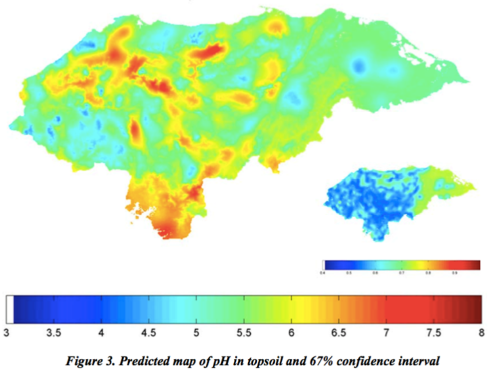
Statistical Techniques in Robotics (16-831, F12) Lecture#21 (Monday November 12)

Gaussian Processes

Lecturer: Drew Bagnell Scribe: Venkatraman Narayanan[1]*, M. Koval and P. Parashar*

1 Applications of Gaussian Processes

(a) Inverse Kinematics of a Robot Arm [1]


3
3.5
4
4.5
5
5.5
6
6.5
7
7.5
8
Figure 3. Predicted map of pH in topsoil and 67% confidence interval

(b) Predictive Soil Modeling [2]

Figure 1: Applications of Gaussian Processes

Gaussian processes can be used as a supervised learning technique for classification as well as regression. An example of a classification task would be to recognize handwritten digits, whereas an example of a regression problem would be to learn the inverse dynamics of a robot arm (Fig. 1(a)). For the latter, the task is to obtain a mapping from the state of the arm (given by the positions, velocities and accelerations of the joints) to the corresponding torques on the joints. Such a mapping can then be used to compute the torques needed to move the arm along a given trajectory. Another example is predictive soil mapping (Fig. 1(b)), where one is given a set of soil samples taken from some regions, and asked to predict the nature of soil in another region. A major benefit of using Gaussian processes to solve these problems is that they can provide confidence measures for the predictions. For instance, in the context of predictive soil mapping, one can use Gaussian processes to decide which regions should be given a higher priority for collecting soil samples, based on the uncertainty of the predictions. The following sections provide a mathematical treatment of Gaussian processes, and their application to regression problems.

1.1 Relationship to Bayesian Linear Regression

We already learned how to use Bayesian linear regression to solve the regression problem. Bayesian linear regression models the output y_i of the system as a linear combination $y = \theta^T x_i + \epsilon$ of the

[1]Some content adapted from previous scribes: Yamuna Krishnamurthy, Stephane Ross, and from *Gaussian Processes for Machine Learning* by Carl Edward Rasmussen and Christopher K. I. Williams

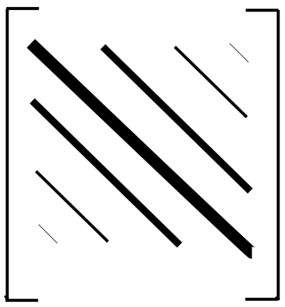


Figure 2: Magnitude of values in covariance matrix, $K_{\mathbf{xx}}$, decrease as the points are further apart.

inputs x_i, corrupted by additive, zero-mean Gaussian noise. This is ideal if we believe that the system we are trying to learn is truly linear. But, how can we apply Bayesian linear regression if we suspect that the system is non-linear?

The simplest solution is to perform the standard trick of replacing x with some non-linear features $\phi(x)$ of the data. Standard Bayesian linear regression is simply special case where $\phi(x) = x$ is the identity function. Another common case is where ϕ encodes polynomial combinations of the features in x. Just as in many machine learning algorithms, we can kernelize Bayesian linear regression by writing the inference step entirely in terms of the inner product between feature vectors (i.e. the kernel function). This yields Gaussian processes regression.

Section 2.1.2 of "Gaussian Processes for Machine Learning" provides more detail about this interpretation of Gaussian processes (available online: `http://www.gaussianprocess.org/gpml/`).

2 Gaussian Process (GP)

A Gaussian process can be thought of as a Gaussian distribution over functions (thinking of functions as infinitely long vectors containing the value of the function at every input). Formally let the input space be $\mathcal{X}$ and $f : \mathcal{X} \to \mathbb{R}$ be a function from the input space to the reals. We then say that f is a Gaussian process if for any vector of inputs $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$ such that $x_i \in \mathcal{X}$ for all i, the vector of outputs $f(\mathbf{x}) = [f(x_1), f(x_2), \ldots, f(x_n)]^T$ is Gaussian distributed.

A Gaussian process is specified by a mean function $\mu : \mathcal{X} \to \mathbb{R}$, such that $\mu(x)$ is the mean of $f(x)$ and a covariance/kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that $k(x_i, x_j)$ is the covariance between $f(x_i)$ and $f(x_j)$. We say $f \sim GP(\mu, k)$ if for any $x_1, x_2, \ldots x_n \in \mathcal{X}$, $[f(x_1), f(x_2), \ldots, f(x_n)]^T$ is Gaussian distributed with mean $[\mu(x_1), \mu(x_2), \ldots, \mu(x_n)]^T$ and $n \times n$ covariance/kernel matrix $K_{\mathbf{xx}}$:

$$K_{\mathbf{xx}} = \begin{bmatrix} k(x_1, x_1) & k(x_1, x_2) & \ldots & k(x_1, x_n) \\ k(x_2, x_1) & k(x_2, x_2) & \ldots & k(x_2, x_n) \\ \vdots & \vdots & \vdots & \vdots \\ k(x_n, x_1) & k(x_n, x_2) & \ldots & k(x_n, x_n) \end{bmatrix}$$

The kernel function must have the following properties:

- Be symmetric. That is, $k(x_i, x_j) = k(x_j, x_i)$

- Be positive definite. That is, the kernel matrix $K_{\mathbf{xx}}$ induced by k for any set of inputs is a positive definite matrix.

Examples of some kernel functions are given below:

- Squared Exponential Kernel (Gaussian/RBF): $k(x_i, x_j) = \exp(\frac{-(x_i - x_j)^2}{2\gamma^2})$ where γ is the length scale of the kernel.
- Laplace Kernel: $k(x_i, x_j) = \exp(\frac{-|x_i - x_j|}{\gamma})$.
- Indicator Kernel: $k(x_i, x_j) = I(x_i = x_j)$, where I is the indicator function.
- Linear Kernel: $k(x_i, x_j) = x_i^T x_j$.

More complicated kernels can be constructed by adding known kernel functions together, as the sum of 2 kernel functions is also a kernel function.

A Gaussian process is a random stochastic process where correlation is introduced between neighboring samples (think of a stochastic process as a sequence of random variables). The covariance matrix $K_{\mathbf{xx}}$ has larger values, for points that are closer to each other, and smaller values for points further apart. This is illustrated in Fig. 2. The thicker the line, the larger the values. This is because the points are correlated by the difference in their means and their variances. If they are highly correlated, then their means are almost same and their covariance is high.

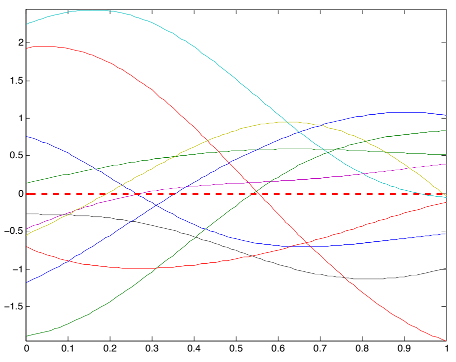
Note: If the number of nodes in a neural network are made infinite, then not only does the network become tractable but in fact it starts behaving as a Gaussian Process!

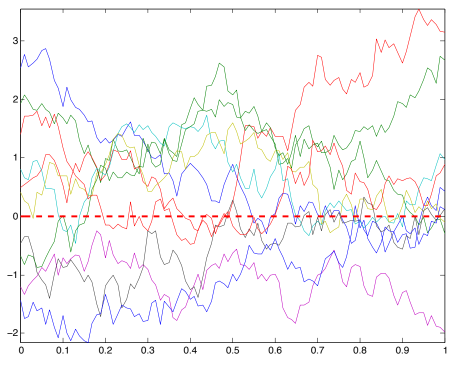
2.1 Visualizing samples from a Gaussian process

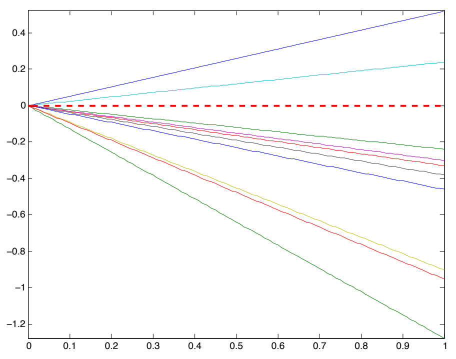
To actually plot samples from a Gaussian process, one can adopt the following procedure:

1. Define the mean function and kernel for the GP. For instance, $\mu = 0$, and $k(x_i, x_j) = \exp(\frac{-(x_i - x_j)^2}{2\gamma^2})$, with $\gamma = 0.5$.
2. Sample inputs x_i; example: $x_i = \varepsilon \times i$, $i = 0, 1, \ldots, \frac{1}{\varepsilon}$
3. Compute the kernel matrix Σ. For the example with $\varepsilon = 0.1$, we would have a 11×11 matrix.
4. Sample from the multivariate Gaussian distribution $N(\mathbf{0}, \Sigma)$
5. Plot the samples

Fig. 3 shows examples of samples drawn from a Gaussian process, for different choice of kernels and $\varepsilon = 0.01$.


(a) Gaussian kernel, $\gamma = 0.5$


(b) Laplace kernel, $\gamma = 0.5$


(c) Linear kernel

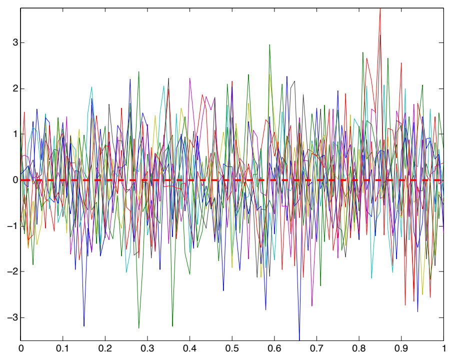
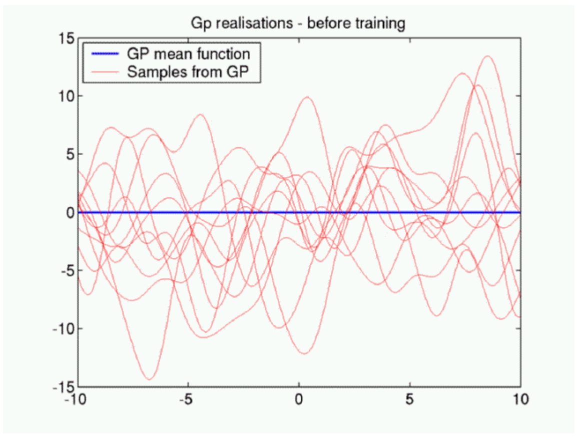
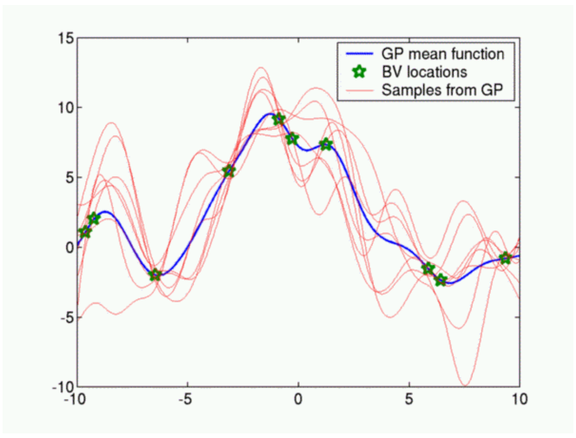

(d) Indicator kernel

Figure 3: Samples from a Gaussian process



(a) Samples from a zero-mean GP prior



(b) Samples from the posterior after a few observations

Figure 4: Gaussian process inference

3 Inference

Gaussian processes are useful as priors over functions for doing non-linear regression. In Fig. 4(a), we see a number of sample functions drawn at random from a prior distribution over functions specified by a particular Gaussian process, which favours smooth functions. This prior is taken to represent our prior belief over the kinds of functions we expect to observe, before seeing any data. Note that $\frac{1}{N}\sum_{i=1}^{N} f_i(x) \to \mu_f(x) = 0$, as $N \to \infty$.

Now, given a set of observed inputs and corresponding output values $(x_1, f(x_1))$, $(x_2, f(x_2))$, $\ldots, (x_n, f(x_n))$, and a Gaussian process prior on f, $f \sim GP(\mu, k)$, we would like to compute the posterior over the value $f(x^*)$ at any query input x^*. Figure 4(b) shows sample functions drawn from the posterior, given some observed (x, y) pairs. We see that the sample functions from the posterior pass close to the observed values, but vary a lot in regions where there are no observations. This shows that uncertainty is reduced near the observed values.

3.1 Computing the Posterior

The posterior can be derived similarly to how the update equations for the Kalman filter were derived. First, we will find the joint distribution of $[f(x^*), f(x_1), f(x_2), \ldots, f(x_n)]^T$, and then use the conditioning rules for a Gaussian to compute the conditional distribution of $f(x^*)|f(x_1), \ldots, f(x_n)$.

Assume for now that the prior mean function $\mu = 0$. By definition of the Gaussian process, the joint distribution $[f(x^*), f(x_1), f(x_2), \ldots, f(x_n)]^T$ is a Gaussian:

$$\begin{bmatrix} f(x^*) \\ f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} k(x^*, x^*) & k(x^*, \mathbf{x})^T \\ k(x^*, \mathbf{x}) & K_{\mathbf{xx}} \end{bmatrix} \right)$$

where $K_{\mathbf{xx}}$ is the kernel matrix defined previously, and

$$k(x^*, \mathbf{x}) = \begin{bmatrix} k(x^*, x_1) \\ k(x^*, x_2) \\ \vdots \\ k(x^*, x_n) \end{bmatrix}$$

Using the conditioning rules we derived for a Gaussian, the posterior for $f(x^*)$ is:

$$f(x^*)|f(\mathbf{x}) \sim N\left(k(x^*, \mathbf{x})^T K_{\mathbf{xx}}^{-1} f(\mathbf{x}), k(x^*, x^*) + k(x^*, \mathbf{x})^T K_{\mathbf{xx}}^{-1} k(x^*, \mathbf{x})\right)$$

The posterior mean $\mathbb{E}(f(x^*)|f(\mathbf{x}))$ can be interpreted in two ways. We could group the last two terms $K_{\mathbf{xx}}^{-1} f(\mathbf{x})$ together and represent the posterior mean as linear combination of the kernel function values:

$$\mathbb{E}(f(x^*)|f(\mathbf{x})) = \sum_{i=1}^{n} \alpha_i k(x^*, x_i)$$

for $\boxed{\alpha = K_{\mathbf{xx}}^{-1} f(\mathbf{x})}$. This means we can compute the mean without explicitly inverting K, by solving $K\alpha = f(\mathbf{x})$ instead. Similarly, by grouping the first two terms $k(x^*, \mathbf{x})^T K_{\mathbf{xx}}^{-1}$, the posterior mean can be represented as a linear combination of the observed function values:

$$\mathbb{E}(f(x^*)|f(\mathbf{x})) = \sum_{i=1}^{n} \beta_i f(x_i)$$

for $\beta = k(x^*, \mathbf{x})^T K^{-1}$.

3.2 Non-zero mean prior

If the prior mean function is non-zero, we can still use the previous derivation by noting that if $f \sim GP(\mu, k)$, then the function $f' = f - \mu$ is a zero-mean Gaussian process $f' \sim GP(0, k)$. Hence, if we have observations from the values of f, we can subtract the prior mean function values to get observations of f', do the inference on f', and finally once we obtain the posterior on $f'(x^*)$, we can simply add back the prior mean $\mu(x^*)$ to the posterior mean, to obtain the posterior on f.

3.3 Noise in observed values

If instead of having noise-free observations of f, we observe $y(x) = f(x) + \varepsilon$, where $\varepsilon \sim N(0, \sigma^2)$ is some zero-mean Gaussian noise, then the joint distribution of $[f(x^*), y(x_1), \ldots y(x_n)]^T$ is also

Gaussian. Hence, we can apply a similar derivation to compute the posterior of $f(x^*)$. Specifically, if the prior mean function $\mu = 0$, we have that:

$$\begin{bmatrix} f(x^*) \\ y(x_1) \\ \dots \\ y(x_n) \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \\ \dots \\ 0 \end{bmatrix}, \begin{bmatrix} k(x^*, x^*) + \sigma^2 & k(x^*, \mathbf{x})^T \\ k(x^*, \mathbf{x}) & K_{\mathbf{xx}} + \sigma^2 I \end{bmatrix} \right)$$

The only difference with respect to the noise-free case is that the covariance matrix of the joint now has an extra σ^2 term on its diagonal. This is because the noise is independent for different observations, and also independent of f (so no covariance between noise terms, and between f and ε). So the posterior on $f(x^*)$ is:

$$f(x^*)|y(\mathbf{x}) \sim N\left(k(x^*, \mathbf{x})^T (K_{\mathbf{xx}} + \sigma^2 I)^{-1} y(\mathbf{x}), k(x^*, x^*) + \sigma^2 + k(x^*, \mathbf{x})^T (K_{\mathbf{xx}} + \sigma^2 I)^{-1} k(x^*, \mathbf{x})\right)$$

3.4 Choosing kernel length scale and noise variance parameters

The kernel length scale (γ) and noise variance (σ^2) parameters are chosen such that they maximize the log likelihood of the observed data. Assuming a Gaussian kernel, we obtain the most likely parameters γ and σ by solving:

$$\max_{\gamma,\sigma} \log P(y(\mathbf{x})|\gamma, \sigma) = \max_{\gamma,\sigma} \left(-\frac{1}{2} y(\mathbf{x})^T (K_{\mathbf{xx}} + \sigma^2 I)^{-1} y(\mathbf{x}) - \frac{1}{2} \log(\det(K_{\mathbf{xx}} + \sigma^2 I)) - \frac{N}{2} \log(2\pi) \right)$$

Here, the determinant term will be small when $K_{\mathbf{xx}}$ is almost diagonal; thus this maximization favors smoother kernels (larger γ). Additionally σ^2 can be chosen to have a higher value to prevent overfitting, since larger values for σ mean we trust observations lesser.

3.5 Computational complexity

One drawback of Gaussian processes is that it scales very badly with the number of observations N. Solving for the coefficients α that define the posterior mean function requires $O(N^3)$ computations. Note that Bayesian Linear Regression (BLR), which can be seen as a special case of GP with the linear kernel, has complexity of only $O(d^3)$ to find the mean weight vector, for a d dimensional input space $\mathcal{X}$. Finally, to make a prediction at any point, Gaussian process requires $O(N\hat{d})$ (where $\hat{d}$ is the complexity of evaluating the kernel), while BLR only requires $O(d)$ computations.

References

[1] Botond Bocsi, Duy Nguyen-Tuong, Lehel Csat, Bernhard Schlkopf, Jan Peters, "Learning inverse kinematics with structured prediction," in *IROS 2011: 698-703*

[2] Juan Pablo Gonzalez, Simon Cook, Thomas Oberthur, Andrew Jarvis, J. Andrew (Drew) Bagnell, and M Bernardine Dias, "Creating Low-Cost Soil Maps for Tropical Agriculture using Gaussian Processes," in *Workshop on AI in ICT for Development (ICTD) at the Twentieth International Joint Conference on Artificial Intelligence (IJCAI 2007), January, 2007.*