Expected value of more precise/context-relevant calibration target data

## Introduction

Calibration target data is one of the key elements of any calibration process. However, like any data, calibration targets are prone to precision and/or validity issues.

The Standard error (SE) is a measure of the precision of a statistical estimate. Specifically, it quantifies the variability or imprecision of a sample statistic, such as the mean or proportion, compared to the true population parameter. We quantify the Expected Value of Perfect Information (EVPI) as the value of reducing uncertainty in decision-making. The EVPI represents the maximum amount a decision-maker would be willing to pay for perfect information that eliminates all uncertainty associated with a decision, reducing the SE to zero.

Bias, on the other hand, refers to the systematic or consistent error in measurements or estimates. It can lead to inaccuracies in the results, and it can result from flaws in study design, data collection methods, or other systematic errors.

## Theoretical derivations

### The problem

#### Prior

*where*

The probability density function (PDF) for a multivariate normal distribution with a mean vector and a covariance matrix is given by:

#### Measurement

*where*

* is a matrix that transforms the vector into another vector (or scalar, depending on the context). The transformation is defined by the structure and values of . represents the transposition of the matrix . Transposing a matrix involves flipping it over its diagonal, turning the matrix’s rows into columns and vice versa.

In the current example: and . If , , and .

* is the measurement error .

The likelihood is given by:

#### Decision [Mark ??]

The decision is to use intervention if and keep the status quo otherwise.

The incremental net benefit is [Mark ??]

*where*

#### Reduced normals

To calculate the distribution of conditional on observing measurement , , note

Given the function

and the assertion that

our goal is to derive expressions for and .

##### Expanding

First, we expand both parts of :

##### Combining and Rearranging

Next, we combine these expansions and rearrange terms to form a quadratic expression in :

The goal is to rewrite in the form of . This form is a general quadratic expression which can be expanded as:

##### Comparing Coefficients

By comparing coefficients from the expanded form of and the general quadratic form, we can identify and :

#### Posterior hyperparameters

So, the posterior distribution of the is:

#### A useful integral}

Using integration by parts:

where is the density of a normal distribution and is the cdf of the standard normal distribution, and the last equation serves to define the function .

### EVPI:

The Standard error (SE) measures the precision of a statistical estimate. Specifically, it quantifies the variability or imprecision of a sample statistic, such as the mean or proportion, compared to the true population parameter.

In a set of competing options, , the optimal choice for decision-makers like NICE is that associated with the highest “utility”. The net benefit (NB) is a utility function given by the following notation for each choice or option .

is the model input parameters representing real-world quantities. Since observed data (parameters) may not be known with certainty, we are uncertain about the value(s) of and, therefore, uncertain about the . Because of the uncertainty associated with the , the optimal choice is determined based on maximising the *expected* .

Which for the set of options, , becomes:

We quantify the Expected Value of Perfect Information (EVPI) as the value of reducing uncertainty in decision-making. The EVPI represents the maximum amount a decision-maker would be willing to pay for perfect information that eliminates all uncertainty associated with their decision, reducing the SE to zero. The EVPI is given by:

In the current example we have one choice where the and the posterior distribution, and its moments, is given by:

To estimate the EVPI, we assume a second choice where the . Hence the EVPI becomes:

## Methods

### Inputs

We use a simple decision model with two inputs, and .

# Define model inputs:  
m1 <- 0  
m2 <- 190  
mu <- c(m1, m2)  
  
n\_observed <- 100  
L <- matrix(rep(c(2, 0), n\_observed), nrow=n\_observed, byrow=TRUE) # Transformation matrix L for 10 observations  
s\_N\_squared <- 2 # Variance of the measurement error, assuming identical for all observations  
y <- rnorm(n\_observed, 0, sqrt(s\_N\_squared)) # Example: 10 observed values generated around 2 with noise  
  
v1 <- 1000  
v2 <- 100  
rho <- 0  
Sigma <- matrix(c(v1, rho \* v1 \* v2, rho \* v1 \* v2, v2), 2, 2)  
  
# Calculate the inverse of Sigma  
Sigma\_inv <- solve(Sigma)  
  
set.seed(1)  
n <- 1e6  
x <- MASS::mvrnorm(n, mu, Sigma)  
x1 <- x[, 1]  
x2 <- x[, 2]

### Getting the hyperparameters

# Calculation of M^-1 and a requires adjustment for multiple observations  
# Adjust L and y to match the dimensions  
M\_inv <- Sigma\_inv + t(L) %\*% solve(diag(rep(s\_N\_squared, n\_observed))) %\*% L  
M <- solve(M\_inv)  
  
a <- M %\*% (Sigma\_inv %\*% mu + t(L) %\*% solve(diag(rep(s\_N\_squared, n\_observed))) %\*% y)  
  
# Print the results  
print("Hyperparameter a:")

[1] "Hyperparameter a:"

print(a)

[,1]  
[1,] 9.812347e-03  
[2,] 1.900000e+02

print("Covariance matrix M:")

[1] "Covariance matrix M:"

print(M)

[,1] [,2]  
[1,] 0.004999975 0  
[2,] 0.000000000 100

# Sample from the posterior  
hyper\_params\_samples <- MASS::mvrnorm(n, a, M)[,1]  
  
# Get the EVPI:  
hyper\_params\_evpi <- mean(pmax(hyper\_params\_samples, 0)) - pmax(mean(hyper\_params\_samples), 0)  
  
print("Hyper parameters EVPI:")

[1] "Hyper parameters EVPI:"

print(hyper\_params\_evpi)

[1] 0.02358901

### Simulated posterior parameters

The model, shown in the code below, estimates mortality (used in the calibration process) and net benefit (to inform our decision).

# Define the model:  
model.function <- function(.x1, .x2) {  
 .model.mortality <- .x1 + .x2  
 .inb <- .x1  
 return(list(model.mortality = .model.mortality, inb = .inb))  
}

We assume that the observed data follows a normal distribution.

# Define a likelihood  
likelihood <- function(.parameter, .target.mean, .target.se) {  
 dnorm(.target.mean, .parameter, .target.se)  
}  
  
y\_mean <- mean(y)  
y\_sd <- sd(y)  
  
  
# Evaluate the model using prior samples  
prior.output <- model.function(x1, x2)  
prior.inb <- prior.output$inb  
prior.evpi <- mean(pmax(prior.inb, 0)) - max(mean(prior.inb), 0)  
prior.evpi

[1] 12.61864

# Estimate the likelihood of prior samples  
likelihood.values <- likelihood(prior.output$model.mortality,  
 .target.mean = y\_mean,  
 .target.se = y\_sd / sqrt(n\_observed))  
  
# Re-sampling x1 and x2 with weights equal to the likelihood values  
  
# posterior.index <- sample(1:n, n, replace = TRUE, prob = likelihood.values)  
# posterior.x <- x[posterior.index, ]  
#   
# posterior.output <- model.function(posterior.x[, 1], posterior.x[, 2])  
#   
# posterior.inb <- posterior.output$inb  
#   
# mean(posterior.inb)  
# sd(posterior.inb)  
# posterior.evpi <- mean(pmax(posterior.inb, 0)) - max(mean(posterior.inb), 0)  
# posterior.evpi