4 *Gaussian models*

4.1 Introduction

In this chapter, we discuss the **multivariate Gaussian** or **multivariate normal** (**MVN**), which is the most widely used joint probability density function for continuous variables. It will form the basis for many of the models we will encounter in later chapters.

Unfortunately, the level of mathematics in this chapter is higher than in many other chapters. In particular, we rely heavily on linear algebra and matrix calculus. This is the price one must pay in order to deal with high-dimensional data. Beginners may choose to skip sections marked with a *. In addition, since there are so many equations in this chapter, we have put a box around those that are particularly important.

4.1.1 Notation

Let us briefly say a few words about notation. We denote vectors by boldface lower case letters, such as $\mathbf{x}$. We denote matrices by boldface upper case letters, such as $\mathbf{X}$. We denote entries in a matrix by non-bold upper case letters, such as X_{ij}.

All vectors are assumed to be column vectors unless noted otherwise. We use $[x_1, \ldots, x_D]$ to denote a column vector created by stacking D scalars. Similarly, if we write $\mathbf{x} = [\mathbf{x}_1, \ldots, \mathbf{x}_D]$, where the left hand side is a tall column vector, we mean to stack the $\mathbf{x}_i$ along the rows; this is usually written as $\mathbf{x} = (\mathbf{x}_1^T, \ldots, \mathbf{x}_D^T)^T$, but that is rather ugly. If we write $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_D]$, where the left hand side is a matrix, we mean to stack the $\mathbf{x}_i$ along the columns, creating a matrix.

4.1.2 Basics

Recall from Section 2.5.2 that the pdf for an MVN in D dimensions is defined by the following:

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \triangleq \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right] \tag{4.1}$$

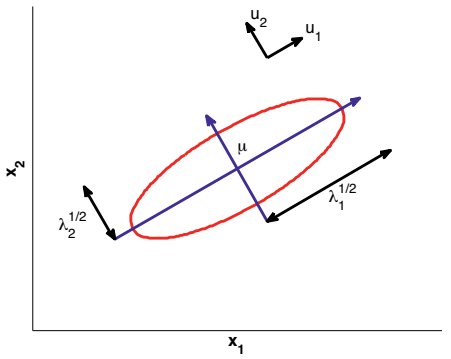


Figure 4.1 Visualization of a 2 dimensional Gaussian density. The major and minor axes of the ellipse are defined by the first two eigenvectors of the covariance matrix, namely $\mathbf{u}_1$ and $\mathbf{u}_2$. Based on Figure 2.7 of (Bishop 2006).

The expression inside the exponent is the Mahalanobis distance between a data vector $\mathbf{x}$ and the mean vector $\boldsymbol{\mu}$, We can gain a better understanding of this quantity by performing an **eigendecomposition** of $\boldsymbol{\Sigma}$. That is, we write $\boldsymbol{\Sigma} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$, where $\mathbf{U}$ is an orthonormal matrix of eigenvectors satsifying $\mathbf{U}^T\mathbf{U} = \mathbf{I}$, and $\boldsymbol{\Lambda}$ is a diagonal matrix of eigenvalues.

Using the eigendecomposition, we have that

$$\boldsymbol{\Sigma}^{-1} = \mathbf{U}^{-T}\boldsymbol{\Lambda}^{-1}\mathbf{U}^{-1} = \mathbf{U}\boldsymbol{\Lambda}^{-1}\mathbf{U}^T = \sum_{i=1}^{D}\frac{1}{\lambda_i}\mathbf{u}_i\mathbf{u}_i^T \tag{4.2}$$

where $\mathbf{u}_i$ is the i'th column of $\mathbf{U}$, containing the i'th eigenvector. Hence we can rewrite the Mahalanobis distance as follows:

$$(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}) = (\mathbf{x}-\boldsymbol{\mu})^T\left(\sum_{i=1}^{D}\frac{1}{\lambda_i}\mathbf{u}_i\mathbf{u}_i^T\right)(\mathbf{x}-\boldsymbol{\mu}) \tag{4.3}$$

$$= \sum_{i=1}^{D}\frac{1}{\lambda_i}(\mathbf{x}-\boldsymbol{\mu})^T\mathbf{u}_i\mathbf{u}_i^T(\mathbf{x}-\boldsymbol{\mu}) = \sum_{i=1}^{D}\frac{y_i^2}{\lambda_i} \tag{4.4}$$

where $y_i \triangleq \mathbf{u}_i^T(\mathbf{x}-\boldsymbol{\mu})$. Recall that the equation for an ellipse in 2d is

$$\frac{y_1^2}{\lambda_1} + \frac{y_2^2}{\lambda_2} = 1 \tag{4.5}$$

Hence we see that the contours of equal probability density of a Gaussian lie along ellipses. This is illustrated in Figure 4.1. The eigenvectors determine the orientation of the ellipse, and the eigenvalues determine how elongated it is.

In general, we see that the Mahalanobis distance corresponds to Euclidean distance in a transformed coordinate system, where we shift by $\boldsymbol{\mu}$ and rotate by $\mathbf{U}$.

4.1.3 MLE for an MVN

We now describe one way to estimate the parameters of an MVN, using MLE. In later sections, we will discuss Bayesian inference for the parameters, which can mitigate overfitting, and can provide a measure of confidence in our estimates.

Theorem 4.1.1 (MLE for a Gaussian). *If we have N iid samples $\mathbf{x}_i \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, then the MLE for the parameters is given by*

$$\hat{\boldsymbol{\mu}}_{mle} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i \triangleq \overline{\mathbf{x}} \tag{4.6}$$

$$\hat{\boldsymbol{\Sigma}}_{mle} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})^T = \frac{1}{N}(\sum_{i=1}^{N}\mathbf{x}_i\mathbf{x}_i^T) - \overline{\mathbf{x}}\,\overline{\mathbf{x}}^T \tag{4.7}$$

That is, the MLE is just the empirical mean and empirical covariance. In the univariate case, we get the following familiar results:

$$\hat{\mu} = \frac{1}{N}\sum_i x_i = \overline{x} \tag{4.8}$$

$$\hat{\sigma}^2 = \frac{1}{N}\sum_i (x_i - \overline{x})^2 = \left(\frac{1}{N}\sum_i x_i^2\right) - (\overline{x})^2 \tag{4.9}$$

4.1.3.1 Proof *

To prove this result, we will need several results from matrix algebra, which we summarize below. In the equations, $\mathbf{a}$ and $\mathbf{b}$ are vectors, and $\mathbf{A}$ and $\mathbf{B}$ are matrices. Also, the notation $\mathrm{tr}(\mathbf{A})$ refers to the **trace** of a matrix, which is the sum of its diagonals: $\mathrm{tr}(\mathbf{A}) = \sum_i A_{ii}$.

$$\begin{aligned}
\frac{\partial(\mathbf{b}^T\mathbf{a})}{\partial\mathbf{a}} &= \mathbf{b} \\
\frac{\partial(\mathbf{a}^T\mathbf{A}\mathbf{a})}{\partial\mathbf{a}} &= (\mathbf{A} + \mathbf{A}^T)\mathbf{a} \\
\frac{\partial}{\partial\mathbf{A}}\mathrm{tr}(\mathbf{B}\mathbf{A}) &= \mathbf{B}^T \\
\frac{\partial}{\partial\mathbf{A}}\log|\mathbf{A}| &= \mathbf{A}^{-T} \triangleq (\mathbf{A}^{-1})^T \\
\mathrm{tr}(\mathbf{A}\mathbf{B}\mathbf{C}) &= \mathrm{tr}(\mathbf{C}\mathbf{A}\mathbf{B}) = \mathrm{tr}(\mathbf{B}\mathbf{C}\mathbf{A})
\end{aligned} \tag{4.10}$$

The last equation is called the **cyclic permutation property** of the trace operator. Using this, we can derive the widely used **trace trick**, which reorders the scalar inner product $\mathbf{x}^T\mathbf{A}\mathbf{x}$ as follows

$$\mathbf{x}^T\mathbf{A}\mathbf{x} = \mathrm{tr}(\mathbf{x}^T\mathbf{A}\mathbf{x}) = \mathrm{tr}(\mathbf{x}\mathbf{x}^T\mathbf{A}) = \mathrm{tr}(\mathbf{A}\mathbf{x}\mathbf{x}^T) \tag{4.11}$$

Proof. We can now begin with the proof. The log-likelihood (dropping additive constants) is given by

$$\ell(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log p(\mathcal{D}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{N}{2}\log|\boldsymbol{\Lambda}| - \frac{1}{2}\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})^T\boldsymbol{\Lambda}(\mathbf{x}_i - \boldsymbol{\mu}) \tag{4.12}$$

where $\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}$ is the precision matrix.

Using the substitution $\mathbf{y}_i = \mathbf{x}_i - \boldsymbol{\mu}$ and the chain rule of calculus, we have

$$\frac{\partial}{\partial\boldsymbol{\mu}}(\mathbf{x}_i - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu}) = \frac{\partial}{\partial\mathbf{y}_i}\mathbf{y}_i^T\boldsymbol{\Sigma}^{-1}\mathbf{y}_i\frac{\partial\mathbf{y}_i}{\partial\boldsymbol{\mu}} \tag{4.13}$$

$$= -1(\boldsymbol{\Sigma}^{-1} + \boldsymbol{\Sigma}^{-T})\mathbf{y}_i \tag{4.14}$$

Hence

$$\frac{\partial}{\partial\boldsymbol{\mu}}\ell(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = -\frac{1}{2}\sum_{i=1}^{N} -2\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu}) = \boldsymbol{\Sigma}^{-1}\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu}) = 0 \tag{4.15}$$

$$\hat{\boldsymbol{\mu}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i = \overline{\mathbf{x}} \tag{4.16}$$

So the MLE of $\boldsymbol{\mu}$ is just the empirical mean.

Now we can use the trace-trick to rewrite the log-likelihood for $\boldsymbol{\Lambda}$ as follows:

$$\ell(\boldsymbol{\Lambda}) = \frac{N}{2}\log|\boldsymbol{\Lambda}| - \frac{1}{2}\sum_i \mathrm{tr}[(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T\boldsymbol{\Lambda}] \tag{4.17}$$

$$= \frac{N}{2}\log|\boldsymbol{\Lambda}| - \frac{1}{2}\mathrm{tr}\left[\mathbf{S}_\mu\boldsymbol{\Lambda}\right] \tag{4.18}$$

$$\tag{4.19}$$

where

$$\mathbf{S}_\mu \triangleq \sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T \tag{4.20}$$

is the scatter matrix centered on $\boldsymbol{\mu}$. Taking derivatives of this expression with respect to $\boldsymbol{\Lambda}$ yields

$$\frac{\partial\ell(\boldsymbol{\Lambda})}{\partial\boldsymbol{\Lambda}} = \frac{N}{2}\boldsymbol{\Lambda}^{-T} - \frac{1}{2}\mathbf{S}_\mu^T = 0 \tag{4.21}$$

$$\boldsymbol{\Lambda}^{-T} = \boldsymbol{\Lambda}^{-1} = \boldsymbol{\Sigma} = \frac{1}{N}\mathbf{S}_\mu \tag{4.22}$$

so

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T \tag{4.23}$$

which is just the empirical covariance matrix centered on $\boldsymbol{\mu}$. If we plug-in the MLE $\boldsymbol{\mu} = \overline{\mathbf{x}}$ (since both parameters must be simultaneously optimized), we get the standard equation for the MLE of a covariance matrix. □

4.1.4 Maximum entropy derivation of the Gaussian *

In this section, we show that the multivariate Gaussian is the distribution with maximum entropy subject to having a specified mean and covariance (see also Section 9.2.6). This is one reason the Gaussian is so widely used: the first two moments are usually all that we can reliably estimate from data, so we want a distribution that captures these properties, but otherwise makes as few additional assumptions as possible.

To simplify notation, we will assume the mean is zero. The pdf has the form

$$p(\mathbf{x}) = \frac{1}{Z}\exp(-\frac{1}{2}\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x}) \tag{4.24}$$

If we define $f_{ij}(\mathbf{x}) = x_i x_j$ and $\lambda_{ij} = \frac{1}{2}(\Sigma^{-1})_{ij}$, for $i, j \in \{1, \ldots, D\}$, we see that this is in the same form as Equation 9.74. The (differential) entropy of this distribution (using log base e) is given by

$$h(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})) = \frac{1}{2}\ln\left[(2\pi e)^D|\boldsymbol{\Sigma}|\right] \tag{4.25}$$

We now show the MVN has maximum entropy amongst all distributions with a specified covariance $\boldsymbol{\Sigma}$.

Theorem 4.1.2. *Let $q(\mathbf{x})$ be any density satisfying $\int q(\mathbf{x})x_i x_j d\mathbf{x} = \Sigma_{ij}$. Let $p = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. Then $h(q) \leq h(p)$.*

Proof. (From (Cover and Thomas 1991, p234).) We have

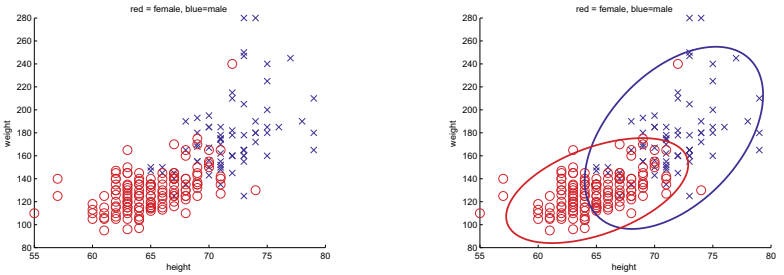
$$\begin{aligned} 0 &\leq \mathbb{KL}\left(q||p\right) = \int q(\mathbf{x})\log\frac{q(\mathbf{x})}{p(\mathbf{x})}d\mathbf{x} && (4.26)\\ &= -h(q) - \int q(\mathbf{x})\log p(\mathbf{x})d\mathbf{x} && (4.27)\\ &=^* -h(q) - \int p(\mathbf{x})\log p(\mathbf{x})d\mathbf{x} && (4.28)\\ &= -h(q) + h(p) && (4.29) \end{aligned}$$

where the key step in Equation 4.28 (marked with a *) follows since q and p yield the same moments for the quadratic form encoded by $\log p(\mathbf{x})$. □

4.2 Gaussian discriminant analysis

One important application of MVNs is to define the class conditional densities in a generative classifier, i.e.,

$$p(\mathbf{x}|y = c, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \tag{4.30}$$



(a) (b)

Figure 4.2 (a) Height/weight data. (b) Visualization of 2d Gaussians fit to each class. 95% of the probability mass is inside the ellipse. Figure generated by `gaussHeightWeight`.

The resulting technique is called (Gaussian) **discriminant analysis** or GDA (even though it is a generative, not discriminative, classifier — see Section 8.6 for more on this distinction). If $\boldsymbol{\Sigma}_c$ is diagonal, this is equivalent to naive Bayes.

We can classify a feature vector using the following decision rule, derived from Equation 2.13:

$$\hat{y}(\mathbf{x}) = \operatorname*{argmax}_c \left[\log p(y=c|\boldsymbol{\pi}) + \log p(\mathbf{x}|\boldsymbol{\theta}_c)\right] \tag{4.31}$$

When we compute the probability of $\mathbf{x}$ under each class conditional density, we are measuring the distance from $\mathbf{x}$ to the center of each class, $\boldsymbol{\mu}_c$, using Mahalanobis distance. This can be thought of as a **nearest centroids classifier**.

As an example, Figure 4.2 shows two Gaussian class-conditional densities in 2d, representing the height and weight of men and women. We can see that the features are correlated, as is to be expected (tall people tend to weigh more). The ellipses for each class contain 95% of the probability mass. If we have a uniform prior over classes, we can classify a new test vector as follows:

$$\hat{y}(\mathbf{x}) = \operatorname*{argmin}_c (\mathbf{x}-\boldsymbol{\mu}_c)^T \boldsymbol{\Sigma}_c^{-1} (\mathbf{x}-\boldsymbol{\mu}_c) \tag{4.32}$$

4.2.1 Quadratic discriminant analysis (QDA)

The posterior over class labels is given by Equation 2.13. We can gain further insight into this model by plugging in the definition of the Gaussian density, as follows:

$$p(y=c|\mathbf{x},\boldsymbol{\theta}) = \frac{\pi_c |2\pi\boldsymbol{\Sigma}_c|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_c)^T\boldsymbol{\Sigma}_c^{-1}(\mathbf{x}-\boldsymbol{\mu}_c)\right]}{\sum_{c'} \pi_{c'} |2\pi\boldsymbol{\Sigma}_{c'}|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_{c'})^T\boldsymbol{\Sigma}_{c'}^{-1}(\mathbf{x}-\boldsymbol{\mu}_{c'})\right]} \tag{4.33}$$

Thresholding this results in a quadratic function of $\mathbf{x}$. The result is known as **quadratic discriminant analysis** (QDA). Figure 4.3 gives some examples of what the decision boundaries look like in 2D.[1]

1. See `http://home.comcast.net/~tom.fawcett/public_html/ML-gallery/pages/` for an interesting visualization of decision boundaires in 2d for a variety of classificatino methods.

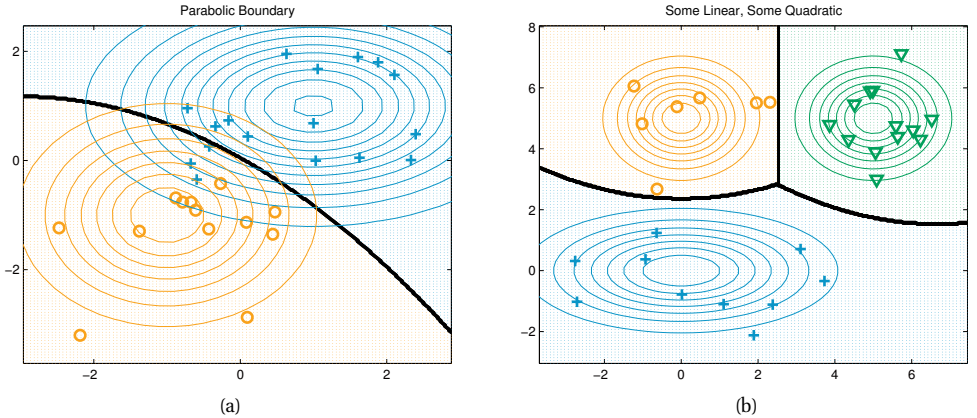


Figure 4.3 Quadratic decision boundaries in 2D for the 2 and 3 class case. Figure generated by `discrimAnalysisDboundariesDemo`.

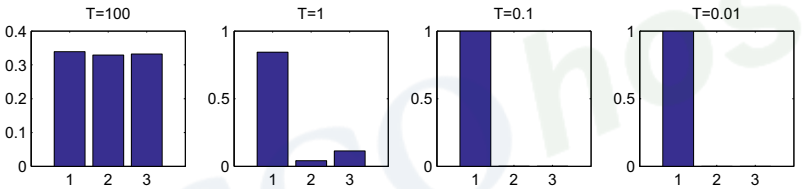


Figure 4.4 Softmax distribution $\mathcal{S}(\boldsymbol{\eta}/T)$, where $\boldsymbol{\eta} = (3, 0, 1)$, at different temperatures T. When the temperature is high (left), the distribution is uniform, whereas when the temperature is low (right), the distribution is "spiky", with all its mass on the largest element. Figure generated by `softmaxDemo2`.

4.2.2 Linear discriminant analysis (LDA)

We now consider a special case in which the covariance matrices are **tied** or **shared** across classes, $\boldsymbol{\Sigma}_c = \boldsymbol{\Sigma}$. In this case, we can simplify Equation 4.33 as follows:

$$p(y = c|\mathbf{x}, \boldsymbol{\theta}) \propto \pi_c \exp\left[\boldsymbol{\mu}_c^T \boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c\right] \tag{4.34}$$

$$= \exp\left[\boldsymbol{\mu}_c^T \boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c + \log \pi_c\right] \exp[-\frac{1}{2}\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x}] \tag{4.35}$$

Since the quadratic term $\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x}$ is independent of c, it will cancel out in the numerator and denominator. If we define

$$\gamma_c = -\frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c + \log \pi_c \tag{4.36}$$

$$\boldsymbol{\beta}_c = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c \tag{4.37}$$

then we can write

$$p(y=c|\mathbf{x},\boldsymbol{\theta}) = \frac{e^{\boldsymbol{\beta}_c^T\mathbf{x}+\gamma_c}}{\sum_{c'} e^{\boldsymbol{\beta}_{c'}^T\mathbf{x}+\gamma_{c'}}} = \mathcal{S}(\boldsymbol{\eta})_c \tag{4.38}$$

where $\boldsymbol{\eta} = [\boldsymbol{\beta}_1^T\mathbf{x}+\gamma_1, \ldots, \boldsymbol{\beta}_C^T\mathbf{x}+\gamma_C]$, and $\mathcal{S}$ is the **softmax** function, defined as follows:

$$\mathcal{S}(\boldsymbol{\eta})_c = \frac{e^{\eta_c}}{\sum_{c'=1}^{C} e^{\eta_{c'}}} \tag{4.39}$$

The softmax function is so-called since it acts a bit like the max function. To see this, let us divide each η_c by a constant T called the **temperature**. Then as $T \to 0$, we find

$$\mathcal{S}(\boldsymbol{\eta}/T)_c = \begin{cases} 1.0 & \text{if } c = \text{argmax}_{c'}\, \eta_{c'} \\ 0.0 & \text{otherwise} \end{cases} \tag{4.40}$$

In other words, at low temperatures, the distribution spends essentially all of its time in the most probable state, whereas at high temperatures, it visits all states uniformly. See Figure 4.4 for an illustration. Note that this terminology comes from the area of statistical physics, where it is common to use the **Boltzmann distribution**, which has the same form as the softmax function.

An interesting property of Equation 4.38 is that, if we take logs, we end up with a linear function of $\mathbf{x}$. (The reason it is linear is because the $\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x}$ cancels from the numerator and denominator.) Thus the decision boundary between any two classes, say c and c', will be a straight line. Hence this technique is called **linear discriminant analysis** or **LDA**. [2] We can derive the form of this line as follows:

$$p(y=c|\mathbf{x},\boldsymbol{\theta}) = p(y=c'|\mathbf{x},\boldsymbol{\theta}) \tag{4.41}$$

$$\boldsymbol{\beta}_c^T\mathbf{x}+\gamma_c = \boldsymbol{\beta}_{c'}^T\mathbf{x}+\gamma_{c'} \tag{4.42}$$

$$\mathbf{x}^T(\boldsymbol{\beta}_{c'}-\boldsymbol{\beta}) = \gamma_{c'}-\gamma_c \tag{4.43}$$

See Figure 4.5 for some examples.

An alternative to fitting an LDA model and then deriving the class posterior is to directly fit $p(y|\mathbf{x},\mathbf{W}) = \text{Cat}(y|\mathbf{W}\mathbf{x})$ for some $C \times D$ weight matrix $\mathbf{W}$. This is called **multi-class logistic regression**, or **multinomial logistic regression**.[3] We will discuss this model in detail in Section 8.2. The difference between the two approaches is explained in Section 8.6.

4.2.3 Two-class LDA

To gain further insight into the meaning of these equations, let us consider the binary case. In this case, the posterior is given by

$$p(y=1|\mathbf{x},\boldsymbol{\theta}) = \frac{e^{\boldsymbol{\beta}_1^T\mathbf{x}+\gamma_1}}{e^{\boldsymbol{\beta}_1^T\mathbf{x}+\gamma_1} + e^{\boldsymbol{\beta}_0^T\mathbf{x}+\gamma_0}} \tag{4.44}$$

$$= \frac{1}{1+e^{(\boldsymbol{\beta}_0-\boldsymbol{\beta}_1)^T\mathbf{x}+(\gamma_0-\gamma_1)}} = \text{sigm}\left((\boldsymbol{\beta}_1-\boldsymbol{\beta}_0)^T\mathbf{x}+(\gamma_1-\gamma_0)\right) \tag{4.45}$$

2. The abbreviation "LDA", could either stand for "linear discriminant analysis" or "latent Dirichlet allocation" (Section 27.3). We hope the meaning is clear from text.

3. In the language modeling community, this model is called a **maximum entropy** model, for reasons explained in Section 9.2.6.

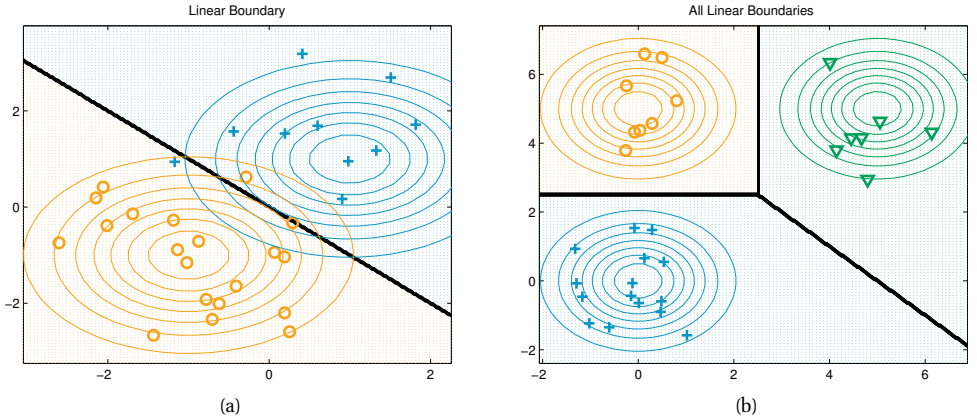


Figure 4.5 Linear decision boundaries in 2D for the 2 and 3 class case. Figure generated by `discrimAnalysisDboundariesDemo`.

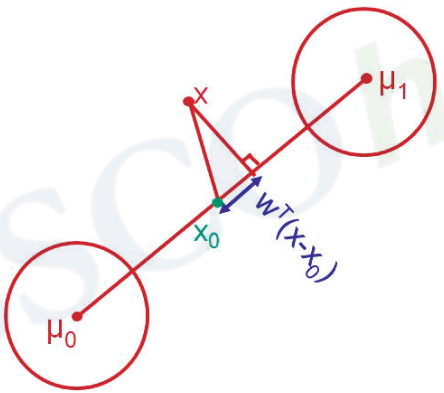


Figure 4.6 Geometry of LDA in the 2 class case where $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \mathbf{I}$.

where $\text{sigm}(\eta)$ refers to the sigmoid function (Equation 1.10).

Now

$$\gamma_1 - \gamma_0 = -\frac{1}{2}\boldsymbol{\mu}_1^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_0^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_0 + \log(\pi_1/\pi_0) \tag{4.46}$$

$$= -\frac{1}{2}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0) + \log(\pi_1/\pi_0) \tag{4.47}$$

So if we define

$$\mathbf{w} = \boldsymbol{\beta}_1 - \boldsymbol{\beta}_0 = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) \tag{4.48}$$

$$\mathbf{x}_0 = \frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0) - (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)\frac{\log(\pi_1/\pi_0)}{(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)} \tag{4.49}$$

then we have $\mathbf{w}^T\mathbf{x}_0 = -(\gamma_1 - \gamma_0)$, and hence

$$p(y=1|\mathbf{x},\boldsymbol{\theta}) \quad = \quad \text{sigm}(\mathbf{w}^T(\mathbf{x}-\mathbf{x}_0)) \tag{4.50}$$

(This is closely related to logistic regression, which we will discuss in Section 8.2.) So the final decision rule is as follows: shift $\mathbf{x}$ by $\mathbf{x}_0$, project onto the line $\mathbf{w}$, and see if the result is positive or negative.

If $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$, then $\mathbf{w}$ is in the direction of $\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0$. So we classify the point based on whether its projection is closer to $\boldsymbol{\mu}_0$ or $\boldsymbol{\mu}_1$. This is illustrated in Figure 4.6. Furthermore, if $\pi_1 = \pi_0$, then $\mathbf{x}_0 = \frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0)$, which is half way between the means. If we make $\pi_1 > \pi_0$, then $\mathbf{x}_0$ gets closer to $\boldsymbol{\mu}_0$, so more of the line belongs to class 1 *a priori*. Conversely if $\pi_1 < \pi_0$, the boundary shifts right. Thus we see that the class prior, π_c, just changes the decision threshold, and not the overall geometry, as we claimed above. (A similar argument applies in the multi-class case.)

The magnitude of $\mathbf{w}$ determines the steepness of the logistic function, and depends on how well-separated the means are, relative to the variance. In psychology and signal detection theory, it is common to define the **discriminability** of a signal from the background noise using a quantity called **d-prime**:

$$d' \triangleq \frac{\mu_1 - \mu_0}{\sigma} \tag{4.51}$$

where μ_1 is the mean of the signal and μ_0 is the mean of the noise, and σ is the standard deviation of the noise. If d' is large, the signal will be easier to discriminate from the noise.

4.2.4 MLE for discriminant analysis

We now discuss how to fit a discriminant analysis model. The simplest way is to use maximum likelihood. The log-likelihood function is as follows:

$$\log p(\mathcal{D}|\boldsymbol{\theta}) \quad = \quad \left[\sum_{i=1}^{N}\sum_{c=1}^{C}\mathbb{I}(y_i = c)\log\pi_c\right] + \sum_{c=1}^{C}\left[\sum_{i:y_i=c}\log\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)\right] \tag{4.52}$$

We see that this factorizes into a term for $\boldsymbol{\pi}$, and C terms for each $\boldsymbol{\mu}_c$ and $\boldsymbol{\Sigma}_c$. Hence we can estimate these parameters separately. For the class prior, we have $\hat{\pi}_c = \frac{N_c}{N}$, as with naive Bayes. For the class-conditional densities, we just partition the data based on its class label, and compute the MLE for each Gaussian:

$$\hat{\boldsymbol{\mu}}_c = \frac{1}{N_c}\sum_{i:y_i=c}\mathbf{x}_i, \quad \hat{\boldsymbol{\Sigma}}_c = \frac{1}{N_c}\sum_{i:y_i=c}(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_c)(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_c)^T \tag{4.53}$$

See `discrimAnalysisFit` for a Matlab implementation. Once the model has been fit, you can make predictions using `discrimAnalysisPredict`, which uses a plug-in approximation.

4.2.5 Strategies for preventing overfitting

The speed and simplicity of the MLE method is one of its greatest appeals. However, the MLE can badly overfit in high dimensions. In particular, the MLE for a full covariance matrix is singular if $N_c < D$. And even when $N_c > D$, the MLE can be ill-conditioned, meaning it is close to singular. There are several possible solutions to this problem:

- Use a diagonal covariance matrix for each class, which assumes the features are conditionally independent; this is equivalent to using a naive Bayes classifier (Section 3.5).

- Use a full covariance matrix, but force it to be the same for all classes, $\boldsymbol{\Sigma}_c = \boldsymbol{\Sigma}$. This is an example of **parameter tying** or **parameter sharing**, and is equivalent to LDA (Section 4.2.2).

- Use a diagonal covariance matrix *and* force it to be shared. This is called diagonal covariance LDA, and is discussed in Section 4.2.7.

- Use a full covariance matrix, but impose a prior and then integrate it out. If we use a conjugate prior, this can be done in closed form, using the results from Section 4.6.3; this is analogous to the "Bayesian naive Bayes" method in Section 3.5.1.2. See (Minka 2000f) for details.

- Fit a full or diagonal covariance matrix by MAP estimation. We discuss two different kinds of prior below.

- Project the data into a low-dimensional subspace and fit the Gaussians there. See Section 8.6.3.3 for a way to find the best (most discriminative) linear projection.

We discuss some of these options below.

4.2.6 Regularized LDA *

Suppose we tie the covariance matrices, so $\boldsymbol{\Sigma}_c = \boldsymbol{\Sigma}$, as in LDA, and furthermore we perform MAP estimation of $\boldsymbol{\Sigma}$ using an inverse Wishart prior of the form $\text{IW}(\text{diag}(\hat{\boldsymbol{\Sigma}}_{mle}), \nu_0)$ (see Section 4.5.1). Then we have

$$\hat{\boldsymbol{\Sigma}} = \lambda \text{diag}(\hat{\boldsymbol{\Sigma}}_{mle}) + (1-\lambda)\hat{\boldsymbol{\Sigma}}_{mle} \tag{4.54}$$

where λ controls the amount of regularization, which is related to the strength of the prior, ν_0 (see Section 4.6.2.1 for details). This technique is known as **regularized discriminant analysis** or RDA (Hastie et al. 2009, p656).

When we evaluate the class conditional densities, we need to compute $\hat{\boldsymbol{\Sigma}}^{-1}$, and hence $\hat{\boldsymbol{\Sigma}}_{mle}^{-1}$, which is impossible to compute if $D > N$. However, we can use the SVD of $\mathbf{X}$ (Section 12.2.3) to get around this, as we show below. (Note that this trick cannot be applied to QDA, which is a nonlinear function of $\mathbf{x}$.)

Let $\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^T$ be the SVD of the design matrix, where $\mathbf{V}$ is $D \times N$, $\mathbf{U}$ is an $N \times N$ orthogonal matrix, and $\mathbf{D}$ is a diagonal matrix of size N. Furthermore, define the $N \times N$ matrix $\mathbf{Z} = \mathbf{U}\mathbf{D}$; this is like a design matrix in a lower dimensional space (since we assume $N < D$). Also, define $\boldsymbol{\mu}_z = \mathbf{V}^T\boldsymbol{\mu}$ as the mean of the data in this reduced space; we can recover the original mean using $\boldsymbol{\mu} = \mathbf{V}\boldsymbol{\mu}_z$, since $\mathbf{V}^T\mathbf{V} = \mathbf{V}\mathbf{V}^T = \mathbf{I}$. With these definitions, we can

rewrite the MLE as follows:

$$\begin{aligned}
\hat{\boldsymbol{\Sigma}}_{mle} &= \frac{1}{N}\mathbf{X}^T\mathbf{X} - \boldsymbol{\mu}\boldsymbol{\mu}^T && (4.55)\\
&= \frac{1}{N}(\mathbf{Z}\mathbf{V}^T)^T(\mathbf{Z}\mathbf{V}^T) - (\mathbf{V}\boldsymbol{\mu}_z)(\mathbf{V}\boldsymbol{\mu}_z)^T && (4.56)\\
&= \frac{1}{N}\mathbf{V}\mathbf{Z}^T\mathbf{Z}\mathbf{V}^T - \mathbf{V}\boldsymbol{\mu}_z\boldsymbol{\mu}_z^T\mathbf{V}^T && (4.57)\\
&= \mathbf{V}(\frac{1}{N}\mathbf{Z}^T\mathbf{Z} - \boldsymbol{\mu}_z\boldsymbol{\mu}_z^T)\mathbf{V}^T && (4.58)\\
&= \mathbf{V}\hat{\boldsymbol{\Sigma}}_z\mathbf{V}^T && (4.59)
\end{aligned}$$

where $\hat{\boldsymbol{\Sigma}}_z$ is the empirical covariance of $\mathbf{Z}$. Hence we can rewrite the MAP estimate as

$$\begin{aligned}
\hat{\boldsymbol{\Sigma}}_{map} &= \mathbf{V}\tilde{\boldsymbol{\Sigma}}_z\mathbf{V}^T && (4.60)\\
\tilde{\boldsymbol{\Sigma}}_z &= \lambda\mathrm{diag}(\hat{\boldsymbol{\Sigma}}_z) + (1-\lambda)\hat{\boldsymbol{\Sigma}}_z && (4.61)
\end{aligned}$$

Note, however, that we never need to actually compute the $D \times D$ matrix $\hat{\boldsymbol{\Sigma}}_{map}$. This is because Equation 4.38 tells us that to classify using LDA, all we need to compute is $p(y = c|\mathbf{x}, \boldsymbol{\theta}) \propto \exp(\delta_c)$, where

$$\delta_c = \mathbf{x}^T\boldsymbol{\beta}_c + \gamma_c, \quad \boldsymbol{\beta}_c = \hat{\boldsymbol{\Sigma}}^{-1}\boldsymbol{\mu}_c, \quad \gamma_c - \frac{1}{2}\boldsymbol{\mu}_c^T\boldsymbol{\beta}_c + \log\pi_c \tag{4.62}$$

We can compute the crucial $\boldsymbol{\beta}_c$ term for RDA without inverting the $D \times D$ matrix as follows:

$$\boldsymbol{\beta}_c = \hat{\boldsymbol{\Sigma}}_{map}^{-1}\boldsymbol{\mu}_c = (\mathbf{V}\tilde{\boldsymbol{\Sigma}}_z\mathbf{V}^T)^{-1}\boldsymbol{\mu}_c = \mathbf{V}\tilde{\boldsymbol{\Sigma}}_z^{-1}\mathbf{V}^T\boldsymbol{\mu}_c = \mathbf{V}\tilde{\boldsymbol{\Sigma}}_z^{-1}\boldsymbol{\mu}_{z,c} \tag{4.63}$$

where $\boldsymbol{\mu}_{z,c} = \mathbf{V}^T\boldsymbol{\mu}_c$ is the mean of the $\mathbf{Z}$ matrix for data belonging to class c. See `rdaFit` for the code.

4.2.7 Diagonal LDA

A simple alternative to RDA is to tie the covariance matrices, so $\boldsymbol{\Sigma}_c = \boldsymbol{\Sigma}$ as in LDA, and then to use a diagonal covariance matrix for each class. This is called the **diagonal LDA** model, and is equivalent to RDA with $\lambda = 1$. The corresponding discriminant function is as follows (compare to Equation 4.33):

$$\delta_c(\mathbf{x}) = \log p(\mathbf{x}, y = c|\boldsymbol{\theta}) = -\sum_{j=1}^{D}\frac{(x_j - \mu_{cj})^2}{2\sigma_j^2} + \log\pi_c \tag{4.64}$$

Typically we set $\hat{\mu}_{cj} = \overline{x}_{cj}$ and $\hat{\sigma}_j^2 = s_j^2$, which is the **pooled empirical variance** of feature j (pooled across classes) defined by

$$s_j^2 = \frac{\sum_{c=1}^{C}\sum_{i:y_i=c}(x_{ij} - \overline{x}_{cj})^2}{N - C} \tag{4.65}$$

In high-dimensional settings, this model can work much better than LDA and RDA (Bickel and Levina 2004).

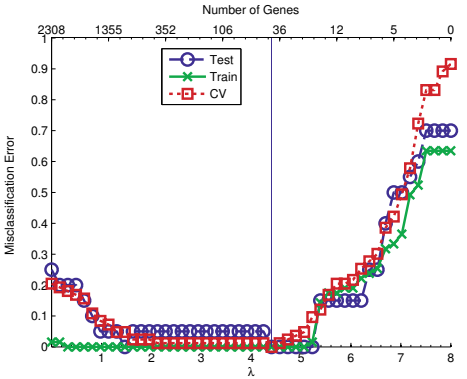


Figure 4.7 Error versus amount of shrinkage for nearest shrunken centroid classifier applied to the SRBCT gene expression data. Based on Figure 18.4 of (Hastie et al. 2009). Figure generated by `shrunkenCentroidsSRBCTdemo`.

4.2.8 Nearest shrunken centroids classifier *

One drawback of diagonal LDA is that it depends on all of the features. In high-dimensional problems, we might prefer a method that only depends on a subset of the features, for reasons of accuracy and interpretability. One approach is to use a screening method, perhaps based on mutual information, as in Section 3.5.4. We now discuss another approach to this problem known as the **nearest shrunken centroids** classifier (Hastie et al. 2009, p652).

The basic idea is to perform MAP estimation for diagonal LDA with a sparsity-promoting (Laplace) prior (see Section 13.3). More precisely, define the class-specific feature mean, μ_{cj}, in terms of the class-independent feature mean, m_j, and a class-specific offset, Δ_{cj}. Thus we have

$$\mu_{cj} = m_j + \Delta_{cj} \tag{4.66}$$

We will then put a prior on the Δ_{cj} terms to encourage them to be strictly zero and compute a MAP estimate. If, for feature j, we find that $\Delta_{cj} = 0$ for all c, then feature j will play no role in the classification decision (since μ_{cj} will be independent of c). Thus features that are not discriminative are automatically ignored. The details can be found in (Hastie et al. 2009, p652) and (Greenshtein and Park 2009). See `shrunkenCentroidsFit` for some code.

Let us give an example of the method in action, based on (Hastie et al. 2009, p652). Consider the problem of classifying a gene expression dataset, which 2308 genes, 4 classes, 63 training samples and 20 test samples. Using a diagonal LDA classifier produces 5 errors on the test set. Using the nearest shrunken centroids classifier produced 0 errors on the test set, for a range of λ values: see Figure 4.7. More importantly, the model is sparse and hence more interpretable: Figure 4.8 plots an unpenalized estimate of the difference, d_{cj}, in gray, as well as the shrunken estimates Δ_{cj} in blue. (These estimates are computed using the value of λ estimated by CV.) We see that only 39 genes are used, out of the original 2308.

Now consider an even harder problem, with 16,603 genes, a training set of 144 patients, a test set of 54 patients, and 14 different types of cancer (Ramaswamy et al. 2001). Hastie et al. (Hastie et al. 2009, p656) report that nearest shrunken centroids produced 17 errors on the test

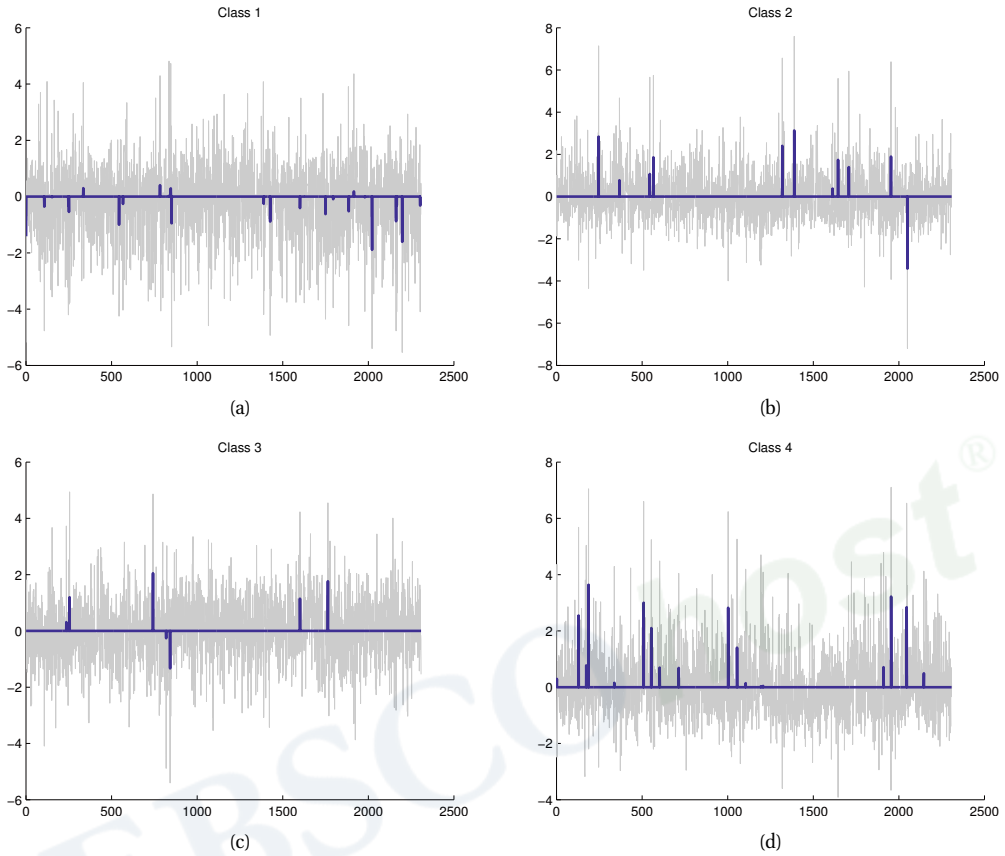


Figure 4.8 Profile of the shrunken centroids corresponding to $\lambda = 4.4$ (CV optimal in Figure 4.7). This selects 39 genes. Based on Figure 18.4 of (Hastie et al. 2009). Figure generated by `shrunkenCentroidsSRBCTdemo`.

set, using 6,520 genes, and that RDA (Section 4.2.6) produced 12 errors on the test set, using all 16,603 genes. The PMTK function `cancerHighDimClassifDemo` can be used to reproduce these numbers.

4.3 Inference in jointly Gaussian distributions

Given a joint distribution, $p(\mathbf{x}_1, \mathbf{x}_2)$, it is useful to be able to compute marginals $p(\mathbf{x}_1)$ and conditionals $p(\mathbf{x}_1|\mathbf{x}_2)$. We discuss how to do this below, and then give some applications. These operations take $O(D^3)$ time in the worst case. See Section 20.4.3 for faster methods.

4.3.1 Statement of the result

Theorem 4.3.1 (Marginals and conditionals of an MVN)**.** *Suppose* $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$ *is jointly Gaussian with parameters*

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{pmatrix}, \quad \boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1} = \begin{pmatrix} \boldsymbol{\Lambda}_{11} & \boldsymbol{\Lambda}_{12} \\ \boldsymbol{\Lambda}_{21} & \boldsymbol{\Lambda}_{22} \end{pmatrix} \tag{4.67}$$

Then the marginals are given by

$$\begin{aligned} p(\mathbf{x}_1) &= \mathcal{N}(\mathbf{x}_1 | \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11}) \\ p(\mathbf{x}_2) &= \mathcal{N}(\mathbf{x}_2 | \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{22}) \end{aligned} \tag{4.68}$$

and the posterior conditional is given by

$$\begin{aligned} p(\mathbf{x}_1|\mathbf{x}_2) &= \mathcal{N}(\mathbf{x}_1 | \boldsymbol{\mu}_{1|2}, \boldsymbol{\Sigma}_{1|2}) \\ \boldsymbol{\mu}_{1|2} &= \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\mathbf{x}_2 - \boldsymbol{\mu}_2) \\ &= \boldsymbol{\mu}_1 - \boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\Lambda}_{12}(\mathbf{x}_2 - \boldsymbol{\mu}_2) \\ &= \boldsymbol{\Sigma}_{1|2}\left(\boldsymbol{\Lambda}_{11}\boldsymbol{\mu}_1 - \boldsymbol{\Lambda}_{12}(\mathbf{x}_2 - \boldsymbol{\mu}_2)\right) \\ \boldsymbol{\Sigma}_{1|2} &= \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21} = \boldsymbol{\Lambda}_{11}^{-1} \end{aligned} \tag{4.69}$$

Equation 4.69 is of such crucial importance in this book that we have put a box around it, so you can easily find it. For the proof, see Section 4.3.4.

We see that both the marginal and conditional distributions are themselves Gaussian. For the marginals, we just extract the rows and columns corresponding to $\mathbf{x}_1$ or $\mathbf{x}_2$. For the conditional, we have to do a bit more work. However, it is not that complicated: the conditional mean is just a linear function of $\mathbf{x}_2$, and the conditional covariance is just a constant matrix that is independent of $\mathbf{x}_2$. We give three different (but equivalent) expressions for the posterior mean, and two different (but equivalent) expressions for the posterior covariance; each one is useful in different circumstances.

4.3.2 Examples

Below we give some examples of these equations in action, which will make them seem more intuitive.

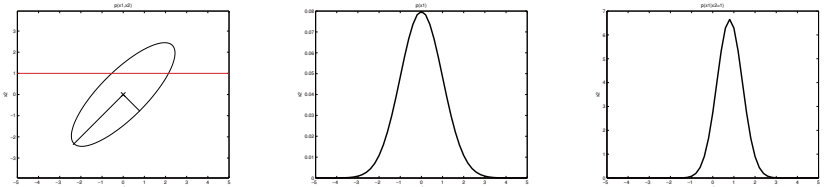
4.3.2.1 Marginals and conditionals of a 2d Gaussian

Let us consider a 2d example. The covariance matrix is

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix} \tag{4.70}$$

The marginal $p(x_1)$ is a 1D Gaussian, obtained by projecting the joint distribution onto the x_1 line:

$$p(x_1) = \mathcal{N}(x_1 | \mu_1, \sigma_1^2) \tag{4.71}$$



(a) (b) (c)

Figure 4.9 (a) A joint Gaussian distribution $p(x_1, x_2)$ with a correlation coefficient of 0.8. We plot the 95% contour and the principal axes. (b) The unconditional marginal $p(x_1)$. (c) The conditional $p(x_1|x_2) = \mathcal{N}(x_1|0.8, 0.36)$, obtained by slicing (a) at height $x_2 = 1$. Figure generated by `gaussCondition2Ddemo2`.

Suppose we observe $X_2 = x_2$; the conditional $p(x_1|x_2)$ is obtained by "slicing" the joint distribution through the $X_2 = x_2$ line (see Figure 4.9):

$$p(x_1|x_2) = \mathcal{N}\left(x_1|\mu_1 + \frac{\rho\sigma_1\sigma_2}{\sigma_2^2}(x_2 - \mu_2),\ \sigma_1^2 - \frac{(\rho\sigma_1\sigma_2)^2}{\sigma_2^2}\right) \tag{4.72}$$

If $\sigma_1 = \sigma_2 = \sigma$, we get

$$p(x_1|x_2) = \mathcal{N}\left(x_1|\mu_1 + \rho(x_2 - \mu_2),\ \sigma^2(1 - \rho^2)\right) \tag{4.73}$$

In Figure 4.9 we show an example where $\rho = 0.8$, $\sigma_1 = \sigma_2 = 1$, $\mu_1 = \mu_2 = 0$, and $x_2 = 1$. We see that $\mathbb{E}[x_1|x_2 = 1] = 0.8$, which makes sense, since $\rho = 0.8$ means that we believe that if x_2 increases by 1 (beyond its mean), then x_1 increases by 0.8. We also see $\text{var}[x_1|x_2 = 1] = 1 - 0.8^2 = 0.36$. This also makes sense: our uncertainty about x_1 has gone down, since we have learned something about x_1 (indirectly) by observing x_2. If $\rho = 0$, we get $p(x_1|x_2) = \mathcal{N}\left(x_1|\mu_1,\ \sigma_1^2\right)$, since x_2 conveys no information about x_1 if they are uncorrelated (and hence independent).

4.3.2.2 Interpolating noise-free data

Suppose we want to estimate a 1d function, defined on the interval $[0, T]$, such that $y_i = f(t_i)$ for N observed points t_i. We assume for now that the data is noise-free, so we want to **interpolate** it, that is, fit a function that goes exactly through the data. (See Section 4.4.2.3 for the noisy data case.) The question is: how does the function behave in between the observed data points? It is often reasonable to assume that the unknown function is smooth. In Chapter 15, we shall see how to encode *priors over functions*, and how to update such a prior with observed values to get a posterior over functions. But in this section, we take a simpler approach, which is adequate for MAP estimation of functions defined on 1d inputs. We follow the presentation of (Calvetti and Somersalo 2007, p135).

We start by discretizing the problem. First we divide the support of the function into D equal subintervals. We then define

$$x_j = f(s_j), \quad s_j = jh, \quad h = \frac{T}{D},\ 1 \le j \le D \tag{4.74}$$

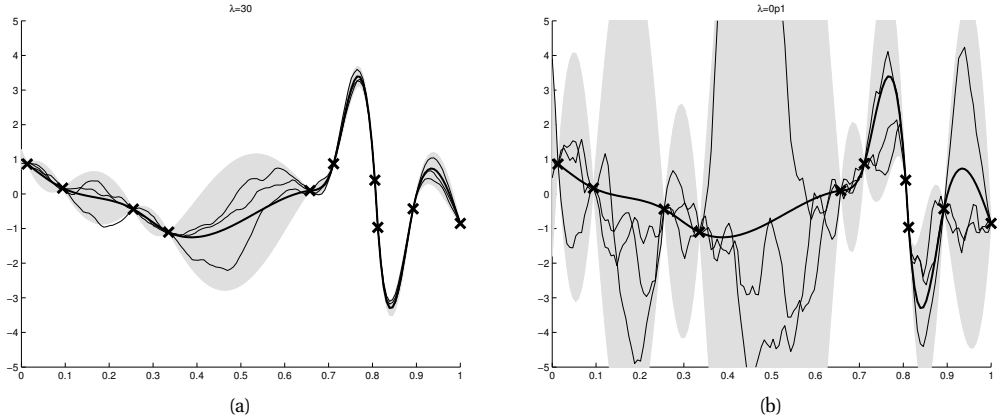


Figure 4.10 Interpolating noise-free data using a Gaussian with prior precision λ. (a) $\lambda = 30$. (b) $\lambda = 0.01$. See also Figure 4.15. Based on Figure 7.1 of (Calvetti and Somersalo 2007). Figure generated by `gaussInterpDemo`.

We can encode our smoothness prior by assuming that x_j is an average of its neighbors, x_{j-1} and x_{j+1}, plus some Gaussian noise:

$$x_j = \frac{1}{2}(x_{j-1} + x_{j+1}) + \epsilon_j, \quad 2 \leq j \leq D-1 \tag{4.75}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, (1/\lambda)\mathbf{I})$. The precision term λ controls how much we think the function will vary: a large λ corresponds to a belief that the function is very smooth, a small λ corresponds to a belief that the function is quite "wiggly". In vector form, the above equation can be written as follows:

$$\mathbf{L}\mathbf{x} = \boldsymbol{\epsilon} \tag{4.76}$$

where $\mathbf{L}$ is the $(D-2) \times D$ second order **finite difference matrix**

$$\mathbf{L} = \frac{1}{2}\begin{pmatrix} -1 & 2 & -1 & & \\ & -1 & 2 & -1 & \\ & & \ddots & & \\ & & & -1 & 2 & -1 \end{pmatrix} \tag{4.77}$$

The corresponding prior has the form

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\mathbf{0}, (\lambda \mathbf{L}^T\mathbf{L})^{-1}) \propto \exp\left(-\frac{\lambda}{2}||\mathbf{L}\mathbf{x}||_2^2\right) \tag{4.78}$$

We will henceforth assume we have scaled $\mathbf{L}$ by λ so we can ignore the λ term, and just write $\boldsymbol{\Lambda} = \mathbf{L}^T\mathbf{L}$ for the precision matrix.

Note that although $\mathbf{x}$ is D-dimensional, the precision matrix $\boldsymbol{\Lambda}$ only has rank $D-2$. Thus this is an improper prior, known as an intrinsic Gaussian random field (see Section 19.4.4 for

more information). However, providing we observe $N \geq 2$ data points, the posterior will be proper.

Now let $\mathbf{x}_2$ be the N noise-free observations of the function, and $\mathbf{x}_1$ be the $D-N$ unknown function values. Without loss of generality, assume that the unknown variables are ordered first, then the known variables. Then we can partition the $\mathbf{L}$ matrix as follows:

$$\mathbf{L} = [\mathbf{L}_1,\ \mathbf{L}_2],\ \mathbf{L}_1 \in \mathbb{R}^{(D-2)\times(D-N)},\ \mathbf{L}_2 \in \mathbb{R}^{(D-2)\times(N)} \tag{4.79}$$

We can also partition the precision matrix of the joint distribution:

$$\mathbf{\Lambda} = \mathbf{L}^T\mathbf{L} = \begin{pmatrix} \mathbf{\Lambda}_{11} & \mathbf{\Lambda}_{12} \\ \mathbf{\Lambda}_{21} & \mathbf{\Lambda}_{22} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_1^T\mathbf{L}_1 & \mathbf{L}_1^T\mathbf{L}_2 \\ \mathbf{L}_2^T\mathbf{L}_1 & \mathbf{L}_2^T\mathbf{L}_2 \end{pmatrix} \tag{4.80}$$

Using Equation 4.69, we can write the conditional distribution as follows:

$$p(\mathbf{x}_1|\mathbf{x}_2) = \mathcal{N}(\boldsymbol{\mu}_{1|2}, \mathbf{\Sigma}_{1|2}) \tag{4.81}$$

$$\boldsymbol{\mu}_{1|2} = -\mathbf{\Lambda}_{11}^{-1}\mathbf{\Lambda}_{12}\mathbf{x}_2 = -\mathbf{L}_1^{-1}\mathbf{L}_2\mathbf{x}_2 \tag{4.82}$$

$$\Sigma_{1|2} = \mathbf{\Lambda}_{11}^{-1} \tag{4.83}$$

Note that we can compute the mean by solving the following system of linear equations:

$$\mathbf{L}_1\boldsymbol{\mu}_{1|2} = -\mathbf{L}_2\mathbf{x}_2 \tag{4.84}$$

This is efficient since $\mathbf{L}_1$ is tridiagonal. Figure 4.10 gives an illustration of these equations. We see that the posterior mean $\boldsymbol{\mu}_{1|2}$ equals the observed data at the specified points, and smoothly interpolates in between, as desired.

It is also interesting to plot the 95% **pointwise marginal credibility intervals**, $\mu_j \pm 2\sqrt{\Sigma_{1|2,jj}}$, shown in grey. We see that the variance goes up as we move away from the data. We also see that the variance goes up as we decrease the precision of the prior, λ. Interestingly, λ has no effect on the posterior mean, since it cancels out when multiplying $\mathbf{\Lambda}_{11}$ and $\mathbf{\Lambda}_{12}$. By contrast, when we consider noisy data in Section 4.4.2.3, we will see that the prior precision affects the smoothness of posterior mean estimate.

The marginal credibility intervals do not capture the fact that neighboring locations are correlated. We can represent that by drawing complete functions (i.e., vectors $\mathbf{x}$) from the posterior, and plotting them. These are shown by the thin lines in Figure 4.10. These are not quite as smooth as the posterior mean itself. This is because the prior only penalizes first-order differences. See Section 4.4.2.3 for further discussion of this point.

4.3.2.3 Data imputation

Suppose we are missing some entries in a design matrix. If the columns are correlated, we can use the observed entries to predict the missing entries. Figure 4.11 shows a simple example. We sampled some data from a 20-dimensional Gaussian, and then deliberately "hid" 50% of the data in each row. We then inferred the missing entries given the observed entries, using the true (generating) model. More precisely, for each row i, we compute $p(\mathbf{x}_{\mathbf{h}_i}|\mathbf{x}_{\mathbf{v}_i}, \boldsymbol{\theta})$, where $\mathbf{h}_i$ and $\mathbf{v}_i$ are the indices of the hidden and visible entries in case i. From this, we compute the marginal distribution of each missing variable, $p(x_{h_{ij}}|\mathbf{x}_{\mathbf{v}_i}, \boldsymbol{\theta})$. We then plot the mean of this distribution, $\hat{x}_{ij} = \mathbb{E}[x_j|\mathbf{x}_{\mathbf{v}_i}, \boldsymbol{\theta}]$; this represents our "best guess" about the true value of that entry, in the

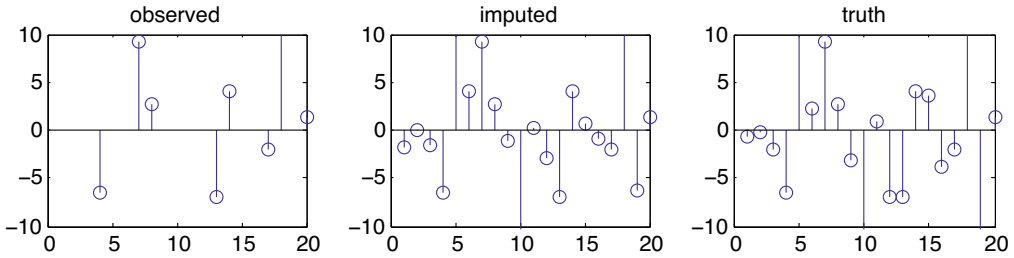


Figure 4.11 Illustration of data imputation. Left column: visualization of three rows of the data matrix with missing entries. Middle column: mean of the posterior predictive, based on partially observed data in that row, but the true model parameters. Right column: true values. Figure generated by `gaussImputationDemo`.

sense that it minimizes our expected squared error (see Section 5.7 for details). Figure 4.11 shows that the estimates are quite close to the truth. (Of course, if $j \in \mathbf{v}_i$, the expected value is equal to the observed value, $\hat{x}_{ij} = x_{ij}$.)

We can use $\text{var}\left[x_{h_{ij}}|\mathbf{x}_{\mathbf{v}_i}, \boldsymbol{\theta}\right]$ as a measure of confidence in this guess, although this is not shown. Alternatively, we could draw multiple samples from $p(\mathbf{x}_{\mathbf{h}_i}|\mathbf{x}_{\mathbf{v}_i}, \boldsymbol{\theta})$; this is called **multiple imputation**.

In addition to imputing the missing entries, we may be interested in computing the likelihood of each partially observed row in the table, $p(\mathbf{x}_{\mathbf{v}_i}|\boldsymbol{\theta})$, which can be computed using Equation 4.68. This is useful for detecting outliers (atypical observations).

4.3.3 Information form

Suppose $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. One can show that $\mathbb{E}[\mathbf{x}] = \boldsymbol{\mu}$ is the mean vector, and $\text{cov}[\mathbf{x}] = \boldsymbol{\Sigma}$ is the covariance matrix. These are called the **moment parameters** of the distribution. However, it is sometimes useful to use the **canonical parameters** or **natural parameters**, defined as

$$\boldsymbol{\Lambda} \triangleq \boldsymbol{\Sigma}^{-1}, \quad \boldsymbol{\xi} \triangleq \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \tag{4.85}$$

We can convert back to the moment parameters using

$$\boldsymbol{\mu} = \boldsymbol{\Lambda}^{-1}\boldsymbol{\xi}, \quad \boldsymbol{\Sigma} = \boldsymbol{\Lambda}^{-1} \tag{4.86}$$

(These two parameterizations are explained in more detail in Section 9.2, when we discuss the exponential family.)

Using the canonical parameters, we can write the MVN in **information form** (also called **canonical form**) as follows:

$$\mathcal{N}_c(\mathbf{x}|\boldsymbol{\xi}, \boldsymbol{\Lambda}) \quad = \quad (2\pi)^{-D/2}|\boldsymbol{\Lambda}|^{\frac{1}{2}} \ \exp\left[-\frac{1}{2}(\mathbf{x}^T\boldsymbol{\Lambda}\mathbf{x} + \boldsymbol{\xi}^T\boldsymbol{\Lambda}^{-1}\boldsymbol{\xi} - 2\mathbf{x}^T\boldsymbol{\xi})\right] \tag{4.87}$$

where we use the notation $\mathcal{N}_c()$ to distinguish it from the moment parameterization $\mathcal{N}()$.

It is also possible to derive the marginalization and conditioning formulas in information form. We find

$$p(\mathbf{x}_2) = \mathcal{N}_c(\mathbf{x}_2|\boldsymbol{\xi}_2 - \boldsymbol{\Lambda}_{21}\boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\xi}_1, \boldsymbol{\Lambda}_{22} - \boldsymbol{\Lambda}_{21}\boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\Lambda}_{12}) \quad (4.88)$$

$$p(\mathbf{x}_1|\mathbf{x}_2) = \mathcal{N}_c(\mathbf{x}_1|\boldsymbol{\xi}_1 - \boldsymbol{\Lambda}_{12}\mathbf{x}_2, \boldsymbol{\Lambda}_{11}) \quad (4.89)$$

Thus we see that marginalization is easier in moment form, and conditioning is easier in information form.

Another operation that is significantly easier in information form is multiplying two Gaussians. One can show that

$$\mathcal{N}_c(\xi_f, \lambda_f)\mathcal{N}_c(\xi_g, \lambda_g) \propto \mathcal{N}_c(\xi_f + \xi_g, \lambda_f + \lambda_g) \quad (4.90)$$

However, in moment form, things are much messier:

$$\mathcal{N}(\mu_f, \sigma_f^2)\mathcal{N}(\mu_g, \sigma_g^2) \propto \mathcal{N}\left(\frac{\mu_f\sigma_g^2 + \mu_g\sigma_f^2}{\sigma_g^2 + \sigma_g^2}, \frac{\sigma_f^2\sigma_g^2}{\sigma_g^2 + \sigma_g^2}\right) \quad (4.91)$$

4.3.4 Proof of the result *

We now prove Theorem 4.3.1. Readers who are intimidated by heavy matrix algebra can safely skip this section. We first derive some results that we will need here and elsewhere in the book. We will return to the proof at the end.

4.3.4.1 Inverse of a partitioned matrix using Schur complements

The key tool we need is a way to invert a partitioned matrix. This can be done using the following result.

Theorem 4.3.2 (Inverse of a partitioned matrix)**.** *Consider a general partitioned matrix*

$$\mathbf{M} = \begin{pmatrix} \mathbf{E} & \mathbf{F} \\ \mathbf{G} & \mathbf{H} \end{pmatrix} \quad (4.92)$$

where we assume $\mathbf{E}$ *and* $\mathbf{H}$ *are invertible. We have*

$$\mathbf{M}^{-1} = \begin{pmatrix} (\mathbf{M}/\mathbf{H})^{-1} & -(\mathbf{M}/\mathbf{H})^{-1}\mathbf{F}\mathbf{H}^{-1} \\ -\mathbf{H}^{-1}\mathbf{G}(\mathbf{M}/\mathbf{H})^{-1} & \mathbf{H}^{-1} + \mathbf{H}^{-1}\mathbf{G}(\mathbf{M}/\mathbf{H})^{-1}\mathbf{F}\mathbf{H}^{-1} \end{pmatrix} \quad (4.93)$$

$$= \begin{pmatrix} \mathbf{E}^{-1} + \mathbf{E}^{-1}\mathbf{F}(\mathbf{M}/\mathbf{E})^{-1}\mathbf{G}\mathbf{E}^{-1} & -\mathbf{E}^{-1}\mathbf{F}(\mathbf{M}/\mathbf{E})^{-1} \\ -(\mathbf{M}/\mathbf{E})^{-1}\mathbf{G}\mathbf{E}^{-1} & (\mathbf{M}/\mathbf{E})^{-1} \end{pmatrix} \quad (4.94)$$

where

$$\mathbf{M}/\mathbf{H} \triangleq \mathbf{E} - \mathbf{F}\mathbf{H}^{-1}\mathbf{G} \quad (4.95)$$

$$\mathbf{M}/\mathbf{E} \triangleq \mathbf{H} - \mathbf{G}\mathbf{E}^{-1}\mathbf{F} \quad (4.96)$$

We say that $\mathbf{M}/\mathbf{H}$ *is the* **Schur complement** *of* $\mathbf{M}$ *wrt* $\mathbf{H}$*. Equation 4.93 is called the* **partitioned inverse formula**.

Proof. If we could block diagonalize $\mathbf{M}$, it would be easier to invert. To zero out the top right block of $\mathbf{M}$ we can pre-multiply as follows

$$\begin{pmatrix} \mathbf{I} & -\mathbf{F}\mathbf{H}^{-1} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{E} & \mathbf{F} \\ \mathbf{G} & \mathbf{H} \end{pmatrix} = \begin{pmatrix} \mathbf{E} - \mathbf{F}\mathbf{H}^{-1}\mathbf{G} & \mathbf{0} \\ \mathbf{G} & \mathbf{H} \end{pmatrix} \tag{4.97}$$

Similarly, to zero out the bottom left we can post-multiply as follows

$$\begin{pmatrix} \mathbf{E} - \mathbf{F}\mathbf{H}^{-1}\mathbf{G} & \mathbf{0} \\ \mathbf{G} & \mathbf{H} \end{pmatrix} \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{H}^{-1}\mathbf{G} & \mathbf{I} \end{pmatrix} = \begin{pmatrix} \mathbf{E} - \mathbf{F}\mathbf{H}^{-1}\mathbf{G} & \mathbf{0} \\ \mathbf{0} & \mathbf{H} \end{pmatrix} \tag{4.98}$$

Putting it all together we get

$$\underbrace{\begin{pmatrix} \mathbf{I} & -\mathbf{F}\mathbf{H}^{-1} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}}_{\mathbf{X}} \underbrace{\begin{pmatrix} \mathbf{E} & \mathbf{F} \\ \mathbf{G} & \mathbf{H} \end{pmatrix}}_{\mathbf{M}} \underbrace{\begin{pmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{H}^{-1}\mathbf{G} & \mathbf{I} \end{pmatrix}}_{\mathbf{Z}} = \underbrace{\begin{pmatrix} \mathbf{E} - \mathbf{F}\mathbf{H}^{-1}\mathbf{G} & \mathbf{0} \\ \mathbf{0} & \mathbf{H} \end{pmatrix}}_{\mathbf{W}} \tag{4.99}$$

Taking the inverse of both sides yields

$$\mathbf{Z}^{-1}\mathbf{M}^{-1}\mathbf{X}^{-1} = \mathbf{W}^{-1} \tag{4.100}$$

and hence

$$\mathbf{M}^{-1} = \mathbf{Z}\mathbf{W}^{-1}\mathbf{X} \tag{4.101}$$

Substituting in the definitions we get

$$\begin{pmatrix} \mathbf{E} & \mathbf{F} \\ \mathbf{G} & \mathbf{H} \end{pmatrix}^{-1} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{H}^{-1}\mathbf{G} & \mathbf{I} \end{pmatrix} \begin{pmatrix} (\mathbf{M}/\mathbf{H})^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{H}^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{I} & -\mathbf{F}\mathbf{H}^{-1} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \tag{4.102}$$

$$= \begin{pmatrix} (\mathbf{M}/\mathbf{H})^{-1} & \mathbf{0} \\ -\mathbf{H}^{-1}\mathbf{G}(\mathbf{M}/\mathbf{H})^{-1} & \mathbf{H}^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{I} & -\mathbf{F}\mathbf{H}^{-1} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \tag{4.103}$$

$$= \begin{pmatrix} (\mathbf{M}/\mathbf{H})^{-1} & -(\mathbf{M}/\mathbf{H})^{-1}\mathbf{F}\mathbf{H}^{-1} \\ -\mathbf{H}^{-1}\mathbf{G}(\mathbf{M}/\mathbf{H})^{-1} & \mathbf{H}^{-1} + \mathbf{H}^{-1}\mathbf{G}(\mathbf{M}/\mathbf{H})^{-1}\mathbf{F}\mathbf{H}^{-1} \end{pmatrix} \tag{4.104}$$

Alternatively, we could have decomposed the matrix $\mathbf{M}$ in terms of $\mathbf{E}$ and $\mathbf{M}/\mathbf{E} = (\mathbf{H} - \mathbf{G}\mathbf{E}^{-1}\mathbf{F})$, yielding

$$\begin{pmatrix} \mathbf{E} & \mathbf{F} \\ \mathbf{G} & \mathbf{H} \end{pmatrix}^{-1} = \begin{pmatrix} \mathbf{E}^{-1} + \mathbf{E}^{-1}\mathbf{F}(\mathbf{M}/\mathbf{E})^{-1}\mathbf{G}\mathbf{E}^{-1} & -\mathbf{E}^{-1}\mathbf{F}(\mathbf{M}/\mathbf{E})^{-1} \\ -(\mathbf{M}/\mathbf{E})^{-1}\mathbf{G}\mathbf{E}^{-1} & (\mathbf{M}/\mathbf{E})^{-1} \end{pmatrix} \tag{4.105}$$

□

4.3.4.2 The matrix inversion lemma

We now derive some useful corollaries of the above result.

Corollary 4.3.1 (Matrix inversion lemma)**.** *Consider a general partitioned matrix* $\mathbf{M} = \begin{pmatrix} \mathbf{E} & \mathbf{F} \\ \mathbf{G} & \mathbf{H} \end{pmatrix}$, *where we assume* $\mathbf{E}$ *and* $\mathbf{H}$ *are invertible. We have*

$$(\mathbf{E} - \mathbf{F}\mathbf{H}^{-1}\mathbf{G})^{-1} = \mathbf{E}^{-1} + \mathbf{E}^{-1}\mathbf{F}(\mathbf{H} - \mathbf{G}\mathbf{E}^{-1}\mathbf{F})^{-1}\mathbf{G}\mathbf{E}^{-1} \tag{4.106}$$

$$(\mathbf{E} - \mathbf{F}\mathbf{H}^{-1}\mathbf{G})^{-1}\mathbf{F}\mathbf{H}^{-1} = \mathbf{E}^{-1}\mathbf{F}(\mathbf{H} - \mathbf{G}\mathbf{E}^{-1}\mathbf{F})^{-1} \tag{4.107}$$

$$|\mathbf{E} - \mathbf{F}\mathbf{H}^{-1}\mathbf{G}| = |\mathbf{H} - \mathbf{G}\mathbf{E}^{-1}\mathbf{F}||\mathbf{H}^{-1}||\mathbf{E}| \tag{4.108}$$

The first two equations are known as the **matrix inversion lemma** or the **Sherman-Morrison-Woodbury formula**. The third equation is known as the **matrix determinant lemma**. A typical application in machine learning / statistics is the following. Let $\mathbf{E} = \boldsymbol{\Sigma}$ be a $N \times N$ diagonal matrix, let $\mathbf{F} = \mathbf{G}^T = \mathbf{X}$ of size $N \times D$, where $N \gg D$, and let $\mathbf{H}^{-1} = -\mathbf{I}$. Then we have

$$(\boldsymbol{\Sigma} + \mathbf{X}\mathbf{X}^T)^{-1} = \boldsymbol{\Sigma}^{-1} - \boldsymbol{\Sigma}^{-1}\mathbf{X}(\mathbf{I} + \mathbf{X}^T\boldsymbol{\Sigma}^{-1}\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{\Sigma}^{-1} \tag{4.109}$$

The LHS takes $O(N^3)$ time to compute, the RHS takes time $O(D^3)$ to compute.

Another application concerns computing a **rank one update** of an inverse matrix. Let $H = -1$ (a scalar), $\mathbf{F} = \mathbf{u}$ (a column vector), and $\mathbf{G} = \mathbf{v}^T$ (a row vector). Then we have

$$(\mathbf{E} + \mathbf{u}\mathbf{v}^T)^{-1} = \mathbf{E}^{-1} + \mathbf{E}^{-1}\mathbf{u}(-1 - \mathbf{v}^T\mathbf{E}^{-1}\mathbf{u})^{-1}\mathbf{v}^T\mathbf{E}^{-1} \tag{4.110}$$

$$= \mathbf{E}^{-1} - \frac{\mathbf{E}^{-1}\mathbf{u}\mathbf{v}^T\mathbf{E}^{-1}}{1 + \mathbf{v}^T\mathbf{E}^{-1}\mathbf{u}} \tag{4.111}$$

This is useful when we incrementally add a data vector to a design matrix, and want to update our sufficient statistics. (One can derive an analogous formula for removing a data vector.)

Proof. To prove Equation 4.106, we simply equate the top left block of Equation 4.93 and Equation 4.94. To prove Equation 4.107, we simple equate the top right blocks of Equations 4.93 and 4.94. The proof of Equation 4.108 is left as an exercise. □

4.3.4.3 Proof of Gaussian conditioning formulas

We can now return to our original goal, which is to derive Equation 4.69. Let us factor the joint $p(\mathbf{x}_1, \mathbf{x}_2)$ as $p(\mathbf{x}_2)p(\mathbf{x}_1|\mathbf{x}_2)$ as follows:

$$E = \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1 - \boldsymbol{\mu}_1 \\ \mathbf{x}_2 - \boldsymbol{\mu}_2\end{pmatrix}^T \begin{pmatrix}\boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22}\end{pmatrix}^{-1} \begin{pmatrix}\mathbf{x}_1 - \boldsymbol{\mu}_1 \\ \mathbf{x}_2 - \boldsymbol{\mu}_2\end{pmatrix}\right\} \tag{4.112}$$

Using Equation 4.102 the above exponent becomes

$$E = \exp\left\{-\frac{1}{2}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}^T\begin{pmatrix}\mathbf{I} & \mathbf{0}\\ -\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21} & \mathbf{I}\end{pmatrix}\begin{pmatrix}(\boldsymbol{\Sigma}/\boldsymbol{\Sigma}_{22})^{-1} & \mathbf{0}\\ \mathbf{0} & \boldsymbol{\Sigma}_{22}^{-1}\end{pmatrix}\right. \tag{4.113}$$

$$\left.\times\begin{pmatrix}\mathbf{I} & -\boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\\ \mathbf{0} & \mathbf{I}\end{pmatrix}\begin{pmatrix}\mathbf{x}_1-\boldsymbol{\mu}_1\\ \mathbf{x}_2-\boldsymbol{\mu}_2\end{pmatrix}\right\} \tag{4.114}$$

$$= \exp\left\{-\frac{1}{2}(\mathbf{x}_1-\boldsymbol{\mu}_1-\boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2))^T(\boldsymbol{\Sigma}/\boldsymbol{\Sigma}_{22})^{-1}\right. \tag{4.115}$$

$$\left.(\mathbf{x}_1-\boldsymbol{\mu}_1-\boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2))\right\}\times\exp\left\{-\frac{1}{2}(\mathbf{x}_2-\boldsymbol{\mu}_2)^T\boldsymbol{\Sigma}_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2)\right\} \tag{4.116}$$

This is of the form

$$\exp(\text{quadratic form in } \mathbf{x}_1,\mathbf{x}_2)\times\exp(\text{quadratic form in } \mathbf{x}_2) \tag{4.117}$$

Hence we have successfully factorized the joint as

$$p(\mathbf{x}_1,\mathbf{x}_2) = p(\mathbf{x}_1|\mathbf{x}_2)p(\mathbf{x}_2) \tag{4.118}$$

$$= \mathcal{N}(\mathbf{x}_1|\boldsymbol{\mu}_{1|2},\boldsymbol{\Sigma}_{1|2})\mathcal{N}(\mathbf{x}_2|\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_{22}) \tag{4.119}$$

where the parameters of the conditional distribution can be read off from the above equations using

$$\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1+\boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\mathbf{x}_2-\boldsymbol{\mu}_2) \tag{4.120}$$

$$\boldsymbol{\Sigma}_{1|2} = \boldsymbol{\Sigma}/\boldsymbol{\Sigma}_{22}=\boldsymbol{\Sigma}_{11}-\boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21} \tag{4.121}$$

We can also use the fact that $|\mathbf{M}| = |\mathbf{M}/\mathbf{H}||\mathbf{H}|$ to check the normalization constants are correct:

$$(2\pi)^{(d_1+d_2)/2}|\boldsymbol{\Sigma}|^{\frac{1}{2}} = (2\pi)^{(d_1+d_2)/2}(|\boldsymbol{\Sigma}/\boldsymbol{\Sigma}_{22}|\ |\boldsymbol{\Sigma}_{22}|)^{\frac{1}{2}} \tag{4.122}$$

$$= (2\pi)^{d_1/2}|\boldsymbol{\Sigma}/\boldsymbol{\Sigma}_{22}|^{\frac{1}{2}}\ (2\pi)^{d_2/2}|\boldsymbol{\Sigma}_{22}|^{\frac{1}{2}} \tag{4.123}$$

where $d_1=\dim(\mathbf{x}_1)$ and $d_2=\dim(\mathbf{x}_2)$.

We leave the proof of the other forms of the result in Equation 4.69 as an exercise.

4.4 Linear Gaussian systems

Suppose we have two variables, $\mathbf{x}$ and $\mathbf{y}$. Let $\mathbf{x}\in\mathbb{R}^{D_x}$ be a hidden variable, and $\mathbf{y}\in\mathbb{R}^{D_y}$ be a noisy observation of $\mathbf{x}$. Let us assume we have the following prior and likelihood:

$$\boxed{\begin{aligned}p(\mathbf{x}) &= \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_x,\boldsymbol{\Sigma}_x)\\ p(\mathbf{y}|\mathbf{x}) &= \mathcal{N}(\mathbf{y}|\mathbf{A}\mathbf{x}+\mathbf{b},\boldsymbol{\Sigma}_y)\end{aligned}} \tag{4.124}$$

where $\mathbf{A}$ is a matrix of size $D_y \times D_x$. This is an example of a **linear Gaussian system**. We can represent this schematically as $\mathbf{x} \rightarrow \mathbf{y}$, meaning $\mathbf{x}$ generates $\mathbf{y}$. In this section, we show how to "invert the arrow", that is, how to infer $\mathbf{x}$ from $\mathbf{y}$. We state the result below, then give several examples, and finally we derive the result. We will see many more applications of these results in later chapters.

4.4.1 Statement of the result

Theorem 4.4.1 (Bayes rule for linear Gaussian systems)**.** *Given a linear Gaussian system, as in Equation 4.124, the posterior $p(\mathbf{x}|\mathbf{y})$ is given by the following:*

$$\begin{aligned} p(\mathbf{x}|\mathbf{y}) &= \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{x|y}, \boldsymbol{\Sigma}_{x|y}) \\ \boldsymbol{\Sigma}_{x|y}^{-1} &= \boldsymbol{\Sigma}_x^{-1} + \mathbf{A}^T \boldsymbol{\Sigma}_y^{-1} \mathbf{A} \\ \boldsymbol{\mu}_{x|y} &= \boldsymbol{\Sigma}_{x|y}[\mathbf{A}^T \boldsymbol{\Sigma}_y^{-1} (\mathbf{y} - \mathbf{b}) + \boldsymbol{\Sigma}_x^{-1} \boldsymbol{\mu}_x] \end{aligned} \tag{4.125}$$

In addition, the normalization constant $p(\mathbf{y})$ is given by

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\boldsymbol{\mu}_x + \mathbf{b}, \boldsymbol{\Sigma}_y + \mathbf{A}\boldsymbol{\Sigma}_x\mathbf{A}^T) \tag{4.126}$$

For the proof, see Section 4.4.3.

4.4.2 Examples

In this section, we give some example applications of the above result.

4.4.2.1 Inferring an unknown scalar from noisy measurements

Suppose we make N noisy measurements y_i of some underlying quantity x; let us assume the measurement noise has fixed precision $\lambda_y = 1/\sigma^2$, so the likelihood is

$$p(y_i|x) = \mathcal{N}(y_i|x, \lambda_y^{-1}) \tag{4.127}$$

Now let us use a Gaussian prior for the value of the unknown source:

$$p(x) = \mathcal{N}(x|\mu_0, \lambda_0^{-1}) \tag{4.128}$$

We want to compute $p(x|y_1, \ldots, y_N, \sigma^2)$. We can convert this to a form that lets us apply Bayes rule for Gaussians by defining $\mathbf{y} = (y_1, \ldots, y_N)$, $\mathbf{A} = \mathbf{1}_N^T$ (an $1 \times N$ row vector of 1's), and $\boldsymbol{\Sigma}_y^{-1} = \mathrm{diag}(\lambda_y \mathbf{I})$. Then we get

$$p(x|\mathbf{y}) = \mathcal{N}(x|\mu_N, \lambda_N^{-1}) \tag{4.129}$$

$$\lambda_N = \lambda_0 + N\lambda_y \tag{4.130}$$

$$\mu_N = \frac{N\lambda_y \overline{y} + \lambda_0 \mu_0}{\lambda_N} = \frac{N\lambda_y}{N\lambda_y + \lambda_0}\overline{y} + \frac{\lambda_0}{N\lambda_y + \lambda_0}\mu_0 \tag{4.131}$$

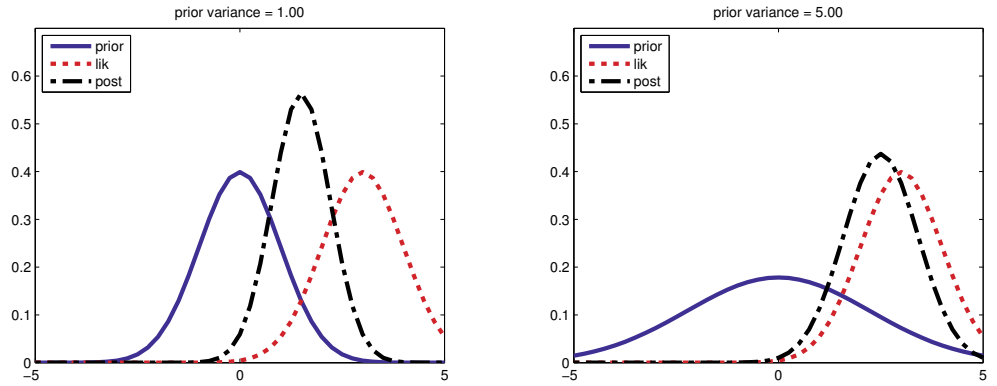


Figure 4.12 Inference about x given a noisy observation $y = 3$. (a) Strong prior $\mathcal{N}(0, 1)$. The posterior mean is "shrunk" towards the prior mean, which is 0. (a) Weak prior $\mathcal{N}(0, 5)$. The posterior mean is similar to the MLE. Figure generated by `gaussInferParamsMean1d`.

These equations are quite intuitive: the posterior precision λ_N is the prior precision λ_0 plus N units of measurement precision λ_y. Also, the posterior mean μ_N is a convex combination of the MLE $\overline{y}$ and the prior mean μ_0. This makes it clear that the posterior mean is a compromise between the MLE and the prior. If the prior is weak relative to the signal strength (λ_0 is small relative to λ_y), we put more weight on the MLE. If the prior is strong relative to the signal strength (λ_0 is large relative to λ_y), we put more weight on the prior. This is illustrated in Figure 4.12, which is very similar to the analogous results for the beta-binomial model in Figure 3.6.

Note that the posterior mean is written in terms of $N\lambda_y\overline{y}$, so having N measurements each of precision λ_y is like having one measurement with value $\overline{y}$ and precision $N\lambda_y$.

We can rewrite the results in terms of the posterior variance, rather than posterior precision, as follows:

$$p(x|\mathcal{D}, \sigma^2) = \mathcal{N}(x|\mu_N, \tau_N^2) \tag{4.132}$$

$$\tau_N^2 = \frac{1}{\frac{N}{\sigma^2} + \frac{1}{\tau_0^2}} = \frac{\sigma^2\tau_0^2}{N\tau_0^2 + \sigma^2} \tag{4.133}$$

$$\mu_N = \tau_N^2\left(\frac{\mu_0}{\tau_0^2} + \frac{N\overline{y}}{\sigma^2}\right) = \frac{\sigma^2}{N\tau_0^2 + \sigma^2}\mu_0 + \frac{N\tau_0^2}{N\tau_0^2 + \sigma^2}\overline{y} \tag{4.134}$$

where $\tau_0^2 = 1/\lambda_0$ is the prior variance and $\tau_N^2 = 1/\lambda_N$ is the posterior variance.

We can also compute the posterior sequentially, by updating after each observation. If $N = 1$, we can rewrite the posterior after seeing a single observation as follows (where we define $\Sigma_y = \sigma^2$, $\Sigma_0 = \tau_0^2$ and $\Sigma_1 = \tau_1^2$ to be the variances of the likelihood, prior and

posterior):

$$p(x|y) = \mathcal{N}(x|\mu_1, \Sigma_1) \tag{4.135}$$

$$\Sigma_1 = \left(\frac{1}{\Sigma_0} + \frac{1}{\Sigma_y}\right)^{-1} = \frac{\Sigma_y \Sigma_0}{\Sigma_0 + \Sigma_y} \tag{4.136}$$

$$\mu_1 = \Sigma_1 \left(\frac{\mu_0}{\Sigma_0} + \frac{y}{\Sigma_y}\right) \tag{4.137}$$

We can rewrite the posterior mean in 3 different ways:

$$\mu_1 = \frac{\Sigma_y}{\Sigma_y + \Sigma_0}\mu_0 + \frac{\Sigma_0}{\Sigma_y + \Sigma_0} y \tag{4.138}$$

$$= \mu_0 + (y - \mu_0)\frac{\Sigma_0}{\Sigma_y + \Sigma_0} \tag{4.139}$$

$$= y - (y - \mu_0)\frac{\Sigma_y}{\Sigma_y + \Sigma_0} \tag{4.140}$$

The first equation is a convex combination of the prior and the data. The second equation is the prior mean adjusted towards the data. The third equation is the data adjusted towards the prior mean; this is called **shrinkage**. These are all equivalent ways of expressing the tradeoff between likelihood and prior. If Σ_0 is small relative to Σ_Y, corresponding to a strong prior, the amount of shrinkage is large (see Figure 4.12(a)), whereas if Σ_0 is large relative to Σ_y, corresponding to a weak prior, the amount of shrinkage is small (see Figure 4.12(b)).

Another way to quantify the amount of shrinkage is in terms of the **signal-to-noise ratio**, which is defined as follows:

$$\text{SNR} \triangleq \frac{\mathbb{E}\left[X^2\right]}{\mathbb{E}\left[\epsilon^2\right]} = \frac{\Sigma_0 + \mu_0^2}{\Sigma_y} \tag{4.141}$$

where $x \sim \mathcal{N}(\mu_0, \Sigma_0)$ is the true signal, $y = x + \epsilon$ is the observed signal, and $\epsilon \sim \mathcal{N}(0, \Sigma_y)$ is the noise term.

4.4.2.2 Inferring an unknown vector from noisy measurements

Now consider N vector-valued observations, $\mathbf{y}_i \sim \mathcal{N}(\mathbf{x}, \mathbf{\Sigma}_y)$, and a Gaussian prior, $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_0, \mathbf{\Sigma}_0)$. Setting $\mathbf{A} = \mathbf{I}$, $\mathbf{b} = \mathbf{0}$, and using $\overline{\mathbf{y}}$ for the effective observation with precision $N\mathbf{\Sigma}_y^{-1}$, we have

$$p(\mathbf{x}|\mathbf{y}_1, \ldots, \mathbf{y}_N) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_N, \mathbf{\Sigma}_N) \tag{4.142}$$

$$\mathbf{\Sigma}_N^{-1} = \mathbf{\Sigma}_0^{-1} + N\mathbf{\Sigma}_y^{-1} \tag{4.143}$$

$$\boldsymbol{\mu}_N = \mathbf{\Sigma}_N(\mathbf{\Sigma}_y^{-1}(N\overline{\mathbf{y}}) + \mathbf{\Sigma}_0^{-1}\boldsymbol{\mu}_0) \tag{4.144}$$

See Figure 4.13 for a 2d example. We can think of $\mathbf{x}$ as representing the true, but unknown, location of an object in 2d space, such as a missile or airplane, and the $\mathbf{y}_i$ as being noisy observations, such as radar "blips". As we receive more blips, we are better able to localize the source. In Section 18.3.1, we will see how to extend this example to track moving objects using the famous Kalman filter algorithm.