

# Eliciting Dependent Distributions using Multivariate Normal Copulas

*Jeremy E. Oakley*

*2018-08-16*

## 1 Introduction

We illustrate the process of incorporating dependence between uncertain quantities using a multivariate (bivariate) normal copula.

As an example, suppose a clinical trial is to be conducted for a new treatment. The trial will take place at two centres, with treatment intended to last for one year. It is believed that a proportion of patients recruited at each centre will not complete the treatment and hence drop out of the trial. Denote these two uncertain proportions by  $X_1$  and  $X_2$ . We suppose that patient characteristics are believed to be different at the second centre, such that the expert is expecting a slightly higher drop-out rate (although she is not certain that  $X_2 > X_1$ ).

## 2 Eliciting the marginal distributions

We first elicit marginal distributions for  $X_1$  and  $X_2$ . We suppose the experts states

$$P(X_1 \leq 0.12) = 0.25, P(X_1 \leq 0.15) = 0.5, P(X_1 \leq 0.20) = 0.75,$$

$$P(X_2 \leq 0.15) = 0.25, P(X_2 \leq 0.2) = 0.5, P(X_2 \leq 0.25) = 0.75.$$

We fit distributions to each of these set of judgements.

```
library(SHELF)
p <- c(0.25, 0.5, 0.75)
v1 <- c(0.12, 0.15, 0.2)
v2 <- c(0.15, 0.2, 0.25)
myfit1 <- fitdist(vals = v1, probs = p, lower = 0, upper = 1)
myfit2 <- fitdist(vals = v2, probs = p, lower = 0, upper = 1)
```

We choose to use beta distributions for each marginal, and suppose that we have gone through the process of feedback with the expert, so that she is satisfied with the two fitted distributions. The two fitted distributions are plotted below. (To aid comparison, lower and upper axes limits `x1` and `xu` are specified in the `plotfit` commands.)

```
plotfit(myfit1, d = "beta", xl = 0, xu = 0.5)
```

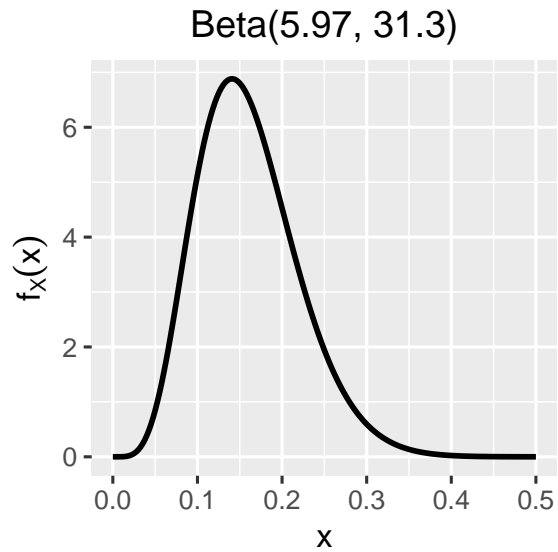


Figure 1: The fitted marginal distribution for  $X_1$ .

```
plotfit(myfit2, d = "beta", xl = 0, xu = 0.5)
```

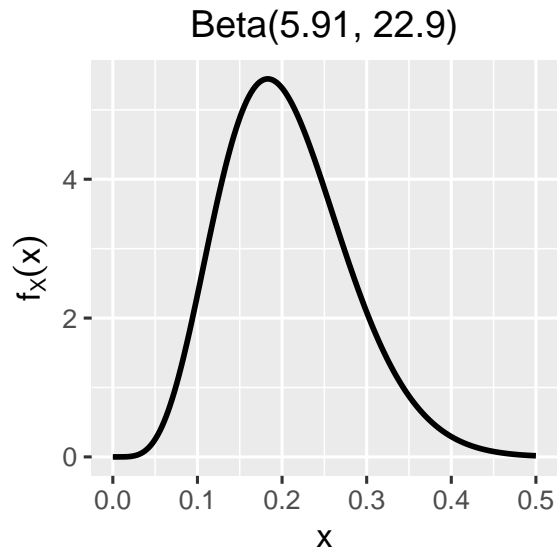


Figure 2: The fitted marginal distribution for  $X_2$ .

### 3 Incorporating dependence

We can incorporate dependence through the use of a bivariate normal copula. The general form of the joint density function of  $X_1$  and  $X_2$  is a little complex, but simulating from the joint distribution is more straightforward. The idea is as follows.

1. We can simulate a random value of  $X$  from any univariate probability distribution using inversion: we sample  $U$  from the  $U[0, 1]$  distribution, and then set our generated value of  $X$  to be the solution  $x$  of

$$P(X \leq x) = U.$$

2. We can sample *dependent* values of  $X_1$  and  $X_2$  from two separate marginal distributions by sampling *dependent* uniforms  $U_1$  and  $U_2$ , and then setting the generated values of  $X_1$  and  $X_2$  to be the solutions  $x_1$  and  $x_2$  of

$$P(X_1 \leq x_1) = U_1, P(X_2 \leq x_2) = U_2.$$

3. In the bivariate normal copula method, we generate dependent uniforms  $U_1$  and  $U_2$  by sampling  $z_1, z_2$  from the bivariate normal distribution

$$\begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} \sim N \left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & r \\ r & 1 \end{pmatrix} \right\},$$

(with the choice of  $r$  discussed shortly), and then setting  $U_1 = P(Z_1 \leq z_1)$  and  $U_2 = P(Z_2 \leq z_2)$ .

To obtain  $r$ , the expert is asked to consider her concordance probability

$$p = P(X_1 > 0.15, X_2 > 0.2, \text{ or } X_1 < 0.15, X_2 < 0.2),$$

i.e. her probability that the two uncertain proportions are either both above their medians or both below their medians. Values 0, 0.5 and 1 for this probability corresponding to perfect negative correlation, independence, and perfect positive correlation respectively. Suppose she judges  $p = 0.8$ , in that if one proportion is higher than her median, she believes the other is likely to be also. The correlation parameter  $r$  can be obtained as

$$r = \sin \left( 2\pi \left( \frac{p}{2} - 0.25 \right) \right) = 0.81.$$

The function `copulaSample` can be used to sample from the required distribution. The function generalises to  $d$  dependent variables  $X_1, \dots, X_d$  (though we don't recommend using this method for  $d > 3$ ), and takes as input a matrix of concordance probabilities, where element  $i, j$  of this matrix is the probability

$$P(X_i > m_i, X_j > m_j \text{ or } X_i < m_i, X_j < m_j),$$

where  $m_i$  and  $m_j$  are the corresponding elicited medians. It is only necessary to specify the upper triangular elements of this matrix. Note that with  $d > 2$ , the elicited pairwise concordance probabilities may not result in a positive-definite variance matrix. We discuss this further at the end.

```
conc.prob <- matrix(0, 2, 2)
conc.prob[1, 2] <- 0.8
X <- copulaSample(myfit1, myfit2, cp = conc.prob,
                  n = 1000, d = c("Beta", "Beta"))
```

The object  $X$  is a  $1000 \times 2$  matrix, with  $i$ -th column corresponding to  $X_i$ . We verify that the sample has the right properties, checking the sample quartiles and quadrant probability.

```
quantile(X[, 1], probs = c(0.25, 0.5, 0.75))
```

```
##      25%      50%      75%  
## 0.11700 0.15150 0.19525
```

```
quantile(X[, 2], probs = c(0.25, 0.5, 0.75))
```

```
##      25%      50%      75%  
## 0.150 0.195 0.251
```

```
mean((X[, 1] > 0.15 & X[, 2] > 0.2) | (X[, 1] < 0.15 & X[, 2] < 0.2))
```

```
## [1] 0.811
```

We plot the sample below.

```
library(ggplot2)  
ggplot(data.frame(X), aes(x = X1, y = X2)) +  
  geom_point(alpha = 0.1, colour = "red") +  
  geom_hline(yintercept = 0.2) +  
  geom_vline(xintercept = 0.15) +  
  labs(x=expression(X1), y = expression(X2))
```

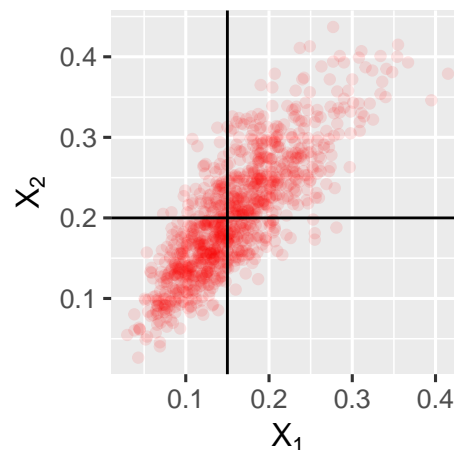


Figure 3: A sample from the joint distribution of  $X_1, X_2$ . The horizontal and vertical lines indicated the elicited medians. Note that expert has judged a probability of 0.8 of  $X_1$  and  $X_2$  being either both above or both below their median values, and so approximately 80% of the points are in the top right and bottom left quadrants.

Note that an interactive tool is available for specifying a concordance probability and viewing a corresponding joint sample for two uncertain quantities. In this example, it would be run with the command

```
elicitConcProb(myfit1, myfit2, m1 = 0.15, m2 = 0.2, d = c("Beta", "Beta"))
```

## 4 Coherent concordance probabilities and positive definite correlation matrices

With three or more uncertain quantities, it is possible to specify pair-wise concordance probabilities that are not coherent, resulting in a correlation matrix that is not positive definite. To illustrate this, suppose we have three uncertain quantities  $X_1, X_2$  and  $X_3$ , with elicited medians  $m_1, m_2$  and  $m_3$  and consider the following three statements:

1.  $X_1$  is strongly positively correlated with  $X_2$ ;
2.  $X_1$  is strongly positively correlated with  $X_3$ ;
3.  $X_2$  is strongly *negatively* correlated with  $X_3$ .

Statement 3 is not ‘consistent’ with the first two: Statements 1 and 2 imply that we expect to see either both  $X_2$  and  $X_3$  above their medians (if  $X_1 > m_1$ ), or both below their medians (if  $X_1 < m_1$ ), but Statement 3 says that we expect to see one of  $X_2, X_3$  above its median, and the other to be below.

Continuing the example, suppose we have elicited a marginal distribution for  $X_3$  (with the elicited quartiles being 0.2, 0.25 and 0.35.)

```
v3 <- c(0.2, 0.25, 0.35)
myfit3 <- fitdist(vals = v3, probs = p, lower = 0, upper = 1)
```

Defining

$$p_{i,j} = P(X_i > m_i, X_j > m_j \text{ or } X_i < m_i, X_j < m_j),$$

the expert has already stated  $p_{1,2} = 0.8$ , and we suppose she now also judges

$$p_{1,3} = 0.7, \quad p_{2,3} = 0.1$$

We set up the matrix of concordance probabilities as follows (upper diagonal elements only )

```
conc.prob <- matrix(0, 3, 3)
conc.prob[1, 2] <- 0.8
conc.prob[1, 3] <- 0.7
conc.prob[2, 3] <- 0.1
```

so we have the matrix

```
conc.prob
```

```
##      [,1] [,2] [,3]
## [1,]    0  0.8  0.7
## [2,]    0  0.0  0.1
## [3,]    0  0.0  0.0
```

We now attempt to obtain a joint sample using a multivariate normal copula

```
theta <- copulaSample(myfit1, myfit2, myfit3,
                      cp = conc.prob, n = 1000,
                      d = rep("Beta", 3))
```

```
## Elicited correlation matrix is not positive definite.
## Consider adjusting one of the concordance probabilities
## to be within the following limits.
##
##           p_{1,2}  p_{1,3}  p_{2,3}
## lower      0.2      0.1      0.5
## upper      0.4      0.3      0.9
```

The output tells us how any one of the concordance probabilities should be adjusted to make it cohere with the other two. (This feature is only available for three uncertain quantities.) For example,  $p_{2,3}$  would need to be specified in the range  $[0.5, 0.9]$ , given the values of  $p_{1,2}$  and  $p_{1,3}$ .

The valid ranges for each concordance probability (given the other two) are obtained by considering a multivariate normal random vector

$$\begin{pmatrix} Z_1 \\ Z_2 \\ Z_3 \end{pmatrix} \sim N \left\{ \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & r_{1,2} & r_{1,3} \\ r_{2,1} & 1 & r_{2,3} \\ r_{3,1} & r_{3,2} & 1 \end{pmatrix} \right\},$$

where each correlation parameter  $r_{i,j}$  is obtained from a concordance probability  $p_{i,j}$  via

$$r_{i,j} = \sin \left( 2\pi \left( \frac{p_{i,j}}{2} - 0.25 \right) \right)$$

To obtain the range for any concordance probability  $p_{i,j}$  given the other two  $p_{i,k}$ ,  $p_{j,k}$ , we note that

$$\text{Var}(Z_i | Z_j, Z_k) = 1 - (r_{i,j} \ r_{i,k}) \begin{pmatrix} 1 & r_{j,k} \\ r_{j,k} & 1 \end{pmatrix}^{-1} \begin{pmatrix} r_{i,j} \\ r_{i,k} \end{pmatrix},$$

and so, given the values of  $r_{j,k}$  and  $r_{i,k}$ , we can solve a quadratic equation to find the limits of  $r_{i,j}$  (and hence  $p_{i,j}$ ) such that  $\text{Var}(Z_i | Z_j, Z_k) > 0$ .

To confirm that  $p_{2,3}$  would need to be in the range  $[0.5, 0.9]$ , we do

```
conc.prob[2, 3] <- 0.9
theta <- copulaSample(myfit1, myfit2, myfit3,
                      cp = conc.prob, n = 1000,
                      d = rep("Beta", 3))
```

and to plot the samples we do

```
GGally::ggpairs(data.frame(theta),
  lower = list(continuous = GGally::wrap(GGally::ggally_points,
    color = "red",
    alpha = 0.1)),
  columnLabels = c("X[1]", "X[2]", "X[3]"),
  labeller = ggplot2::label_parsed)
```

