3. Gaussian relationships

Simon Vaughan *

July 8, 2016

In this section we discuss some of the connections between Gaussian distributions in differing dimensions.

The bivariate (two dimensional) Gaussian

The bivariate Gaussian pdf is given by

$$p(\mathbf{y}|\boldsymbol{\mu}, \Sigma) = \frac{1}{(2\pi)^{M/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{y}-\boldsymbol{\mu})\right) \tag{1}$$

where $\mathbf{y} = (y_1, y_2)^T$, $\boldsymbol{\mu} = (\mu_1, \mu_2)^T$ are the 2-vector means of each variables and Σ is the 2×2 symmetric covariance matrix

$$\Sigma = \begin{bmatrix} a & c \\ c & b \end{bmatrix} \tag{2}$$

The diagonal elements (a and b) are the variances of y_1 and y_2, respectively, e.g. $var(y_1) = \sigma^2_{y1} = a$. The off-diagonal element c is the covariance of y_1 and y_2.

Marginal and conditional densities

Writing $p(\mathbf{y}) = p(y_1, y_2)$ we see that this is the *joint* distribution of y_1 and y_2. We can use the rules of probability theory (see e.g. chapter 4 of *Scientific Inference*) to rewrite this as

$$p(y_1, y_2) = p(y_1|y_2)p(y_2) = p(y_2|y_1)p(y_1) \tag{3}$$

Here $p(y_1|y_2)$ is a *conditional distribution* for y_1 given y_2. It is a density for y_1, once we are given a value for y_2. And $p(y_2)$ is a *marginal* distribution for y_2. It is a density for y_2 irrespective of y_1.

A *marginal* distribution is obtained by 'integrating out' some of the variables. In the bivariate case we have two possible marginal distributions

$$p(y_1) = \int_{-\infty}^{+\infty} p(y_1, y_2)dy_2 \tag{4}$$

$$= \int_{-\infty}^{+\infty} p(y_1|y_2)p(y_2)dy_2 \tag{5}$$

*Email: sav2@le.ac.uk

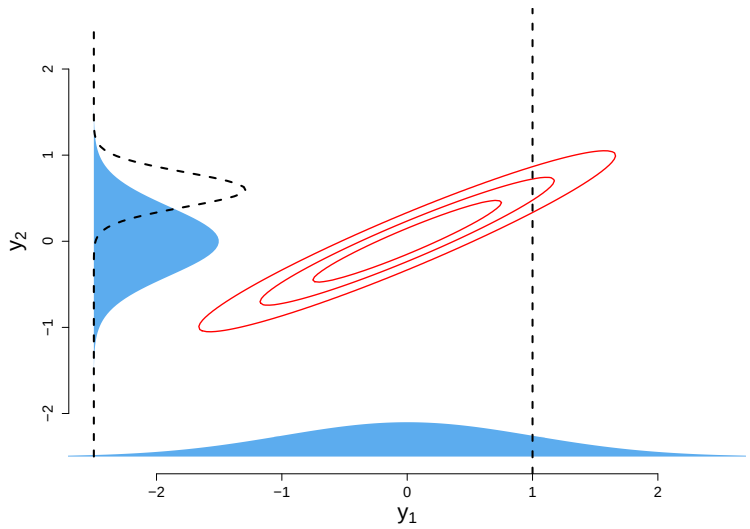


Figure 1: The contours represent a bivariate Gaussian density $p(y_1, y_2)$. The solid blue curves on the bottom and left are the marginal densities $p(y_1)$ and $p(y_2)$, respectively. The dashed curve is the conditional density $p(y_2|y_1 = 1)$, with the line $y_1 = 1$ also marked as a dashed line.

and also

$$p(y_2) = \int_{-\infty}^{+\infty} p(y_1, y_2) dy_1 \tag{6}$$

$$= \int_{-\infty}^{+\infty} p(y_2|y_1) p(y_1) dy_1 \tag{7}$$

We can think of the marginal distribution $p(y_1)$ as the probability density of y_1 irrespective of y_2. We take the conditional density, which tells us how y_1 is distributed given a value for y_2, and we average this over the distribution of y_2.

Gaussian marginal distributions

The marginal distribution of a multivariate Gaussian is another Gaussian. So its density has the same form as equation 1, but the mean $\boldsymbol{\mu}$ and covariance matrix Σ are replaced by the vector and matrix formed from subsets of the mean vector and covariance matrix of the joint density.

In the bivariate case, the marginal density for y_1 is obtained by taking $\boldsymbol{\mu} = \mu_1$ and $\Sigma = a$. We get

$$p(y_1|\boldsymbol{\mu}, \Sigma) = \frac{1}{(2\pi)|a|^{1/2}} \exp\left(-\frac{1}{2}(y_1 - \mu_1)a^{-1}(y_1 - \mu)\right) \tag{8}$$

$$= \frac{1}{2\pi\sigma_{y1}^2} \exp\left(-\frac{1}{2}\frac{(y_1 - \mu_1)^2}{\sigma_{y1}^2}\right) \tag{9}$$

and we would get a similar expression for $p(y_2)$. This is just another Gaussian distribution.

Similar rules apply for higher dimentional Gaussian distributions. If we partition an n-dimensional vector $\mathbf{y}$ into

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix} \tag{10}$$

where $\mathbf{y}_1$ is an l-dimensional vector and $\mathbf{y}_2$ is an m-dimensional vector ($n = l + m$). We can partition the mean vector and covariance matrix in a similar manner

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix} \tag{11}$$

and

$$\Sigma = \begin{bmatrix} A & C^T \\ C & B \end{bmatrix} \tag{12}$$

where A, B, C are matrices (sizes $l \times l$, $m \times m$ and $m \times l$, respectively).

The (marginal) density $p(\mathbf{y}_1)$ of the l-dimensional vector $\mathbf{y}_1$ is a Gaussian with mean vector $\boldsymbol{\mu}_1$ and covariance matrix A.

Gaussian conditional distributions

Similarly, if we have a bivariate Gaussian density $p(y_1, y_2)$, the conditional distribution of y_1 given y_2 is also a Gaussian.

$$p(y_2|y_1) = \frac{p(y_1, y_2)}{p(y_1)} \tag{13}$$

(compare with eqn 6). After some messy algebra (!) it is possible to show that – with y_1 constant – this has a Gaussian distribution with mean and variance

$$\mu_{2|1} = \mu_2 + c(y_1 - \mu_1)/a \tag{14}$$

$$\sigma^2_{2|1} = b - ca^{-1}c \tag{15}$$

Similarly, the conditional distribution of vector $\mathbf{y}_2$ given $\mathbf{y}_1$ is also a Gaussian. In this case the mean and covariance are given by

$$mean(\mathbf{y}_2|\mathbf{y}_1) = \boldsymbol{\mu}_2 + CA^{-1}(\mathbf{y}_1 - \boldsymbol{\mu}_1) \tag{16}$$

and

$$cov(\mathbf{y}_2|\mathbf{y}_1) = B - CA^{-1}C^T \tag{17}$$

(A proof of this can be found in many good books on multivariate statistics.) If we know (or assume) the mean of $\mathbf{y}$ is a constant scalar ($\boldsymbol{\mu} = \mu$) then we can write $\mathbf{y}'_1 = \mathbf{y}_1 - \mu$.

$$\mathbf{y}_2|\mathbf{y}_1 \sim N(\overline{\mathbf{y}_2}, D) \tag{18}$$

i.e. $\mathbf{y}_2$ given $\mathbf{y}_1$ is distributed as a Gaussian with mean

$$\overline{\mathbf{y}_2} = \mu + CA^{-1}\mathbf{y}'_1 \tag{19}$$

and covariance matrix

$$D = B - CA^{-1}C^T \tag{20}$$

The matrix D is known as the 'Schur complement of A in Σ' (sometimes written Σ/A).

Equation 19 might seem odd. Didn't we already assume the mean value was $\boldsymbol{\mu} = \mu$? Yes, but that is an average over all realisations of the process. Once we have observed some value for y at time t we can make a more informed prediction of the value y_* at some nearby time t_* based on our knowledge of the covariance function. If the covariance function produces slowing varying curves then our estimate of y_* should be close to y. We use our knowledge of the covariance function and the data to improve estimates of any other y_*, for this particular realisation of the process.

Also, equation 20 seems odd, in the sense that it makes no use of the known values $\mathbf{y}_1$. The output covariance D depends only on the input times of the known values ($\mathbf{t_1}$), the times at which we wish to prediction ($\mathbf{t}_2$), and the auto-covariance function. Equation 20 gives the covariance for $\mathbf{y}_2$ as the difference between two terms. The first is simply the covariance of $\mathbf{y}_2$ in the absence of any information about $\mathbf{y}_1$ (the marginal distribution of $\mathbf{y}_2$). The second term is positive, and makes use of information about the times of the known vaues ($\mathbf{t}_1$) and the autocovariance function to 'bring down the error' on $\mathbf{y}_2$ near times where the values of $\mathbf{y}_1$ provide information.

Worked example

Let's simulate a GP at $m = 500$ times.

```
# define vector of times for observations
m <- 500
t.f <- seq(-0.5, 1.5, length = m)
```

Now, we must define the ACV

```
# define covariance function
acv <- function(tau, theta) {
  A <- abs( theta[1] )
  l <- abs( theta[2] )
  acov <- A * exp(-0.5 * (tau / l)^2)
  return(acov)
}
```

Notice how this takes two inputs: `tau` is a matrix of all the $\tau_{ij} = |t_i - t_j|$ values, and `theta` is a vector listing the parameters needed to specify the ACV, i.e. $\boldsymbol{\theta} = \{A, l\}$. For 'safety' reasons we use the absolute value of these to ensure that they do not become negative, which could produce an invalid ACV.

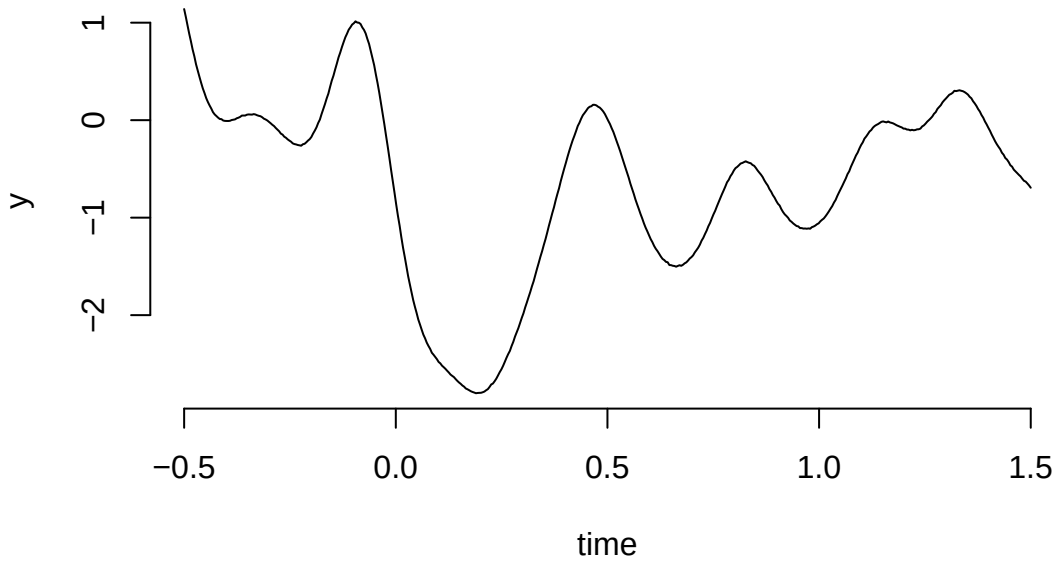
Then we define the mean $\boldsymbol{\mu} = 0$ and populate the elements of the desired covariance matrix S

```
mu <- array(0, dim = m)         # set all means to zero
tau <- abs( outer(t.f, t.f, "-") ) # compute t_j - t_i
theta <- c(1.0, 0.1)            # define ACV parameters
S <- acv(tau, theta)            # acf(tau) gives S_{ij} matrix
diag(S) <- diag(S) + 1e-5       # add a tiny 'epsilon' to help ensure pos. def.
```

Now we can use what we know about simulating GPs to produce a realisation.

```r
# produce Gaussian vector
f <- mvtnorm::rmvnorm(1, mean = mu, sigma = S, method = "chol")

# plot the 'true' function f(t)
plot(t.f, f, type = "l", bty = "n", xlab = "time", ylab = "y")
```



Now we have computed a realisation $f(t)$ of a GP with excellent sampling and precision, we now make it more like real data by keeping only a few samples and adding random noise to each value.

$$y(t_i) = f(t_i) + \epsilon_i \tag{21}$$

where t_i are a few randomly chosen times, and ϵ_i are independently distributed random variates with standard deviation σ_i: $\epsilon \sim N(0, \sigma_i)$.

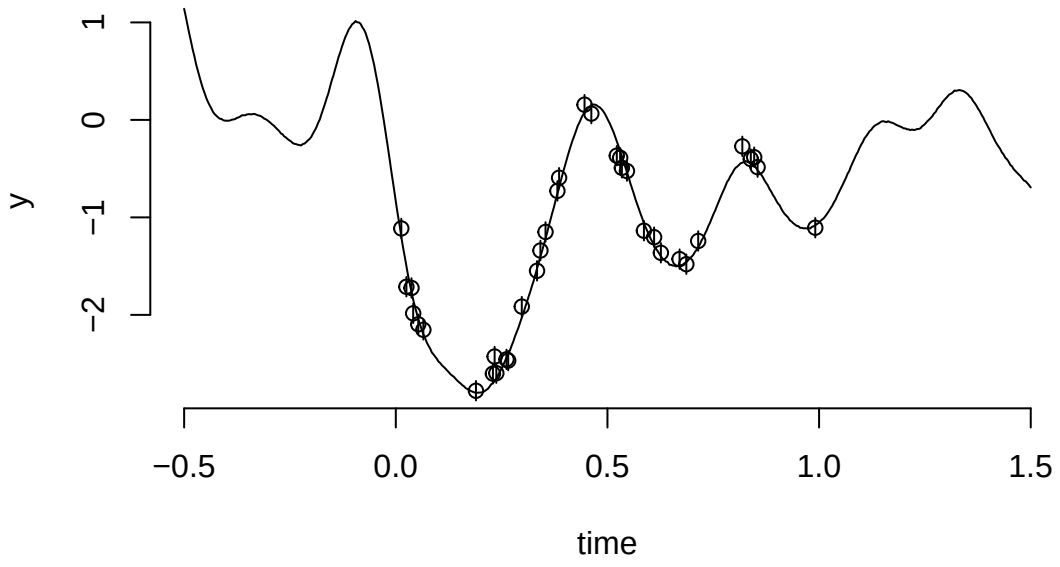
```r
# now observe f(t) only at n random times
n <- 35
indx <- sample(126:375, size = n)
indx <- sort(indx)
t <- t.f[indx]
y <- f[indx]

# now add measurement errors
dy <- rep(0.1, n)
epsilon <- rnorm(n, mean = 0, sd = dy)
y <- y + epsilon

# plot the observations
```

```
points(t, y, pch = 1)

# plot error bars
segments(t, y-dy, t, y+dy)
```



Now let's *reconstruct* $y(t)$ based on just the 'observed' (i.e. noisy) data points.

```
# times for reconstruction
  m <- 120
  t.star <- seq(-0.5, 1.5, length = m)

# compute model covariance matrix at observed delays
  tau.obs <- abs( outer(t, t, "-") )
  K <- acv(tau.obs, theta)

# compute matrix of covariances between observations and predictions
  tau.ki <- abs( outer(t.star, t, "-") )
  tau.kk <- abs( outer(t.star, t.star, "-") )
  K.ki <- acv(tau.ki, theta)
  K.ik <- t(K.ki)
  K.kk <- acv(tau.kk, theta)
  rm(tau.obs, tau.ki, tau.kk)
```

We used the `rm()` function to delete from memory the arrays that we no longer need. (Otherwise we have lots of large arrays left over from the calculation.)

Now let's check everything has the right shape. Here `K` is the matrix $K(t,t)$ $[n \times n]$ containing the ACV values between all the observation times, t. `K.ki` is the matrix $K(t_*,t)$ $[m \times n]$ containing the

ACV values between the m prediction times, t_*, and the n observation times, t. `K.ik` is the transpose of this. `K.kk` is the matrix $K(t_*, t_*)$ $[m \times m]$ containing the ACV values between all the prediction times, t_*.

```
# check the sizes of the matrices we have made
  dim(K)

## [1] 35 35

  dim(K.ki)

## [1] 120  35

  dim(K.ik)

## [1]  35 120

  dim(K.kk)

## [1] 120 120

  length(y)

## [1] 35

  length(dy)

## [1] 35
```

Now we have all the matricies and vectors defined we can do the algebra.

```
# eqn 2.20 of R&W - add the "error" term to the covariance matrix
  C <- K + dy^2 * diag(1, NCOL(K))

# compute the inverse covariance matrix
# using chol() then chol2inv() is faster than using solve()
  C.inv <- chol2inv( chol(C) )

# eqn 2.23 of R&W - predict the mean value at prediction times
  y.star <- as.vector(K.ki %*% C.inv %*% y)

# eqn 2.24 of R&W - predict the covariances at all prediction times
  cov.star <- K.kk - K.ki %*% C.inv %*% K.ik

# extract the diagonal elements from the covariance
# matrix, i.e. the variances at times t.pr. Store this as a vector.
# Use the absolute value to avoid any negative values creeping in
# due to numerical errors. Take sqrt to get standard deviation (=sigma)
  dy.star <- as.vector( sqrt( abs( diag(cov.star) ) ) )
```

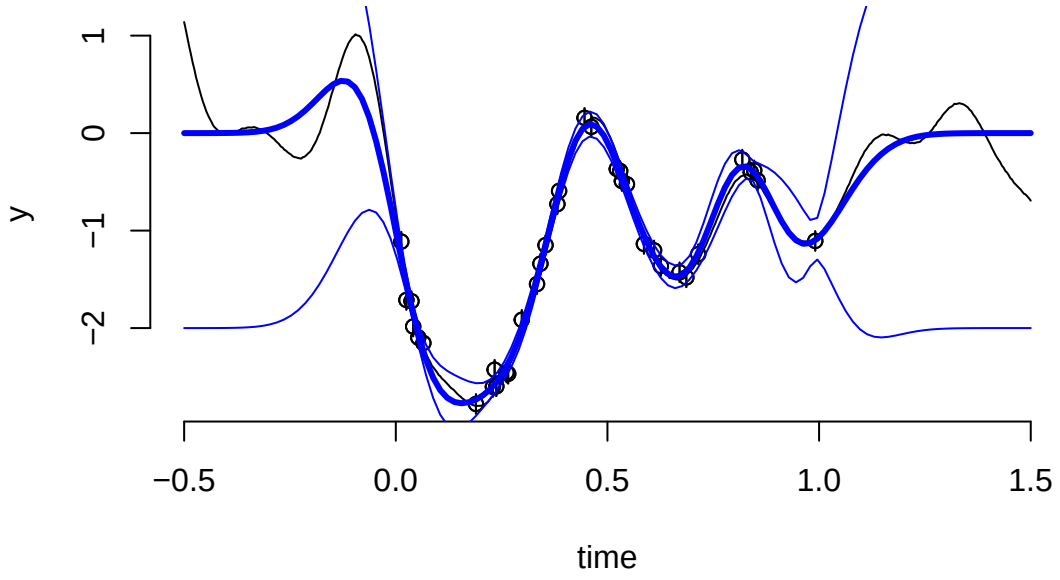
```
  # define the output product
  result <- list(t = t.star, y = y.star, dy = dy.star, var = cov.star)
```

I suspect it is possible to make the above lines of matrix algebra faster, e.g. by avoiding explictly computing the $[n \times n]$ matrix inverse C^{-1}, and by not repeating the $K(t_*, t)C^{-1}$ matrix multiplication $[m \times n] \times [n \times n]$.

Now plot the results.

```
# plot the reconstruction mean
  lines(result$t, result$y, col = "blue", lwd = 3)

# plot a 'snake' of the mean +/-2 standard deviations
  lines(result$t, result$y-2*result$dy, col = "blue", lwd = 1)
  lines(result$t, result$y+2*result$dy, col = "blue", lwd = 1)
```



Reconstructing a Gaussian Process

Why is this useful? It means that if we know some values of $\mathbf{y}$ we can make a more informed estimate of what the other values of $\mathbf{y}$ are. If we know $\mathbf{y}_2$ we can write down or compute using the conditional distribution of $\mathbf{y}_2$. In practice, this means that if we are studying some random Gaussian process, and we have a few observations at times $\mathbf{t_2}$ we can write down the distribution of un-observed values for any other times $\mathbf{t_1}$.

This is often useful for interpolation (estimate values of y between two observations), or extrapolation to times before/after all available observations (e.g. forecasting future values). I call these together *reconstruction*. We can start with a noisy and irregularly spaced series of observations, plus

knowledge of the covariance structure of the process that produced the observations, and compute the full conditional distribution (Gaussian mean and covariance) for the process at any set of times.

There is a very detailed discussion of using GP theory for reconstruction in G. B. Rybicki & W. H. Press (1992; Astrophysical Journal, v389, pp169-176).

But, the above assumes we know $ACV(\tau)$ and can therefore compute Σ for any (observed or prediction) times. How do we know what this function should be? And how to we know $\boldsymbol{\mu} = 0$? We will address these points later.