$z = \theta_{\text{const}} + \theta_T T + \theta_A A + \theta_E E + \theta_V V + \theta_S S + v,$$

which is a model with all input variables except "first blow" (F). Among the three models being considered in this expanded Table 13.3, note that cross-validation continues to favor the full linear model (13.9).

13.8 There exists some useful special structure with leave-one-out cross-validation when applied to linear models. When the ith measurement $(\boldsymbol{h}_i, z_i)$ is excluded from the original n measurements, show that the ordinary least-squares estimate of $\boldsymbol{\theta}$ is:

$$\left[(\boldsymbol{H}_n^T \boldsymbol{H}_n)^{-1} + \frac{(\boldsymbol{H}_n^T \boldsymbol{H}_n)^{-1} \boldsymbol{h}_i \boldsymbol{h}_i^T (\boldsymbol{H}_n^T \boldsymbol{H}_n)^{-1}}{1 - \boldsymbol{h}_i^T (\boldsymbol{H}_n^T \boldsymbol{H}_n)^{-1} \boldsymbol{h}_i} \right] \left(\boldsymbol{H}_n^T \boldsymbol{Z}_n - \boldsymbol{h}_i z_i \right).$$

(The value of this result will be apparent in the next exercise.) (Hint: Begin with the basic batch least-squares formula $\hat{\boldsymbol{\theta}}^{(n)} = (\boldsymbol{H}_n^T \boldsymbol{H}_n)^{-1} \boldsymbol{H}_n^T \boldsymbol{Z}_n$ and apply the matrix inversion lemma [matrix relationship (xxii) in Appendix A] as needed.)

13.9 Using the result in Exercise 13.8, show that the difference between the prediction and actual outcome for the ith measurement in leave-one-out cross-validation is $(\boldsymbol{h}_i^T \hat{\boldsymbol{\theta}}^{(n)} - z_i) / [1 - \boldsymbol{h}_i^T (\boldsymbol{H}_n^T \boldsymbol{H}_n)^{-1} \boldsymbol{h}_i]$ for all i. (That is, the only regression estimate needed to form the overall MSE for a given model is the estimate $\hat{\boldsymbol{\theta}}^{(n)}$ from *all* the data.)

13.10 For the special case of the signal-plus-noise problem in Example 13.7 where $q_i = q$ for all i, compute $\boldsymbol{F}_n(\boldsymbol{\theta})$ using the vector-product form in (13.12).

Show the equality of this expression to the Hessian-based form in (13.16) when $q_i = q$. (Hint: If $X \sim N(0, a^2)$, then $E(X^3) = 0$ and $E(X^4) = 3a^4$.)

13.11 Suppose that i.i.d. binary data z_i satisfy $P(z_i = 0) = 1 - \theta$ and $P(z_i = 1) = \theta$. Using Fisher information, show that the sample mean $\bar{z}$ is a minimum-variance unbiased estimator for θ.

13.12 Consider data z_i that are i.i.d. according to the exponential distribution function $1 - e^{-\lambda z}$, where λ, $z > 0$. Using Fisher information, show that the sample mean $\bar{z}$ is a minimum-variance unbiased estimator of $\theta = 1/\lambda$. (Note that $E(z) = 1/\lambda$ and $\text{var}(z) = 1/\lambda^2$.)

13.13 Let $\boldsymbol{\phi} = \boldsymbol{\phi}(\boldsymbol{\theta})$ be a one-to-one, continuously differentiable transformation. Show that the information matrix for $\boldsymbol{\phi}$ is $(\partial\boldsymbol{\theta}^T/\partial\boldsymbol{\phi})\boldsymbol{F}_n(\boldsymbol{\theta})(\partial\boldsymbol{\theta}/\partial\boldsymbol{\phi}^T)$. (Note: The derivative $\partial\boldsymbol{\theta}^T/\partial\boldsymbol{\phi}$ is guaranteed to exist by the inverse function theorem; see, e.g., Apostol, 1974, p. 417.)

13.14 Provide justification for the statement in Subsection 13.3.4 that the test statistic $(\hat{\boldsymbol{\theta}}_n - \bar{\boldsymbol{\theta}})^T \boldsymbol{F}_n(\bar{\boldsymbol{\theta}})(\hat{\boldsymbol{\theta}}_n - \bar{\boldsymbol{\theta}})$ represents an approximate sum of p squared $N(0, 1)$ random variables where $\bar{\boldsymbol{\theta}}$ is some value being tested under the null hypothesis (i.e., the test statistic is approximately chi-squared distributed with p degrees of freedom).

13.15 For a statistical model of your choice, implement the Monte Carlo resampling scheme in Subsection 13.3.5 for determining the information matrix. For this model:

(a) Compute the true information matrix.

(b) Estimate the information matrix using only likelihood evaluations (no gradients).

(c) Estimate the information matrix using gradients of the log-likelihood function. Relative to truth as determined in part (a), comment on the relative accuracy of the results in parts (b) and (c).