Gaussian processes

Chuong B. Do

December 1, 2007

Many of the classical machine learning algorithms that we talked about during the first half of this course fit the following pattern: given a training set of i.i.d. examples sampled from some unknown distribution,

1. solve a convex optimization problem in order to identify the single "best fit" model for the data, and

2. use this estimated model to make "best guess" predictions for future test input points.

In these notes, we will talk about a different flavor of learning algorithms, known as **Bayesian methods**. Unlike classical learning algorithm, Bayesian algorithms do not attempt to identify "best-fit" models of the data (or similarly, make "best guess" predictions for new test inputs). Instead, they compute a posterior distribution over models (or similarly, compute posterior predictive distributions for new test inputs). These distributions provide a useful way to quantify our uncertainty in model estimates, and to exploit our knowledge of this uncertainty in order to make more robust predictions on new test points.

We focus on **regression** problems, where the goal is to learn a mapping from some input space $\mathcal{X} = \mathbf{R}^n$ of n-dimensional vectors to an output space $\mathcal{Y} = \mathbf{R}$ of real-valued targets. In particular, we will talk about a kernel-based fully Bayesian regression algorithm, known as Gaussian process regression. The material covered in these notes draws heavily on many different topics that we discussed previously in class (namely, the probabilistic interpretation of linear regression[1], Bayesian methods[2], kernels[3], and properties of multivariate Gaussians[4]).

The organization of these notes is as follows. In Section 1, we provide a brief review of multivariate Gaussian distributions and their properties. In Section 2, we briefly review Bayesian methods in the context of probabilistic linear regression. The central ideas underlying Gaussian processes are presented in Section 3, and we derive the full Gaussian process regression model in Section 4.

[1]See course lecture notes on "Supervised Learning, Discriminative Algorithms."
[2]See course lecture notes on "Regularization and Model Selection."
[3]See course lecture notes on "Support Vector Machines."
[4]See course lecture notes on "Factor Analysis."

1 Multivariate Gaussians

A vector-valued random variable $x \in \mathbf{R}^n$ is said to have a **multivariate normal (or Gaussian) distribution** with mean $\mu \in \mathbf{R}^n$ and covariance matrix $\Sigma \in \mathbf{S}^n_{++}$ if

$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right). \tag{1}$$

We write this as $x \sim \mathcal{N}(\mu, \Sigma)$. Here, recall from the section notes on linear algebra that $\mathbf{S}^n_{++}$ refers to the space of symmetric positive definite $n \times n$ matrices.[5]

Generally speaking, Gaussian random variables are extremely useful in machine learning and statistics for two main reasons. First, they are extremely common when modeling "noise" in statistical algorithms. Quite often, noise can be considered to be the accumulation of a large number of small independent random perturbations affecting the measurement process; by the Central Limit Theorem, summations of independent random variables will tend to "look Gaussian." Second, Gaussian random variables are convenient for many analytical manipulations, because many of the integrals involving Gaussian distributions that arise in practice have simple closed form solutions. In the remainder of this section, we will review a number of useful properties of multivariate Gaussians.

Consider a random vector $x \in \mathbf{R}^n$ with $x \sim \mathcal{N}(\mu, \Sigma)$. Suppose also that the variables in x have been partitioned into two sets $x_A = [x_1 \ \cdots \ x_r]^T \in \mathbf{R}^r$ and $x_B = [x_{r+1} \ \cdots \ x_n]^T \in \mathbf{R}^{n-r}$ (and similarly for μ and Σ), such that

$$x = \begin{bmatrix} x_A \\ x_B \end{bmatrix} \qquad \mu = \begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix} \qquad \Sigma = \begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix}.$$

Here, $\Sigma_{AB} = \Sigma_{BA}^T$ since $\Sigma = E[(x-\mu)(x-\mu)^T] = \Sigma^T$. The following properties hold:

1. **Normalization.** The density function normalizes, i.e.,

   $$\int_x p(x; \mu, \Sigma) dx = 1.$$

   This property, though seemingly trivial at first glance, turns out to be immensely useful for evaluating all sorts of integrals, even ones which appear to have no relation to probability distributions at all (see Appendix A.1)!

2. **Marginalization.** The marginal densities,

   $$p(x_A) = \int_{x_B} p(x_A, x_B; \mu, \Sigma) dx_B$$
   $$p(x_B) = \int_{x_A} p(x_A, x_B; \mu, \Sigma) dx_A$$

[5] There are actually cases in which we would want to deal with multivariate Gaussian distributions where Σ is positive semidefinite but not positive definite (i.e., Σ is not full rank). In such cases, Σ^{-1} does not exist, so the definition of the Gaussian density given in (1) does not apply. For instance, see the course lecture notes on "Factor Analysis."

are Gaussian:

$$
\begin{aligned}
x_A &\sim \mathcal{N}(\mu_A, \Sigma_{AA}) \\
x_B &\sim \mathcal{N}(\mu_B, \Sigma_{BB}).
\end{aligned}
$$

3. **Conditioning.** The conditional densities

$$
\begin{aligned}
p(x_A \mid x_B) &= \frac{p(x_A, x_B; \mu, \Sigma)}{\int_{x_A} p(x_A, x_B; \mu, \Sigma) dx_A} \\
p(x_B \mid x_A) &= \frac{p(x_A, x_B; \mu, \Sigma)}{\int_{x_B} p(x_A, x_B; \mu, \Sigma) dx_B}
\end{aligned}
$$

are also Gaussian:

$$
\begin{aligned}
x_A \mid x_B &\sim \mathcal{N}\big(\mu_A + \Sigma_{AB}\Sigma_{BB}^{-1}(x_B - \mu_B), \Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA}\big) \\
x_B \mid x_A &\sim \mathcal{N}\big(\mu_B + \Sigma_{BA}\Sigma_{AA}^{-1}(x_A - \mu_A), \Sigma_{BB} - \Sigma_{BA}\Sigma_{AA}^{-1}\Sigma_{AB}\big).
\end{aligned}
$$

A proof of this property is given in Appendix A.2.

4. **Summation.** The sum of independent Gaussian random variables (with the same dimensionality), $y \sim \mathcal{N}(\mu, \Sigma)$ and $z \sim \mathcal{N}(\mu', \Sigma')$, is also Gaussian:

$$
y + z \sim \mathcal{N}(\mu + \mu', \Sigma + \Sigma').
$$

2 Bayesian linear regression

Let $S = \{(x^{(i)}, y^{(i)})\}_{i=1}^m$ be a training set of i.i.d. examples from some unknown distribution. The standard probabilistic interpretation of linear regression states that

$$
y^{(i)} = \theta^T x^{(i)} + \varepsilon^{(i)}, \qquad i = 1, \ldots, m
$$

where the $\varepsilon^{(i)}$ are i.i.d. "noise" variables with independent $\mathcal{N}(0, \sigma^2)$ distributions. It follows that $y^{(i)} - \theta^T x^{(i)} \sim \mathcal{N}(0, \sigma^2)$, or equivalently,

$$
P(y^{(i)} \mid x^{(i)}, \theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right).
$$

For notational convenience, we define

$$
X = \begin{bmatrix} — & (x^{(1)})^T & — \\ — & (x^{(2)})^T & — \\ & \vdots & \\ — & (x^{(m)})^T & — \end{bmatrix} \in \mathbf{R}^{m \times n} \qquad \vec{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix} \in \mathbf{R}^m \qquad \vec{\varepsilon} = \begin{bmatrix} \varepsilon^{(1)} \\ \varepsilon^{(2)} \\ \vdots \\ \varepsilon^{(m)} \end{bmatrix} \in \mathbf{R}^m.
$$

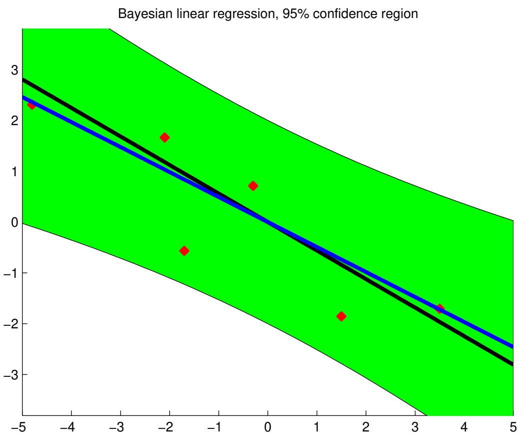


Figure 1: Bayesian linear regression for a one-dimensional linear regression problem, $y^{(i)} = \theta x^{(i)} + \epsilon^{(i)}$, with $\epsilon^{(i)} \sim \mathcal{N}(0, 1)$ i.i.d. noise. The green region denotes the 95% confidence region for predictions of the model. Note that the (vertical) width of the green region is largest at the ends but narrowest in the middle. This region reflects the uncertain in the estimates for the parameter θ. In contrast, a classical linear regression model would display a confidence region of constant width, reflecting only the $\mathcal{N}(0, \sigma^2)$ noise in the outputs.

In Bayesian linear regression, we assume that a **prior distribution** over parameters is also given; a typical choice, for instance, is $\theta \sim \mathcal{N}(0, \tau^2 I)$. Using Bayes's rule, we obtain the **parameter posterior**,

$$p(\theta \mid S) = \frac{p(\theta)p(S \mid \theta)}{\int_{\theta'} p(\theta')p(S \mid \theta')d\theta'} = \frac{p(\theta)\prod_{i=1}^{m} p(y^{(i)} \mid x^{(i)}, \theta)}{\int_{\theta'} p(\theta')\prod_{i=1}^{m} p(y^{(i)} \mid x^{(i)}, \theta')d\theta'}. \tag{2}$$

Assuming the same noise model on testing points as on our training points, the "output" of Bayesian linear regression on a new test point x_* is not just a single guess "y_*", but rather an entire probability distribution over possible outputs, known as the **posterior predictive distribution**:

$$p(y_* \mid x_*, S) = \int_{\theta} p(y_* \mid x_*, \theta)p(\theta \mid S)d\theta. \tag{3}$$

For many types of models, the integrals in (2) and (3) are difficult to compute, and hence, we often resort to approximations, such as MAP estimation (see course lecture notes on "Regularization and Model Selection").

In the case of Bayesian linear regression, however, the integrals actually are tractable! In particular, for Bayesian linear regression, one can show (after much work!) that

$$\theta \mid S \sim \mathcal{N}\left(\frac{1}{\sigma^2}A^{-1}X^T\vec{y}, A^{-1}\right)$$

$$y_* \mid x_*, S \sim \mathcal{N}\left(\frac{1}{\sigma^2}x_*^T A^{-1}X^T\vec{y}, x_*^T A^{-1}x_* + \sigma^2\right)$$

where $A = \frac{1}{\sigma^2}X^TX + \frac{1}{\tau^2}I$. The derivation of these formulas is somewhat involved.[6] Nonetheless, from these equations, we get at least a flavor of what Bayesian methods are all about: the posterior distribution over the test output y_* for a test input x_* is a Gaussian distribution—this distribution reflects the uncertainty in our predictions $y_* = \theta^T x_* + \varepsilon_*$ arising from both the randomness in ε_* and the uncertainty in our choice of parameters θ. In contrast, classical probabilistic linear regression models estimate parameters θ directly from the training data but provide no estimate of how reliable these learned parameters may be (see Figure 1).

3 Gaussian processes

As described in Section 1, multivariate Gaussian distributions are useful for modeling finite collections of real-valued variables because of their nice analytical properties. **Gaussian processes** are the extension of multivariate Gaussians to infinite-sized collections of real-valued variables. In particular, this extension will allow us to think of Gaussian processes as distributions not just over random vectors but in fact distributions over **random functions**.[7]

3.1 Probability distributions over functions with finite domains

To understand how one might paramterize probability distributions over functions, consider the following simple example. Let $\mathcal{X} = \{x_1, \ldots, x_m\}$ be any finite set of elements. Now, consider the set $\mathcal{H}$ of all possible functions mapping from $\mathcal{X}$ to $\mathbf{R}$. For instance, one example of a function $h_0(\cdot) \in \mathcal{H}$ is given by

$$h_0(x_1) = 5, \quad h_0(x_2) = 2.3, \quad h_0(x_2) = -7, \quad \ldots, \quad h_0(x_{m-1}) = -\pi, \quad h_0(x_m) = 8.$$

Since the domain of any $h(\cdot) \in \mathcal{H}$ has only m elements, we can always represent $h(\cdot)$ compactly as an m-dimensional vector, $\vec{h} = \begin{bmatrix} h(x_1) & h(x_2) & \cdots & h(x_m) \end{bmatrix}^T$. In order to specify a probability distribution over functions $h(\cdot) \in \mathcal{H}$, we must associate some "probability density" with each function in $\mathcal{H}$. One natural way to do this is to exploit the one-to-one correspondence between functions $h(\cdot) \in \mathcal{H}$ and their vector representations, $\vec{h}$. In particular, if we specify that $\vec{h} \sim \mathcal{N}(\vec{\mu}, \sigma^2 I)$, then this in turn implies a probability distribution over functions $h(\cdot)$, whose probability density function is given by

$$p(h) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}(h(x_i) - \mu_i)^2\right).$$

[6] For the complete derivation, see, for instance, [1]. Alternatively, read the Appendices, which gives a number of arguments based on the "completion-of-squares" trick, and derive this formula yourself!

[7] Let $\mathcal{H}$ be a class of functions mapping from $\mathcal{X} \to \mathcal{Y}$. A random function $h(\cdot)$ from $\mathcal{H}$ is a function which is randomly drawn from $\mathcal{H}$, according to some probability distribution over $\mathcal{H}$. One potential source of confusion is that you may be tempted to think of random functions as functions whose outputs are in some way stochastic; this is not the case. Instead, a random function $h(\cdot)$, once selected from $\mathcal{H}$ probabilistically, implies a deterministic mapping from inputs in $\mathcal{X}$ to outputs in $\mathcal{Y}$.

In the example above, we showed that probability distributions over functions with finite domains can be represented using a finite-dimensional multivariate Gaussian distribution over function outputs $h(x_1), \ldots, h(x_m)$ at a finite number of input points $x_1, \ldots, x_m$. How can we specify probability distributions over functions when the domain size may be infinite? For this, we turn to a fancier type of probability distribution known as a Gaussian process.

3.2 Probability distributions over functions with infinite domains

A stochastic process is a collection of random variables, $\{h(x) : x \in \mathcal{X}\}$, indexed by elements from some set $\mathcal{X}$, known as the index set.[8] A **Gaussian process** is a stochastic process such that any finite subcollection of random variables has a multivariate Gaussian distribution.

In particular, a collection of random variables $\{h(x) : x \in \mathcal{X}\}$ is said to be drawn from a Gaussian process with **mean function** $m(\cdot)$ and **covariance function** $k(\cdot, \cdot)$ if for any finite set of elements $x_1, \ldots, x_m \in \mathcal{X}$, the associated finite set of random variables $h(x_1), \ldots, h(x_m)$ have distribution,

$$\begin{bmatrix} h(x_1) \\ \vdots \\ h(x_m) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(x_1) \\ \vdots \\ m(x_m) \end{bmatrix}, \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_m) \\ \vdots & \ddots & \vdots \\ k(x_m, x_1) & \cdots & k(x_m, x_m) \end{bmatrix} \right).$$

We denote this using the notation,

$$h(\cdot) \sim \mathcal{GP}(m(\cdot), k(\cdot, \cdot)).$$

Observe that the mean function and covariance function are aptly named since the above properties imply that

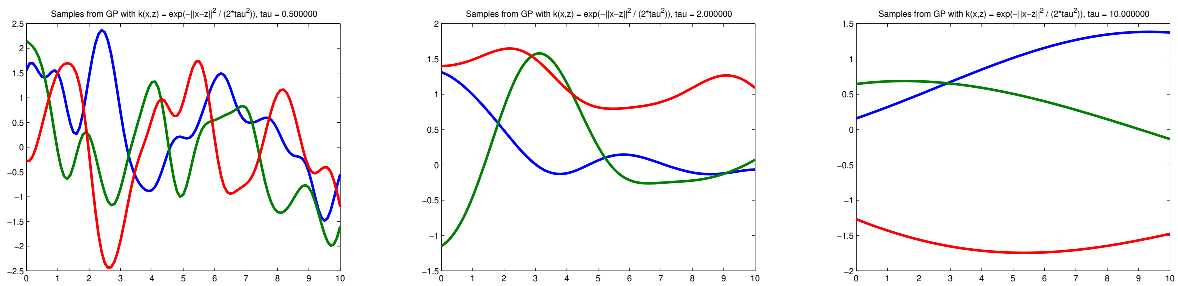
$$\begin{aligned} m(x) &= E[x] \\ k(x, x') &= E[(x - m(x))(x' - m(x')). \end{aligned}$$

for any $x, x' \in \mathcal{X}$.

Intuitively, one can think of a function $h(\cdot)$ drawn from a Gaussian process prior as an extremely high-dimensional vector drawn from an extremely high-dimensional multivariate Gaussian. Here, each dimension of the Gaussian corresponds to an element x from the index set $\mathcal{X}$, and the corresponding component of the random vector represents the value of $h(x)$. Using the marginalization property for multivariate Gaussians, we can obtain the marginal multivariate Gaussian density corresponding to any finite subcollection of variables.

What sort of functions $m(\cdot)$ and $k(\cdot, \cdot)$ give rise to valid Gaussian processes? In general, any real-valued function $m(\cdot)$ is acceptable, but for $k(\cdot, \cdot)$, it must be the case that for any

[8] Often, when $\mathcal{X} = \mathbf{R}$, one can interpret the indices $x \in \mathcal{X}$ as representing times, and hence the variables $h(x)$ represent the temporal evolution of some random quantity over time. In the models that are used for Gaussian process regression, however, the index set is taken to be the input space of our regression problem.



(a) (b) (c)

Figure 2: Samples from a zero-mean Gaussian process prior with $k_{SE}(\cdot, \cdot)$ covariance function, using (a) $\tau = 0.5$, (b) $\tau = 2$, and (c) $\tau = 10$. Note that as the bandwidth parameter τ increases, then points which are farther away will have higher correlations than before, and hence the sampled functions tend to be smoother overall.

set of elements $x_1, \ldots, x_m \in \mathcal{X}$, the resulting matrix

$$K = \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_m) \\ \vdots & \ddots & \vdots \\ k(x_m, x_1) & \cdots & k(x_m, x_m) \end{bmatrix}$$

is a valid covariance matrix corresponding to some multivariate Gaussian distribution. A standard result in probability theory states that this is true provided that K is positive semidefinite. Sound familiar?

The positive semidefiniteness requirement for covariance matrices computed based on arbitrary input points is, in fact, identical to Mercer's condition for kernels! A function $k(\cdot, \cdot)$ is a valid kernel provided the resulting kernel matrix K defined as above is always positive semidefinite for any set of input points $x_1, \ldots, x_m \in \mathcal{X}$. Gaussian processes, therefore, are kernel-based probability distributions in the sense that any valid kernel function can be used as a covariance function!

3.3 The squared exponential kernel

In order to get an intuition for how Gaussian processes work, consider a simple zero-mean Gaussian process,

$$h(\cdot) \sim \mathcal{GP}(0, k(\cdot, \cdot)).$$

defined for functions $h : \mathcal{X} \to \mathbf{R}$ where we take $\mathcal{X} = \mathbf{R}$. Here, we choose the kernel function $k(\cdot, \cdot)$ to be the **squared exponential**[9] kernel function, defined as

$$k_{SE}(x, x') = \exp\left(-\frac{1}{2\tau^2}||x - x'||^2\right)$$

[9]In the context of SVMs, we called this the Gaussian kernel; to avoid confusion with "Gaussian" processes, we refer to this kernel here as the squared exponential kernel, even though the two are formally identical.

for some $\tau > 0$. What do random functions sampled from this Gaussian process look like?

In our example, since we use a zero-mean Gaussian process, we would expect that for the function values from our Gaussian process will tend to be distributed around zero. Furthermore, for any pair of elements $x, x' \in \mathcal{X}$.

- $h(x)$ and $h(x')$ will tend to have high covariance x and x' are "nearby" in the input space (i.e., $||x - x'|| = |x - x'| \approx 0$, so $\exp(-\frac{1}{2\tau^2}||x - x'||^2) \approx 1$).
- $h(x)$ and $h(x')$ will tend to have low covariance when x and x' are "far apart" (i.e., $||x - x'|| \gg 0$, so $\exp(-\frac{1}{2\tau^2}||x - x'||^2) \approx 0$).

More simply stated, functions drawn from a zero-mean Gaussian process prior with the squared exponential kernel will tend to be "locally smooth" with high probability; i.e., nearby function values are highly correlated, and the correlation drops off as a function of distance in the input space (see Figure 2).

4 Gaussian process regression

As discussed in the last section, Gaussian processes provide a method for modelling probability distributions over functions. Here, we discuss how probability distributions over functions can be used in the framework of Bayesian regression.

4.1 The Gaussian process regression model

Let $S = \{(x^{(i)}, y^{(i)})\}_{i=1}^m$ be a training set of i.i.d. examples from some unknown distribution. In the Gaussian process regression model,

$$y^{(i)} = h(x^{(i)}) + \varepsilon^{(i)}, \qquad i = 1, \dots, m$$

where the $\varepsilon^{(i)}$ are i.i.d. "noise" variables with independent $\mathcal{N}(0, \sigma^2)$ distributions. Like in Bayesian linear regression, we also assume a **prior distribution** over functions $h(\cdot)$; in particular, we assume a zero-mean Gaussian process prior,

$$h(\cdot) \sim \mathcal{GP}(0, k(\cdot, \cdot))$$

for some valid covariance function $k(\cdot, \cdot)$.

Now, let $T = \{(x_*^{(i)}, y_*^{(i)})\}_{i=1}^{m_*}$ be a set of i.i.d. testing points drawn from the same unknown distribution as S.[10] For notational convenience, we define

$$X = \begin{bmatrix} — & (x^{(1)})^T & — \\ — & (x^{(2)})^T & — \\ & \vdots & \\ — & (x^{(m)})^T & — \end{bmatrix} \in \mathbf{R}^{m\times n} \qquad \vec{h} = \begin{bmatrix} h(x^{(1)}) \\ h(x^{(2)}) \\ \vdots \\ h(x^{(m)}) \end{bmatrix}, \qquad \vec{\varepsilon} = \begin{bmatrix} \varepsilon^{(1)} \\ \varepsilon^{(2)} \\ \vdots \\ \varepsilon^{(m)} \end{bmatrix}, \quad \vec{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix} \in \mathbf{R}^m,$$

$$X_* = \begin{bmatrix} — & (x_*^{(1)})^T & — \\ — & (x_*^{(2)})^T & — \\ & \vdots & \\ — & (x_*^{(m_*)})^T & — \end{bmatrix} \in \mathbf{R}^{m_*\times n} \quad \vec{h}_* = \begin{bmatrix} h(x_*^{(1)}) \\ h(x_*^{(2)}) \\ \vdots \\ h(x_*^{(m_*)}) \end{bmatrix}, \quad \vec{\varepsilon}_* = \begin{bmatrix} \varepsilon_*^{(1)} \\ \varepsilon_*^{(2)} \\ \vdots \\ \varepsilon_*^{(m)} \end{bmatrix}, \quad \vec{y}_* = \begin{bmatrix} y_*^{(1)} \\ y_*^{(2)} \\ \vdots \\ y_*^{(m_*)} \end{bmatrix} \in \mathbf{R}^{m_*}.$$

Given the training data S, the prior $p(h)$, and the testing inputs X_*, how can we compute the posterior predictive distribution over the testing outputs $\vec{y}_*$? For Bayesian linear regression in Section 2, we used Bayes's rule in order to compute the paramter posterior, which we then used to compute posterior predictive distribution $p(y_* \mid x_*, S)$ for a new test point x_*. For Gaussian process regression, however, it turns out that an even simpler solution exists!

4.2 Prediction

Recall that for any function $h(\cdot)$ drawn from our zero-mean Gaussian process prior with covariance function $k(\cdot,\cdot)$, the marginal distribution over any set of input points belonging to $\mathcal{X}$ must have a joint multivariate Gaussian distribution. In particular, this must hold for the training and test points, so we have

$$\begin{bmatrix} \vec{h} \\ \vec{h}_* \end{bmatrix} \Bigg| X, X_* \sim \mathcal{N}\left(\vec{0}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right),$$

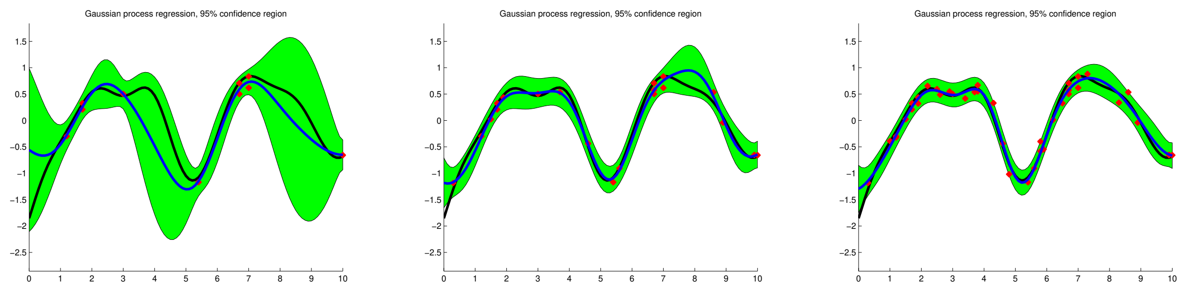
where

$$\begin{aligned} \vec{h} \in \mathbf{R}^m &\text{ such that } \vec{h} = \begin{bmatrix} h(x^{(1)}) & \cdots & h(x^{(m)}) \end{bmatrix}^T \\ \vec{h}_* \in \mathbf{R}^{m_*} &\text{ such that } \vec{h}_* = \begin{bmatrix} h(x_*^{(1)}) & \cdots & h(x_*^{(m)}) \end{bmatrix}^T \\ K(X,X) \in \mathbf{R}^{m\times m} &\text{ such that } (K(X,X))_{ij} = k(x^{(i)}, x^{(j)}) \\ K(X,X_*) \in \mathbf{R}^{m\times m_*} &\text{ such that } (K(X,X_*))_{ij} = k(x^{(i)}, x_*^{(j)}) \\ K(X_*,X) \in \mathbf{R}^{m_*\times m} &\text{ such that } (K(X_*,X))_{ij} = k(x_*^{(i)}, x^{(j)}) \\ K(X_*,X_*) \in \mathbf{R}^{m_*\times m_*} &\text{ such that } (K(X_*,X_*))_{ij} = k(x_*^{(i)}, x_*^{(j)}). \end{aligned}$$

From our i.i.d. noise assumption, we have that

$$\begin{bmatrix} \vec{\varepsilon} \\ \vec{\varepsilon}_* \end{bmatrix} \sim \mathcal{N}\left(\vec{0}, \begin{bmatrix} \sigma^2 I & \vec{0} \\ \vec{0}^T & \sigma^2 I \end{bmatrix}\right).$$

[10]We assume also that T are S are mutually independent.



(a) (b) (c)

Figure 3: Gaussian process regression using a zero-mean Gaussian process prior with $k_{SE}(\cdot, \cdot)$ covariance function (where $\tau = 0.1$), with noise level $\sigma = 1$, and (a) $m = 10$, (b) $m = 20$, and (c) $m = 40$ training examples. The blue line denotes the mean of the posterior predictive distribution, and the green shaded region denotes the 95% confidence region based on the model's variance estimates. As the number of training examples increases, the size of the confidence region shrinks to reflect the diminishing uncertainty in the model estimates. Note also that in panel (a), the 95% confidence region shrinks near training points but is much larger far away from training points, as one would expect.

The sums of independent Gaussian random variables is also Gaussian, so

$$\begin{bmatrix} \vec{y} \\ \vec{y}_* \end{bmatrix} \Bigg| X, X_* = \begin{bmatrix} \vec{h} \\ \vec{h}_* \end{bmatrix} + \begin{bmatrix} \vec{\varepsilon} \\ \vec{\varepsilon}_* \end{bmatrix} \sim \mathcal{N}\left(\vec{0}, \begin{bmatrix} K(X,X) + \sigma^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) + \sigma^2 I \end{bmatrix}\right).$$

Now, using the rules for conditioning Gaussians, it follows that

$$\vec{y_*} \mid \vec{y}, X, X_* \sim \mathcal{N}(\mu^*, \Sigma^*)$$

where

$$\begin{aligned} \mu^* &= K(X_*, X)(K(X,X) + \sigma^2 I)^{-1}\vec{y} \\ \Sigma^* &= K(X_*, X_*) + \sigma^2 I - K(X_*, X)(K(X,X) + \sigma^2 I)^{-1}K(X, X_*). \end{aligned}$$

And that's it! Remarkably, performing prediction in a Gaussian process regression model is very simple, despite the fact that Gaussian processes in themselves are fairly complicated![11]

5 Summary

We close our discussion of our Gaussian processes by pointing out some reasons why Gaussian processes are an attractive model for use in regression problems and in some cases may be preferable to alternative models (such as linear and locally-weighted linear regression):

[11]Interestingly, it turns out that Bayesian linear regression, when "kernelized" in the proper way, turns out to be exactly equivalent to Gaussian process regression! But the derivation of the posterior predictive distribution is far more complicated for Bayesian linear regression, and the effort needed to kernelize the algorithm is even greater. The Gaussian process perspective is certainly much easier!

1. As Bayesian methods, Gaussian process models allow one to quantify uncertainty in predictions resulting not just from intrinsic noise in the problem but also the errors in the parameter estimation procedure. Furthermore, many methods for model selection and hyperparameter selection in Bayesian methods are immediately applicable to Gaussian processes (though we did not address any of these advanced topics here).

2. Like locally-weighted linear regression, Gaussian process regression is non-parametric and hence can model essentially arbitrary functions of the input points.

3. Gaussian process regression models provide a natural way to introduce kernels into a regression modeling framework. By careful choice of kernels, Gaussian process regression models can sometimes take advantage of structure in the data (though, we also did not examine this issue here).

4. Gaussian process regression models, though perhaps somewhat tricky to understand conceptually, nonetheless lead to simple and straightforward linear algebra implementations.

References

[1] Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning.* MIT Press, 2006. Online: `http://www.gaussianprocess.org/gpml/`

Appendix A.1

In this example, we show how the normalization property for multivariate Gaussians can be used to compute rather intimidating multidimensional integrals without performing any real calculus! Suppose you wanted to compute the following multidimensional integral,

$$I(A, b, c) = \int_x \exp\left(-\frac{1}{2}x^T A x - x^T b - c\right) dx,$$

for some $A \in \mathbf{S}^m_{++}$, $b \in \mathbf{R}^m$, and $c \in \mathbf{R}$. Although one could conceivably perform the multidimensional integration directly (good luck!), a much simpler line of reasoning is based on a mathematical trick known as "completion-of-squares." In particular,

$$\begin{aligned}
I(A, b, c) &= \exp(-c) \cdot \int_x \exp\left(-\frac{1}{2}x^T A x - x^T A A^{-1} b\right) dx \\
&= \exp(-c) \cdot \int_x \exp\left(-\frac{1}{2}(x - A^{-1}b)^T A (x - A^{-1}b) - b^T A^{-1} b\right) dx \\
&= \exp\left(-c - b^T A^{-1} b\right) \cdot \int_x \exp\left(-\frac{1}{2}(x - A^{-1}b)^T A (x - A^{-1}b)\right) dx.
\end{aligned}$$

Defining $\mu = A^{-1}b$ and $\Sigma = A^{-1}$, it follows that $I(A, b, c)$ is equal to

$$\frac{(2\pi)^{m/2}|\Sigma|}{\exp(c + b^T A^{-1} b)} \cdot \left[\frac{1}{(2\pi)^{m/2}|\Sigma|} \int_x \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right) dx\right].$$

However, the term in brackets is identical in form to the integral of a multivariate Gaussian! Since we know that a Gaussian density normalizes, it follows that the term in brackets is equal to 1. Therefore,

$$I(A, b, c) = \frac{(2\pi)^{m/2}|A^{-1}|}{\exp(c + b^T A^{-1} b)}.$$

Appendix A.2

We derive the form of the distribution of x_A given x_B; the other result follows immediately by symmetry. Note that

$$\begin{aligned}
p(x_A \mid x_B) &= \frac{1}{\int_{x_A} p(x_A, x_B; \mu, \Sigma) dx_A} \cdot \left[\frac{1}{(2\pi)^{m/2}|\Sigma|} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right)\right] \\
&= \frac{1}{Z_1} \exp\left\{-\frac{1}{2}\left(\begin{bmatrix} x_A \\ x_B \end{bmatrix} - \begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix}\right)^T \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix} \left(\begin{bmatrix} x_A \\ x_B \end{bmatrix} - \begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix}\right)\right\}
\end{aligned}$$

where Z_1 is a proportionality constant which does not depend on x_A, and

$$\Sigma^{-1} = V = \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix}.$$

To simplify this expression, observe that

$$\begin{aligned}
&\left(\begin{bmatrix} x_A \\ x_B \end{bmatrix} - \begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix}\right)^T \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix} \left(\begin{bmatrix} x_A \\ x_B \end{bmatrix} - \begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix}\right) \\
&\qquad = (x_A - \mu_A)^T V_{AA}(x_A - \mu_A) + (x_A - \mu_A)^T V_{AB}(x_B - \mu_B) \\
&\qquad\qquad + (x_B - \mu_B)^T V_{BA}(x_A - \mu_A) + (x_B - \mu_B)^T V_{BB}(x_B - \mu_B).
\end{aligned}$$

Retaining only terms dependent on x_A (and using the fact that $V_{AB} = V_{BA}^T$), we have

$$p(x_A \mid x_B) = \frac{1}{Z_2} \exp\left(-\frac{1}{2}\left[x_A^T V_{AA} x_A - 2x_A^T V_{AA}\mu_A + 2x_A^T V_{AB}(x_B - \mu_B)\right]\right)$$

where Z_2 is a new proportionality constant which again does not depend on x_A. Finally, using the "completion-of-squares" argument (see Appendix A.1), we have

$$p(x_A \mid x_B) = \frac{1}{Z_3} \exp\left(-\frac{1}{2}(x_A - \mu')^T V_{AA}(x_A - \mu')\right)$$

where Z_3 is again a new proportionality constant not depending on x_A, and where $mu' = \mu_A - V_{AA}^{-1} V_{AB}(x_B - \mu_B)$. This last statement shows that the distribution of x_A, conditioned on x_B, again has the form of a multivariate Gaussian. In fact, from the normalization property, it follows immediately that

$$x_A \mid x_B \sim \mathcal{N}(\mu_A - V_{AA}^{-1} V_{AB}(x_B - \mu_B), V_{AA}^{-1}).$$

To complete the proof, we simply note that

$$\begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix} = \begin{bmatrix} (\Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA})^{-1} & -(\Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA})^{-1}\Sigma_{AB}\Sigma_{BB}^{-1} \\ -\Sigma_{BB}^{-1}\Sigma_{BA}(\Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA})^{-1} & (\Sigma_{BB} - \Sigma_{BA}\Sigma_{AA}^{-1}\Sigma_{AB})^{-1} \end{bmatrix}$$

follows from standard formulas for the inverse of a partitioned matrix. Substituting the relevant blocks into the previous expression gives the desired result. □