eman ta zabal zazu

Universidad del País Vasco   Euskal Herriko Unibertsitatea

Activity 1

1 Synopsis.

What we are set about to do. In a previous seminar we saw a simple example on how to estimate areas and integrals by simulation, using the so-called Monte Carlo method. In this activity you will put to good use your new skill to obtain an approximation of a not-so-obvious integral.

What you need. You need to be fully acquainted with the content of the Seminar 1 handout. You will also need access to a computer equiped with R.

2 Problem

Consider the function,

$$f(x,y) = \left(\frac{1}{\sqrt{2\pi}}\right)^2 \frac{1}{|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}\begin{pmatrix} x-m_x & y-m_y \end{pmatrix} \Sigma^{-1} \begin{pmatrix} x-m_x \\ y-m_y \end{pmatrix}\right\} \tag{1}$$

This is the bivariate normal distribution[1], with mean vector $m = \begin{pmatrix} m_x & m_y \end{pmatrix}^T$ and covariance matrix Σ. If we choose,

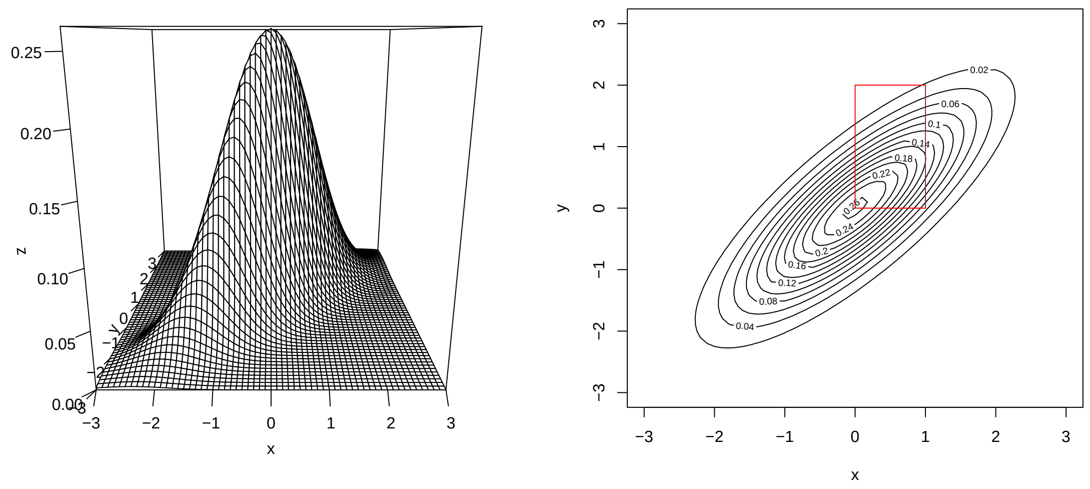
$$m = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} 1 & 0.8 \\ 0.8 & 1 \end{pmatrix} \tag{2}$$

we obtain the density function depicted in Figure 1. You are required to compute approximately the *volume* under the curve and over the region $[0,1] \times [0,2]$ or, in other words,

$$\int_o^1 dx \int_0^2 f(x,y)dy. \tag{3}$$

[1]For information on the multivariate normal, you can turn to any book on Multivariate Analysis or the Wikipedia, `http://en.wikipedia.org/wiki/Multivariate_normal_distribution`.

Figure 1: Normal bivariate density referred to in text: three dimensional view (left panel) and contour level map (right panel) showing the region over which we want the integral.



3 Hints and comments

1. Mimic the strategy outlined in Seminar 1. The volume you want is inside the "box" $B = [0,1] \times [0,2] \times [0,h]$; h is the largest possible value the function can take. You can find it easily: it is reached when the argument of the exponential in (1) takes the value zero[2]. When that happens, the value of $f(x,y)$ is:

```
> Sigma <- matrix(c(1,0.8,0.8,1),2,2)
> Sigma

     [,1] [,2]
[1,]  1.0  0.8
[2,]  0.8  1.0

> h <- ( 1 / ( sqrt(2*pi)) )^2 * (1/ sqrt(det(Sigma)) )
> h

[1] 0.2652582

> 1 * 2 * h

[1] 0.5305165
```

2. You can generate points with uniform distribution inside B quite easily: take X uniform in $[0,1]$, Y uniform in $[0,2]$ and Z uniform in $[0,h]$

3. Do it many times, and count in how many cases $f(X,Y) > Z$, where $f(X,Y)$ is the bivariate density evaluated at point X,Y.

[2]See why?

4. If you do it, say, for 10000 times and 80% of the time you found $f(X, Y) > Z$, you would reason: "The box B volume is 0.53052, and about 80% of the time one point of the box is below the surface $f(x, y)$; hence, the volume below that surface is about $0.80 \times \text{volume}(B) = 0.80 \times 0.5305 = 0.428$."

5. You can code function $f(x, y)$ in R quite easily. In our case,

```
> m     <- c(0,0)
> Sigma <-  matrix(c(1,0.8,0.8,1),2,2)
```

   Consider a point $(x, y) = (1, 2)$:

```
> x <- 1
> y <- 2
```

   The density can be obtained as follows (broken in steps for clarity):

```
> constant <-   1 / (  2*pi*sqrt(det(Sigma) ) )
> InvSigma <- solve(Sigma)
> XYmenosM <-  c(x,y) - m
> exponent <-  t(XYmenosM) %*% InvSigma %*% XYmenosM
> density  <- constant * exp(-0.5 * exponent)
```

   Note that `constant` and `InvSigma` need only be computed once; you only have to compute the bottom three lines for each point. The value of $f(x, y)$ is in `density`:

```
> density

           [,1]
[1,] 0.02177372
```

   It is even easier if you install and load package `mnormt`:

```
> library(mnormt)
```

   Then, you can compute the density simply as

```
> dmnorm(cbind(x,y), m, Sigma)

[1] 0.02177372
```

6. As already mentioned in the previous seminar, there are faster ways to compute the distribution function of a (multivariate) normal random variable: see for instance [1]. Package `mnormt`, [?], provides a function which computes $F(x; m, \Sigma)$, the multivariate normal distribution at point $x = (x_1, x_2, \ldots, x_p)$ (in your case, $p = 2$, $x_1 = x$ and $x_2 = y$). You can use this function to check the result of your simulation.

7. What do you have to show for your work to be graded? A computer printout with your code, comments interspersed explaining what you have done *and the values effectively printed by your program.* It is not enough to say "... and running this program we get 0.5". You need to produce the code and actual output, so we can check that your program effectively calculates what you say it calculates.

References

[1] M. Abramovitz and I. Stegun, editors. *Handbook of Mathematical Functions*. Dover Pub., 1965.