Probabilistic Risk Analysis and Bayesian Decision Theory

Marcel van Oijen* Mark Brewer†

```
knitr::opts_chunk$set( collapse=TRUE, comment=">" )
library(DiagrammeR)
library(DiagrammeRsvg)
library(geodata)
library(mvtnorm)
library(nimble)
library(rsvg)
library(terra)
library(truncnorm)
```

*Independent researcher, Edinburgh, UK - VanOijenMarcel@gmail.com
†BioSS Office, The James Hutton Institute, Craigiebuckler, Aberdeen AB15 8QH

Introduction to Bayesian Decision Theory (BDT)

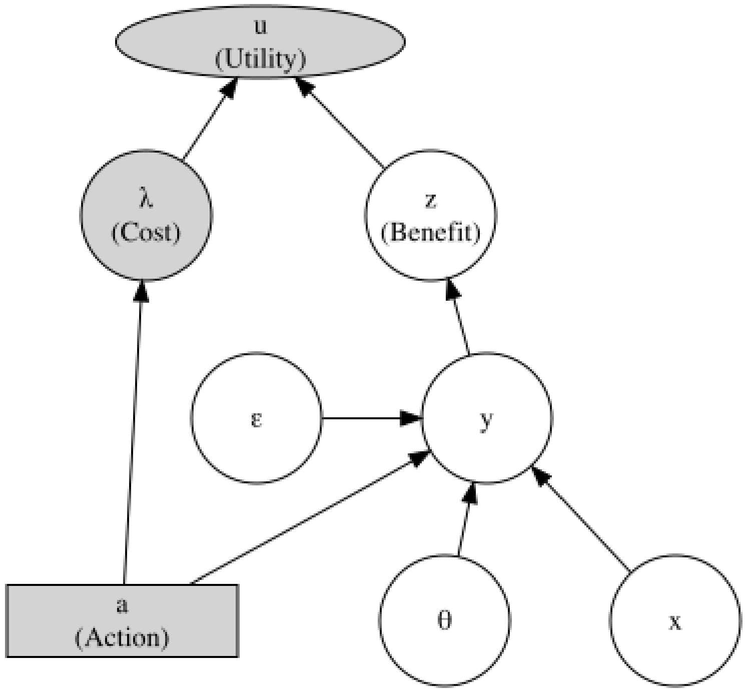

Figure 1: A graphical model for Bayesian decision theory. See text for explanation of symbols.

Example of BDT in action

```
u <- function( a, x, f, t, e, ka, ky ) {
  y       <- f(a,x,t) + e
  cost    <- ka * a
  benefit <- ky * y
  return( benefit - cost ) }
```

```
fy <- function(a,x,t) { t * (1-exp(-a-x)) }
fu <- function( a, x=1, t=1, e=0, ka=0.2, ky=1 ) {
                u( a, x, f=fy, t, e, ka, ky) }
```

$$
\begin{aligned}
x &\sim N[1,1], \\
t &\sim N[1,0.5], \\
e &\sim N[0,1], \\
ka &\sim U[0.1,0.3], \\
ky &\sim U[0.5,1.5].
\end{aligned} \tag{1}
$$

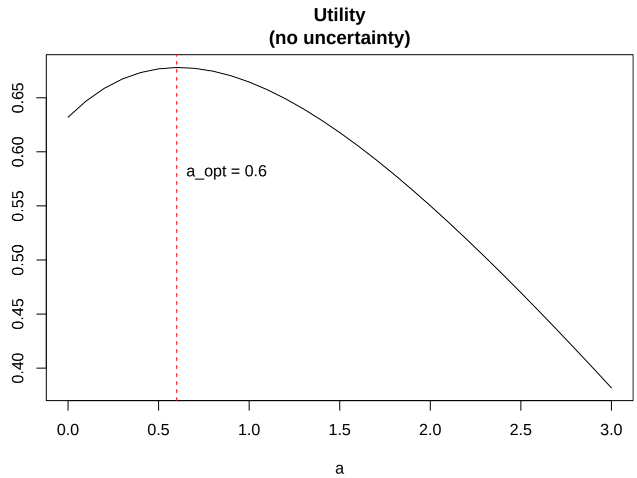

Figure 2: Utility as a function of action a for a negative exponential performance function and linear cost and benefit functions.

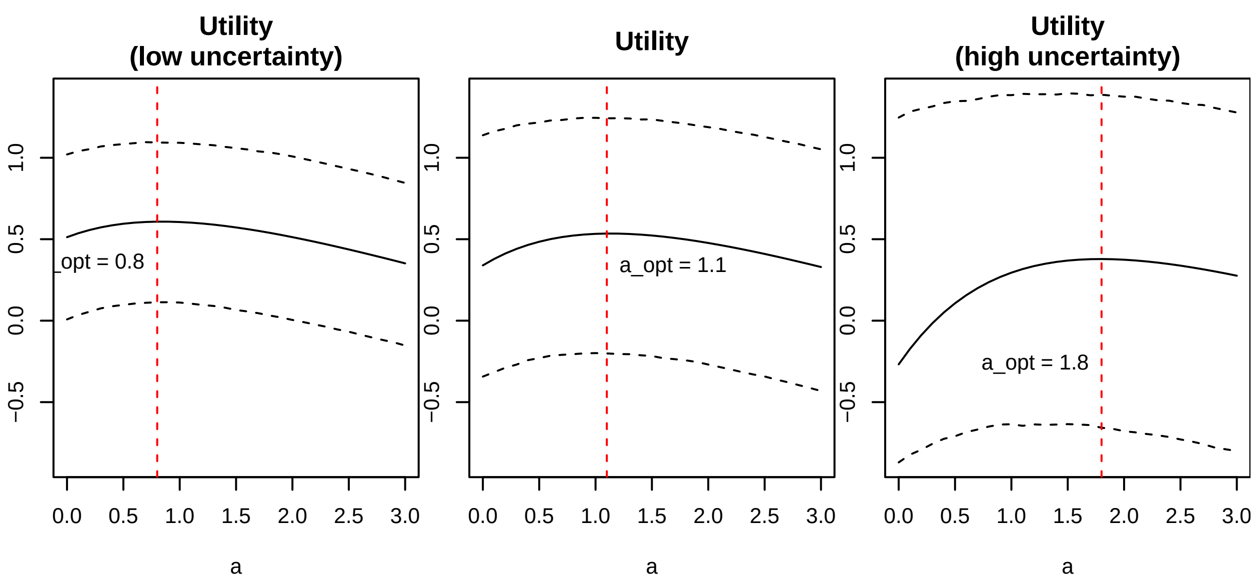

Figure 3: Utility as a function of action a for a negative exponential performance function and linear cost and benefit functions. Solid line: expectation. Dashed lines: Q25 and Q75. Middle panel: uncertainty levels (standard deviations) as indicated in the text. Left panel: uncertainties divided by 1.5. Right panel: uncertainties multiplied by 1.5.

Implementation of BDT using Bayesian Networks

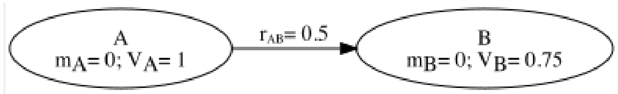

Figure 4: DAG with means and conditional variances specified in the node-ellipses and edge labelled with the regression coefficient.

Switching between the three different specifications of the multivariate Gaussian

```
precMatrix <- function( nodes, Vcond, R ){
  n <- length(nodes) ; W <- 1 / Vcond[1]
  for(k in 2:n){
    rk       <- R[ k, 1:(k-1) ]
    W_top    <- cbind( W * Vcond[k] + rk %*% t(rk), -rk ) / Vcond[k]
    W_bottom <- cbind(                           -t(rk),    1 ) / Vcond[k]
    W        <- rbind( W_top, W_bottom ) }
  rownames(W) <- colnames(W) <- nodes
  return(W) }
```

```
  VcondR <- function( W ) {
    n  <- dim(W)[1] ; Vcond <- rep( NA, n )
    Wk <- W         ; R     <- matrix( 0, nrow=n, ncol=n )
    for(k in n:2){
      ik      <- 1 : (k-1)
      Vcond[k] <- 1 / Wk[ k, k]
      R[k,ik]  <-    -Wk[ k,ik] * Vcond[k]
      Wk       <-     Wk[ik,ik] - as.matrix(R[k,ik]) %*% R[k,ik] / Vcond[k] }
    Vcond[1] <- 1 / Wk[1,1]
    return( list( Vcond=Vcond, R=zapsmall(R) ) ) }
```

Sampling from a GBN and Bayesian updating

Updating a GBN when information about nodes becomes available

```
  GaussCond <- function( mz, Sz, y ) {
    i <- 1 : ( length(mz) - length(y) )
    m  <- mz[i]   + Sz[i,-i] %*% solve(Sz[-i,-i]) %*% (y-mz[-i])
    S  <- Sz[i,i] - Sz[i,-i] %*% solve(Sz[-i,-i]) %*% Sz[-i,i]
    return( list( m=m, S=S ) ) }
```

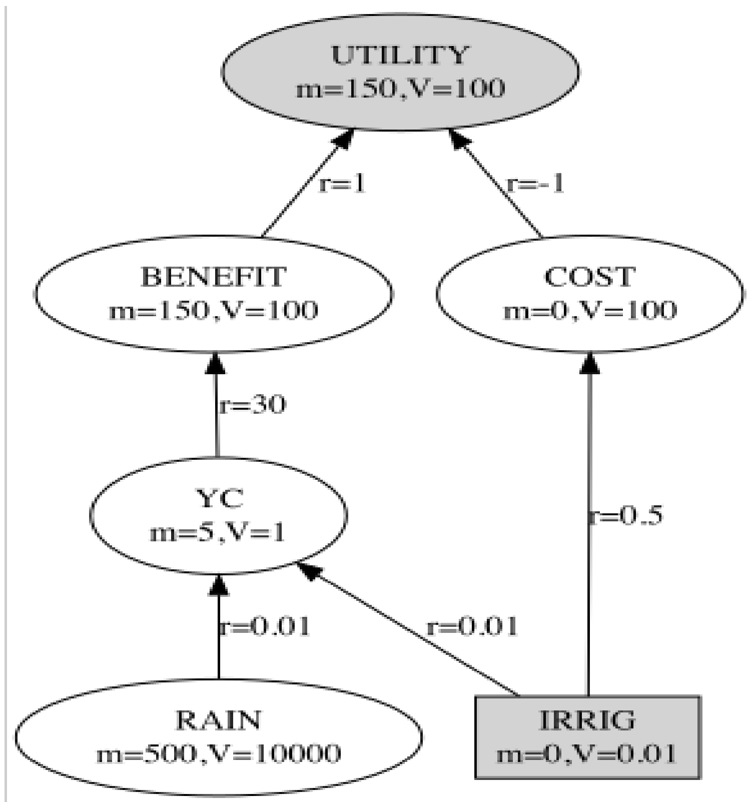

Figure 5: Example of linear BDT represented by a DAG with six nodes and six edges. Each node shows the values of the prior mean and conditional variance. Each edge is labelled with the regression coefficient.

A linear BDT example implemented as a GBN

$$\underline{\text{Original network:}}$$
$$\begin{bmatrix} RAIN \\ IRRIG \\ YC \\ BENEFIT \\ COST \\ UTILITY \end{bmatrix} : \quad \mu = \begin{bmatrix} 500 \\ 0 \\ 5 \\ 150 \\ 0 \\ 150 \end{bmatrix}; \quad \Sigma = \begin{bmatrix} 10000 & 0 & 100 & 3000 & 0 & 3000 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 100 & 0 & 2 & 60 & 0 & 60 \\ 3000 & 0 & 60 & 1900 & 0 & 1900 \\ 0 & 0 & 0 & 0 & 100 & -100 \\ 3000 & 0 & 60 & 1900 & -100 & 2100 \end{bmatrix}; \tag{2}$$
$$W = \begin{bmatrix} 0 & 0 & -0.01 & 0 & 0 & 0 \\ 0 & 100 & -0.01 & 0 & 0 & 0 \\ -0.01 & -0.01 & 10 & -0.3 & 0 & 0 \\ 0 & 0 & -0.3 & 0.02 & -0.01 & -0.01 \\ 0 & 0 & 0 & -0.01 & 0.02 & 0.01 \\ 0 & 0 & 0 & -0.01 & 0.01 & 0.01 \end{bmatrix}.$$

$$\underline{\text{Posterior network after accounting for RAIN:}}$$
$$\begin{bmatrix} IRRIG \\ YC \\ BENEFIT \\ COST \\ UTILITY \end{bmatrix} : \quad \mu = \begin{bmatrix} 0 \\ 6 \\ 180 \\ 0 \\ 180 \end{bmatrix}; \quad \Sigma = \begin{bmatrix} 0.01 & 0 & 0 & 0 & 0 \\ 0 & 1 & 30 & 0 & 30 \\ 0 & 30 & 1000 & 0 & 1000 \\ 0 & 0 & 0 & 100 & -100 \\ 0 & 30 & 1000 & -100 & 1200 \end{bmatrix}. \tag{3}$$

$$\underline{\text{Posterior network after accounting for RAIN and IRRIG:}}$$
$$\begin{bmatrix} YC \\ BENEFIT \\ COST \\ UTILITY \end{bmatrix} : \quad \mu = \begin{bmatrix} 10 \\ 300 \\ 200 \\ 100 \end{bmatrix}; \quad \Sigma = \begin{bmatrix} 1 & 30 & 0 & 30 \\ 30 & 1000 & 0 & 1000 \\ 0 & 0 & 100 & -100 \\ 30 & 1000 & -100 & 1200 \end{bmatrix}. \tag{4}$$

A linear BDT example implemented using `Nimble`

```
BDT.Code <- nimbleCode({
  RAIN     ~ dnorm( 500, sd= 100   )
  IRRIG    ~ dnorm(   0, sd=   0.1 )
  YC      <- (RAIN + IRRIG) * 0.01 + eps.YC
  BENEFIT <-  YC            * 30   + eps.BE
  COST    <-  IRRIG         * 0.5  + eps.CO
  UTILITY <-  BENEFIT - COST       + eps.UT
  eps.YC   ~ dnorm( 0, sd= 1 )
  eps.BE   ~ dnorm( 0, sd=10 )
  eps.CO   ~ dnorm( 0, sd=10 )
  eps.UT   ~ dnorm( 0, sd=10 )
} )
```

Varying IRRIG to identify the value for which E[U] is maximized

A nonlinear BDT example implemented using `Nimble`

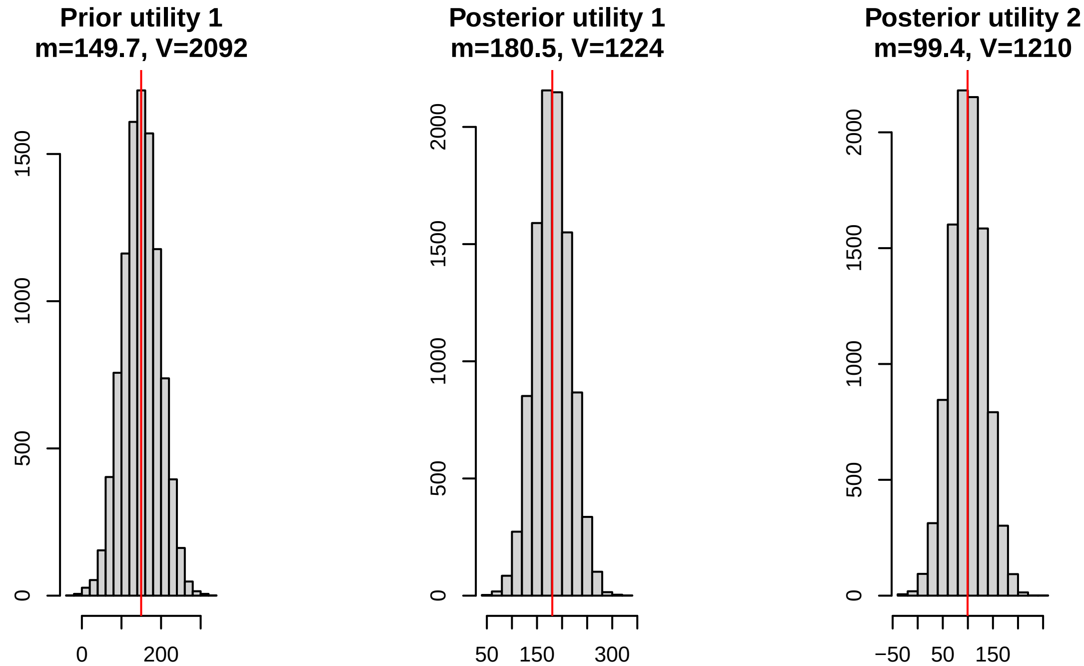

Figure 6: Linear model. Sampling from the marginal Gaussian distribution for utility using the R-package `Nimble`. Left: prior. Middle: posterior after setting rain at 600 $mm\,y^{-1}$. Right: posterior after additionally setting irrigation at 400 $mm\,y^{-1}$.

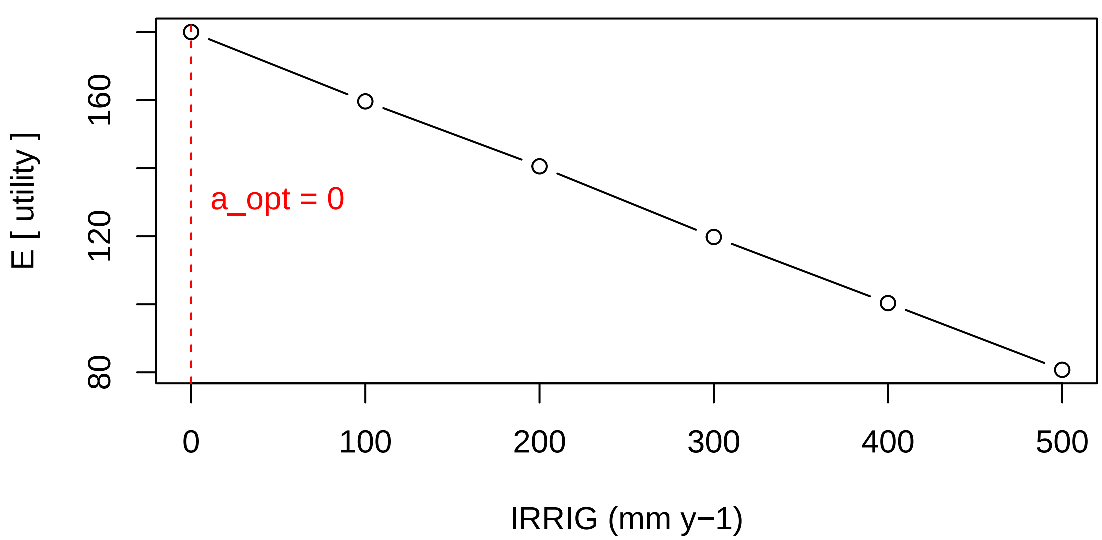

Figure 7: BDT for a linear network: identifying the level of irrigation for which the expectation of utility is maximized.

```
YC <- 50 * (1-exp(-0.001*(RAIN + IRRIG))) + eps.YC
```

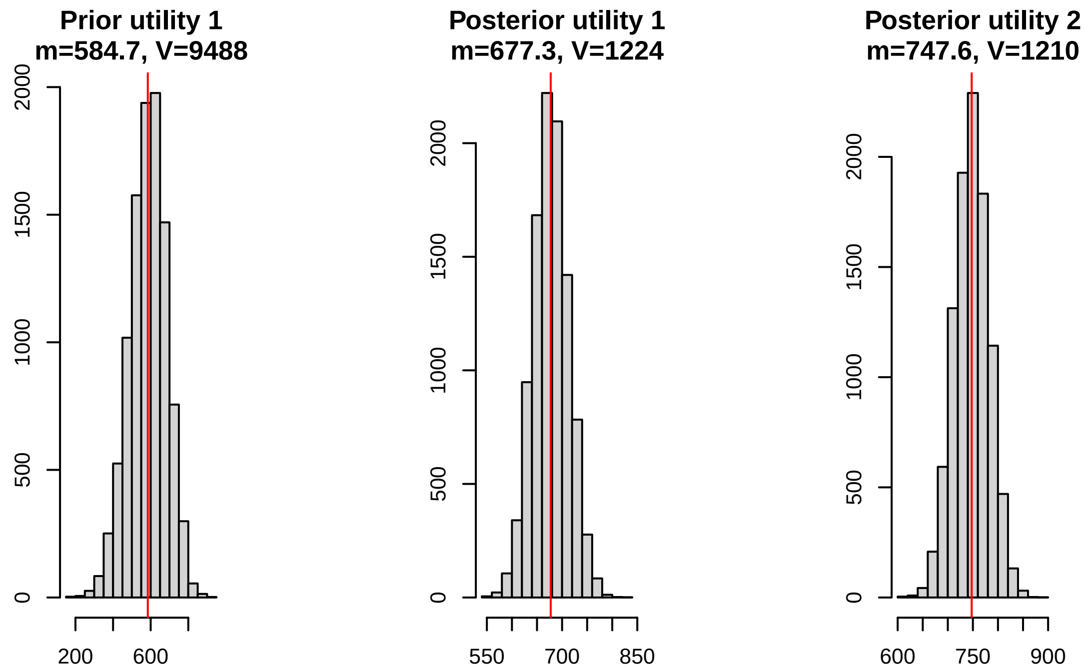

Figure 8: Nonlinear model. Sampling from the marginal Gaussian distribution for utility using the R-package `Nimble`. Left: prior. Middle: posterior after setting rain at 600 $mm\,y^{-1}$. Right: posterior after additionally setting irrigation at 400 $mm\,y^{-1}$.

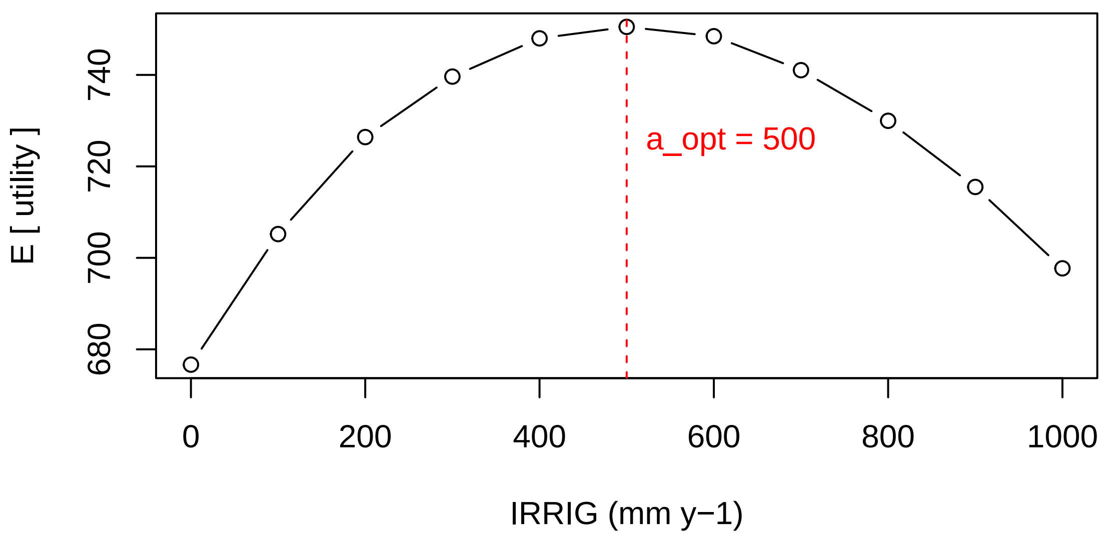

Figure 9: BDT for a nonlinear network: identifying the level of irrigation for which the expectation of utility is maximized.

A spatial example: forestry in Scotland

A decision problem: forest irrigation in Scotland

Computational demand of BDT and emulation

Data

```
spdf_GBR <- gadm( country="GBR", level=0, path="data" ) # Great Britain
```

```
# Code: see https://damariszurell.github.io/EEC-QCB/Occ3_EnvData.html

r_alt_GBR   <- elevation_30s( country='GBR', path='data', mask=F )
r_alt.hires <- crop( r_alt_GBR, ext_SCO )
r_alt       <- resample( r_alt.hires, s_prec )
```

```
# Code: see https://damariszurell.github.io/EEC-QCB/Occ3_EnvData.html
r_tree <- landcover(var='trees', path='data', download=F) # Global
r_tree <- crop( r_tree, ext_SCO )
r_tree <- mask( r_tree, r_alt.hires )
```

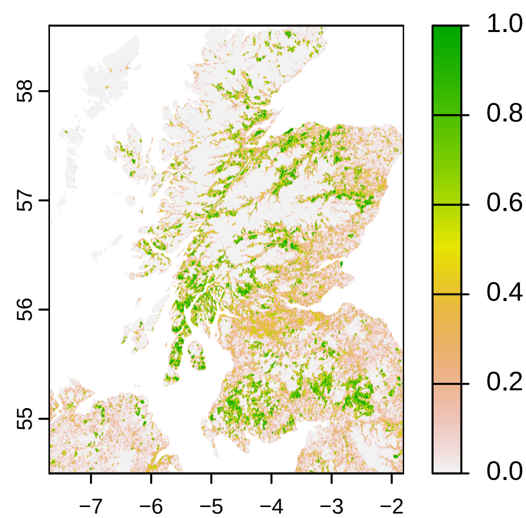

Figure 10: Tree cover in Scotland in 2020 (ESA WorldCover data set).

A simple model for forest yield class (YC)

$$YC = 10 * (1 - exp(-R/1000)) * (1 - A/1000)$$

```
YC <- function( x1, x2 ) {
  alt <- x1  ; prec <- x2
  YC  <- max(0, 10 * (1-exp(-prec/1000)) * (1-alt/1000) )
  return( YC ) }
```

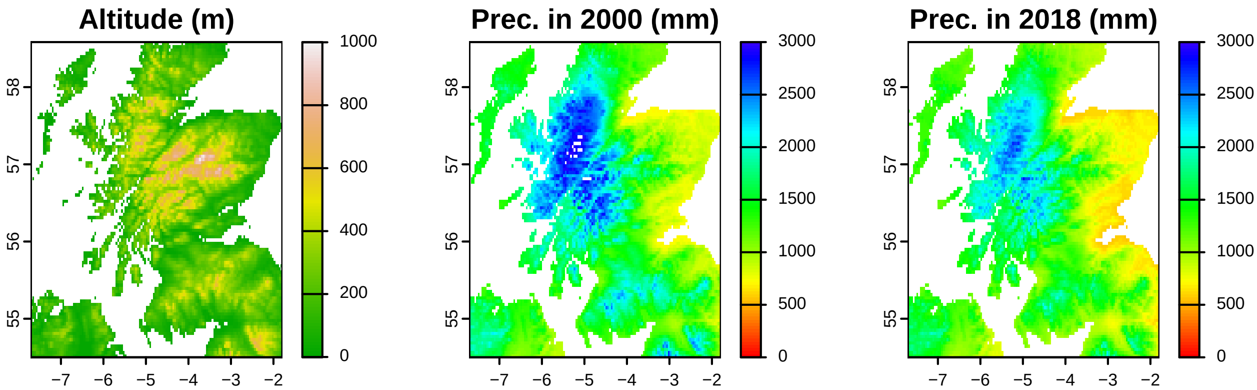

Figure 11: Scotland: altitude and precipitation in 2000 and 2018.

```
  r_prec_2000 <- s_prec[[1]]
# r_prec_2000 <- rast( s_prec, nlyrs=1 )
  r_YC <- xapp( r_alt, r_prec_2000, fun=YC )
  s_YC <- rast( sapply( 1:19, function(i){xapp( r_alt, s_prec[[i]], fun=YC )} ) )
```

Emulation

```
  n_design    <- 50
  alt_design  <- runif( n_design, 0, 1000 )
  prec_design <- runif( n_design, 0, 3000 )
  YC_design   <- sapply( 1:n_design, function(i) {
    YC( alt_design[i], prec_design[i] ) } )
```

```
  x1 <- alt_design   ; x2  <- prec_design          ; y  <- YC_design
  ny <- length(y)
  Vy <- 1            ; phi <- 1000
  x  <- cbind(x1,x2) ; dx  <- as.matrix( dist(x) ) ; Sy <- exp( -dx/phi ) * Vy
```

```
  GP.est <- function( x, y, Sy, mb, Sb, X=cbind(1,x) ) {
    Sb_y <- solve( solve(Sb) + t(X) %*% solve(Sy) %*% X )
    mb_y <- Sb_y %*% ( solve(Sb) %*% mb + t(X) %*% solve(Sy) %*% y )
    return( list( "mb_y"=mb_y, "Sb_y"=Sb_y ) ) }

  mb   <- c(0,0,0) ; Sb <- diag(1,3) ; X <- cbind(1,x)
  b_y  <- GP.est( x, y, Sy, mb=mb, Sb=Sb, X=X )
  mb_y <- b_y$mb_y; Sb_y <- b_y$Sb_y
```

Application of the emulator

```
  GP.pred <- function(x0,x,y,Sy,phi,mb_y,Sb_y,X0=c(1,x0),X=cbind(1,x)) {
    dx0   <- if( is.vector(x) ) { abs(x-x0)
             } else { sapply(1:length(y),function(i){dist(rbind(x0,x[i,]))}) }
    C0    <- Sy[1] * exp( -dx0/phi )
```

```
  m0_y <- X0 %*% mb_y - t(C0) %*% solve(Sy) %*% (X %*% mb_y - y)
     a <- X0 - t(C0) %*% solve(Sy) %*% X
  S0_y <- Sy[1] - t(C0) %*% solve(Sy) %*% C0 + a %*% Sb_y %*% t(a)
  return( list( "m0_y"=m0_y, "S0_y"=S0_y ) ) }

x0   <- c(0,0) ; X0 <- c(1,x0)
y0_y <- GP.pred( x0, x, y, Sy, phi, mb_y, Sb_y, X0=X0, X=X )
m0_y <- y0_y$m0_y ; S0_y <- y0_y$S0_y
```

```
YC_em <- function( x1, x2 ) {
  x0 <- cbind(x1,x2)
  YC <- GP.pred( x0, x, y, Sy, phi, mb_y, Sb_y,
                 X0=c(1,x0), X=cbind(1,x) )$m0_y
  return( YC ) }
YC_em.S <- function( x1, x2 ) {
  x0 <- cbind( x1, x2 )
  S  <- GP.pred( x0, x, y, Sy, phi, mb_y, Sb_y,
                 X0=c(1,x0), X=cbind(1,x) )$S0_y
  return( S ) }
```

```
r_YC_em   <- xapp( r_alt, r_prec_2000, fun=YC_em  )
r_YC_em.S <- xapp( r_alt, r_prec_2000, fun=YC_em.S  )
```

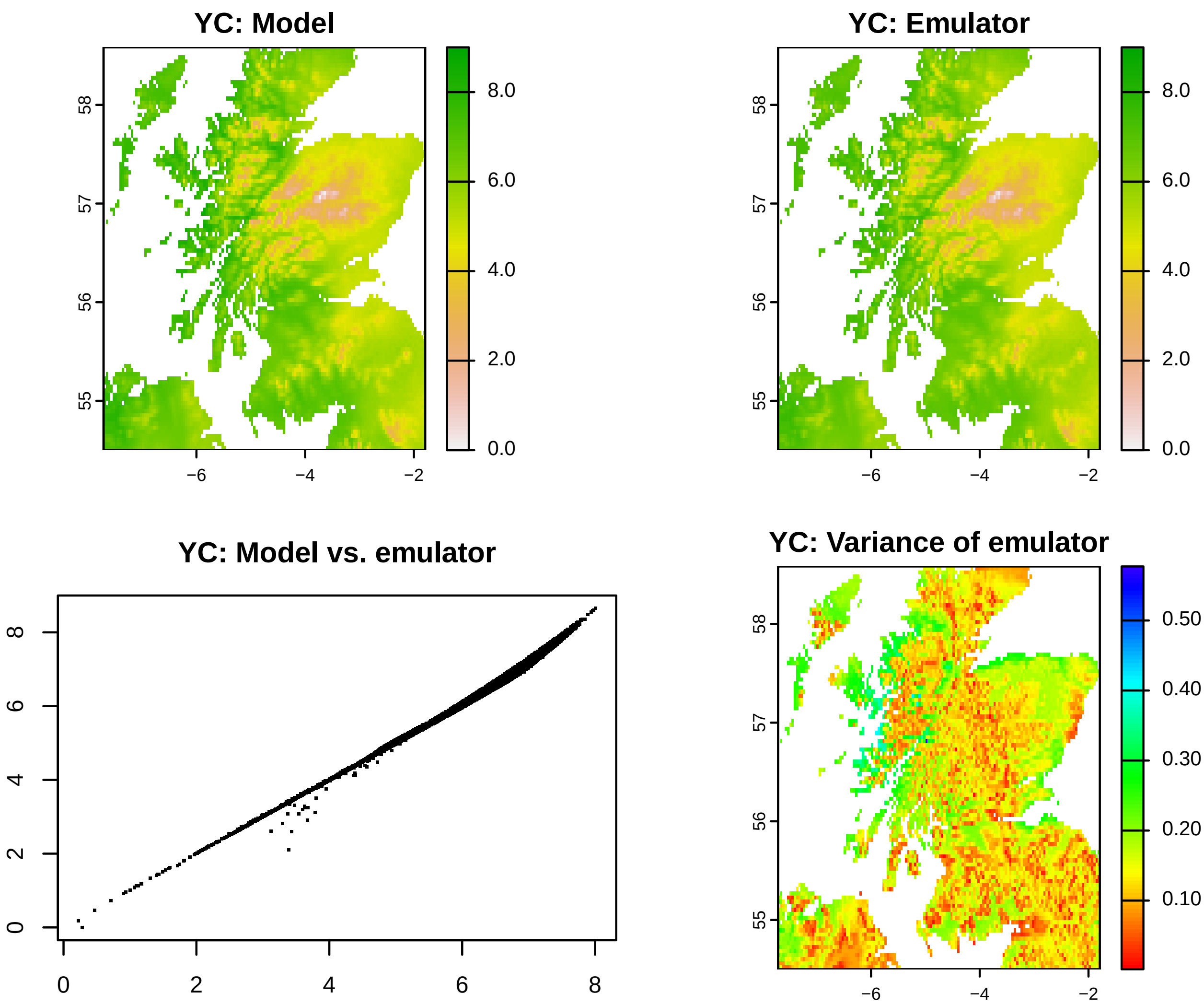

Figure 12: Top left: outputs from the original nonlinear model for yield class (YC, $m^3\, ha^{-1}\, y^{-1}$) applied to precipitation data from the year 2000. Top right: emulated YC. Bottom left: original outputs vs. emulated values. Bottom right: emulator uncertainty (kriging variance).

Spatial BDT using model and emulator

```
ky <- 30 ; r_U     <- r_YC     * ky ; r_U_em     <- r_YC_em     * ky
```

```
IRRIG          <- 500
r_YC.IRRIG     <- xapp( r_alt, r_prec_2000 + IRRIG, fun=YC    )
r_YC_em.IRRIG <- xapp( r_alt, r_prec_2000 + IRRIG, fun=YC_em )
ka             <- 0.1
r_U.IRRIG      <- r_YC.IRRIG    * ky - IRRIG * ka
r_U_em.IRRIG  <- r_YC_em.IRRIG * ky - IRRIG * ka
```

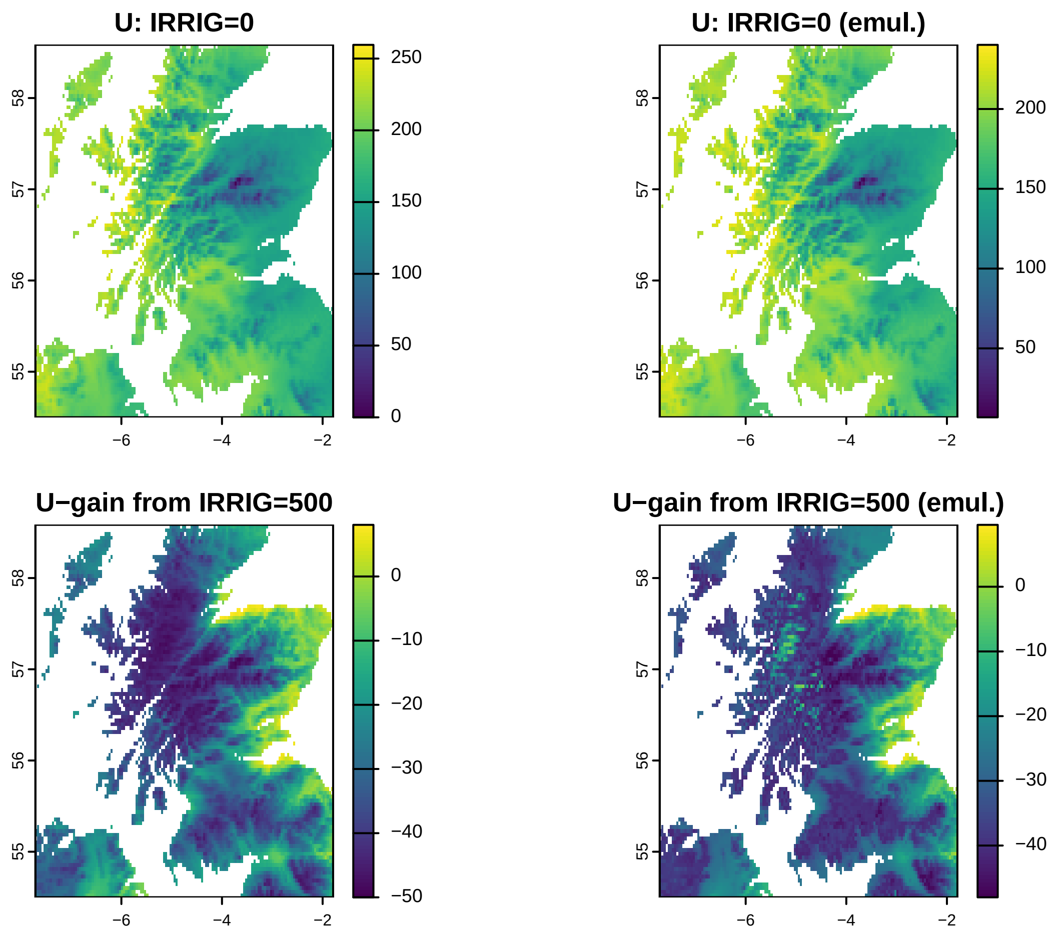

Figure 13: Top: utility in the year 2000 without irrigation. Bottom: gain in utility with irrigation of 500 $mm\,y^{-1}$. Left: original model. Right: emulator.

Multiple action levels

```
nI    <- 10 ; layers <- 1:nI ; IRRIG <- (layers-1)*50
s_U.a <- NULL
for(i in layers) {
```

```
    r_YC.a <- xapp( r_alt, r_prec_2000 + IRRIG[i], fun=YC )
    r_U.a  <- r_YC.a * ky - IRRIG[i] * ka
    s_U.a  <- c( s_U.a, r_U.a ) }
  s_U.a <- rast(s_U.a)
  a.opt <- (which.max( s_U.a ) - 1) * 50
```

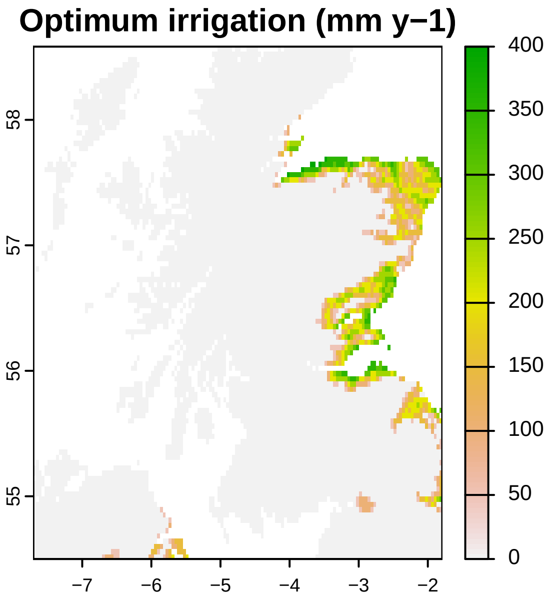

Figure 14: Outcome from a decision problem where action choice was between a large number of irrigation levels.