Supplementary Data 2
In silico decision point qualitative method simulation

All scripts are available for download at `https://www.researchgate.net/profile/Brigitte_Desharnais`.

1 Standard model

```r
# QUAL_Simulation_Standard.R
# In silico decision point qualitative method simulation
# By Brigitte Desharnais, last modificaiton 2019-01-23

#########################################################
########## Parameters to be set by the user ##########
#########################################################

# Establishing the concentration at cut-off
CO_Conc <- 20

# Concentrations to be simulated
Conc <- c(0, 2, 4, 6, 8, 10, 11, 12, 14, 16, 17, 18, 20, 22, 23, 24, 26, 28, 29, 30,
    32, 34, 36, 38, 40)

# Establish the range of possible measurements at cut-off
# To be set, for example, based on preliminary experimental values collected.
Min_Meas <- 0.008
Max_Meas <- 1.050

# Number of total simulations to be performed
# Suggested: >50. Purely guides the number of simulations, can increase this number
    a lot.
```

```
Nb_Sim <- 100

# Number of measurements (i.e. virtual "spiked samples") at each concentration
# Value will depend on what you are trying to model: an experimental setup? (Use
    experimental value.)
# The underlying true value? (Use at least > 50.)
Nb_Meas <- 100

# %RSD at cut-off (will be used to calculate the homoscedastic error)
RSD <- 0.15

#####################################################
#####################################################
#####################################################

# Load necessary packages
library(dplyr)
library(ggplot2)
library(extrafont)

# Initializing an empty data frame to receive the results
Data <- data.frame(Conc = double(), Rate = double(), Iter = integer())

# Perform the specified number of simulations
for(i in 1:Nb_Sim){
  # Setting a known, true measurement at cut-off
  TR_CO <- runif(1, min = Min_Meas, max = Max_Meas)

  # Calculating the B1 value in y = B1*x
  B1 <- TR_CO/CO_Conc

  # Calculating the standard deviation value (homoscedastic data is simulated here)
  SD <- RSD*TR_CO

  # Initializing a vector for the positivity rates to be calculated
  Rate <- rep(0, times = length(Conc))
```

```r

  # For each concentration, calculate the positivity rate
  for(j in 1:length(Conc)){
    # Generate a measurements vector under a normal model
    Meas <- rnorm(Nb_Meas, mean = B1*Conc[j], sd = SD)

    # Calculate the positivity rate and store in the Rate vector
    # Number of measurements which are above the known, true cut-off value
    Rate[j] <- sum(Meas > TR_CO)/Nb_Meas*100
  }

  # Create an iteration vector
  Iter <- rep(i, times = length(Conc))

  # Bind together Conc, Rate and Iter column and append to Data data.frame.
  Temp <- cbind(Conc, Rate, Iter)
  Data <- rbind(Data, Temp)
}

# Convert data.frame to tbl.
Data <- tbl_df(Data)

# Generate a positivity graph (Figure 1(b))
ggplot(Data, aes(x = Conc, y = Rate)) +
  geom_smooth(size = 2, col = "#1824cc") +
  coord_cartesian(xlim = c(10, 30), ylim = c(0, 100)) +
  scale_x_continuous(name = "Concentration (ng/mL)") +
  scale_y_continuous(name = "Positivity Rate (%)") +
  theme(axis.title = element_text(size = 26, family = "Century Gothic"),
        axis.title.y = element_text(margin = margin(t = 0, r = 20, b = 0, l = 0)),
        axis.title.x = element_text(margin = margin(t = 20, r = 0, b = 0, l = 0)),
        axis.text = element_text(size = 22, family = "Century Gothic")) +
  geom_vline(xintercept = 20, col = "black", linetype = "longdash", size = 1)
```

2 Model corrected for sampled cut-off and heteroscedasticity

```r
# QUAL_Simulation_Corrected.R
# In silico decision point qualitative method simulation
# Modified simulation: sampled cut-off, heteroscedasticity
# By Brigitte Desharnais, last modificaiton 2019-01-23

##########################################################
########## Parameters to be set by the user ##########
##########################################################
# Establishing the concentration at cut-off
CO_Conc <- 20

# Concentrations to be simulated
Conc <- c(0, 2, 4, 6, 8, 10, 11, 12, 14, 16, 17, 18, 20, 22, 23, 24, 26, 28, 29, 30,
    32, 34, 36, 38, 40)

# Establish the range of possible measurements at cut-off
# To be set, for example, based on preliminary experimental values collected.
Min_Meas <- 0.008
Max_Meas <- 1.050

# Number of total simulations to be performed
# Suggested: >50. Purely guides the number of simulations, can increase this number
    a lot.
Nb_Sim <- 100

# Number of measurements (i.e. virtual "spiked samples") at each concentration
# Value will depend on what you are trying to model: an experimental setup? (Use
    experimental value.)
# The underlying true value? (Use at least > 50.)
Nb_Meas <- 30

# Number of cut-off measurements to be performed with each batch (sampled cut-off)
# Can set it up to reflect actual experimental work, or to explore the impact of
    changing its value.
Nb_CO <- 2
```

```r

# %RSD at all levels (heteroscedastic data)
# Set based on experimental data or hypothesis (e.g. SWGTOX dictates %RSD below 20%)
    .
RSD <- 0.15

##########################################################
##########################################################
##########################################################

# Load necessary packages
library(dplyr)
library(ggplot2)
library(extrafont)

# Initializing an empty data frame to receive the results
Data <- data.frame(Conc = double(), Rate = double(), Iter = integer())

# Perform the specified number of simulations
for(i in 1:Nb_Sim){
  # Setting a known, true measurement at cut-off
  TR_TR_CO <- runif(1, min = Min_Meas, max = Max_Meas)
  TR_CO <- mean(rnorm(Nb_CO, mean = TR_TR_CO, sd = RSD*TR_TR_CO))

  # Calculating the B1 value in y = B1*x
  B1 <- TR_CO/CO_Conc

  # Initializing a vector for the positivity rates to be calculated
  Rate <- rep(0, times = length(Conc))

  # For each concentration, calculate the positivity rate
  for(j in 1:length(Conc)){
    # Generate a measurements vector under a normal model
    Meas <- rnorm(Nb_Meas, mean = B1*Conc[j], sd = RSD*B1*Conc[j])

    # Calculate the positivity rate and store in the Rate vector
    # Number of measurements which are above the known, true cut-off value
    Rate[j] <- sum(Meas > TR_CO)/Nb_Meas*100
  }

  # Create an iteration vector
  Iter <- rep(i, times = length(Conc))

  # Bind together Conc, Rate and Iter column and append to Data data.frame.
  Temp <- cbind(Conc, Rate, Iter)
  Data <- rbind(Data, Temp)
}

# Convert data.frame to tbl.
Data <- tbl_df(Data)

# Generate a positivity graph
ggplot(Data, aes(x = Conc, y = Rate)) +
  geom_smooth(size = 2, col = "#1824cc") +
  coord_cartesian(xlim = c(10, 30), ylim = c(0, 100)) +
  scale_x_continuous(name = "Concentration (ng/mL)") +
  scale_y_continuous(name = "Positivity Rate (%)") +
  theme(axis.title = element_text(size = 26, family = "Century Gothic"),
        axis.title.y = element_text(margin = margin(t = 0, r = 20, b = 0, l = 0)),
        axis.title.x = element_text(margin = margin(t = 20, r = 0, b = 0, l = 0)),
        axis.text = element_text(size = 22, family = "Century Gothic")) +
  geom_vline(xintercept = 20, col = "black", linetype = "longdash", size = 1)

Summary <- Data %>% group_by(Conc) %>%
  summarise(min = min(Rate), max = max(Rate), mean = mean(Rate),
            Q5 = quantile(Rate, prob = 0.05), Q95 = quantile(Rate, prob = 0.95))
```

3 Corrected model and expected performance evaluation

Use this script to set validation criteria for your own validation (number of samples analyzed to evaluate FNR, FPR, RLR, SLR and SNR) and production (number of cut-off samples analyzed to estimate signal at threshold) conditions. Remember that for samples actually negative, only a TN or FP result is possible; for samples actually positive, only a TP or FN result is possible. The Summary Tables generated should be consulted to obtain validation criteria:

- FPR: $\leq$ 95% quantile at LURL;
- FNR: $\leq$ 95% quantile at UURL;
- RLR: $\geq$ 5% quantile at UURL;
- SLR: $\geq$ 5% quantile at LURL;
- SNR: $\geq$ 5% quantile at UURL;

Note that the RLR is calculated both from LURL and UURL data (indeed, $RLR = 100 - FPR - FNR$). The 5% quantile at UURL is taken as the validation criteria because in an heteroscedastic system (like mass spectrometry systems), this value will always be lower than at the LURL.

Quantiles (5%, 95%) are used as validation criteria instead of the minimum or maximum value obtained, which would also be a valid, albeit less conservative, choice. What is the rationale behind this choice? It parallels hypothesis testing and confidence intervals, in that if, e.g., an FNR of a value $\leq$ 95% quantile at UURL is obtained, the probability of obtaining this value or a more extreme one if the method indeed behaves as it should be is less than 5%, i.e., very unlikely. Other, stricter validation criteria (e.g., the mean rate obtained from the simulations) could be used, but then the risk of unecessarily rejecting perfectly adequate methods would be much higher.

```
# QUAL_Simulation_Performance.R
# In silico decision point qualitative method simulation
# Modified simulation & calculation of all performance statistics
# By Brigitte Desharnais, last modificaiton 2019-01-23

#####################################################
########## Parameters to be set by the user ##########
#####################################################
# Establishing the concentration at cut-off
CO_Conc <- 20

# Concentrations to be simulated
# You can input a series of concentrations to simulate positivity curves.
# Or you can input only the LURL, CO and UURL if you want to set validation criteria
    .
Conc <- c(11, 20, 29)

# Establish the range of possible measurements at cut-off
Min_Meas <- 0.008
Max_Meas <- 1.050

# Number of total simulations to be performed
# Suggested: >50. Purely guides the number of simulations, can increase this number
    a lot.
Nb_Sim <- 100

# Number of measurements (i.e. virtual "spiked samples") at each concentration
# Value will depend on what you are trying to model: an experimental setup? (Use
    experimental value.)
# The underlying true value? (Use at least > 50.)
Nb_Meas <- 30

# Number of cut-off measurements to be performed with each batch (sampled cut-off)
# Can set it up to reflect actual experimental work, or to explore the impact of
    changing its value.
Nb_CO <- 2

# %RSD at all levels (heteroscedastic data)
# Set based on experimental data or hypothesis (e.g. SWGTOX dictates %RSD below 20%)
```

```
    .
RSD <- 0.15

########################################################
########################################################
########################################################

# Load necessary packages
library(dplyr)
library(ggplot2)
library(extrafont)

# Initializing an empty data frame to receive the results
Data <- data.frame(Conc = double(), Nb_Pos = double(), Iter = integer())

# Perform the specified number of simulations
for(i in 1:Nb_Sim){
  # Setting a known, true measurement at cut-off
  TR_TR_CO <- runif(1, min = Min_Meas, max = Max_Meas)
  TR_CO <- mean(rnorm(Nb_CO, mean = TR_TR_CO, sd = RSD*TR_TR_CO))

  # Calculating the B1 value in y = B1*x
  B1 <- TR_CO/CO_Conc

  # Initializing a vector for the positivity rates to be calculated
  Nb_Pos <- rep(0, times = length(Conc))

  # For each concentration, calculate the positivity rate
  for(j in 1:length(Conc)){
    # Generate a measurements vector under a normal model
    Meas <- rnorm(Nb_Meas, mean = B1*Conc[j], sd = RSD*B1*Conc[j])

    # Calculate the positivity rate and store in the Rate vector
    # Number of measurements which are above the known, true cut-off value
    Nb_Pos[j] <- sum(Meas > TR_CO)
  }
```

```r

  # Create an iteration vector
  Iter <- rep(i, times = length(Conc))

  # Bind together Conc, Rate and Iter column and append to Data data.frame.
  Temp <- cbind(Conc, Nb_Pos, Iter)
  Data <- rbind(Data, Temp)
}

# Convert data.frame to tbl.
Data <- tbl_df(Data)

# Remove all observations at the cut-off concentration
#(which can't be classified into "true" or "false" positive/negative)
Data <- Data %>% filter(Conc != CO_Conc)

# Initialize empty columns for true/false positives/negatives
Data$TP <- 0
Data$FP <- 0
Data$TN <- 0
Data$FN <- 0

# Calculate the number of TP, TN, FP, FN for each concentration
for(k in 1:length(Data$Nb_Pos)){
  if(Data$Conc[k] < CO_Conc){
    # For samples actually negative, only a TN or FP result is possible.
    Data$FP[k] <- Data$Nb_Pos[k]
    Data$TN[k] <- Nb_Meas - Data$Nb_Pos[k]
  }else{
    # For samples actually positive, only a TP or FN result is possible.
    Data$TP[k] <- Data$Nb_Pos[k]
    Data$FN[k] <- Nb_Meas - Data$Nb_Pos[k]
  }
}

# Calculation of the performance statistics
```

```r
Data$FNR <- Data$FN/(Data$FN + Data$TP)*100
Data$FPR <- Data$FP/(Data$FP + Data$TN)*100
Data$REL <- (Data$TP + Data$TN) / (Data$TP + Data$TN + Data$FP + Data$FN)
Data$SEL <- Data$TN / (Data$TN + Data$FP)*100
Data$SEN <- Data$TP / (Data$TP +Data$FN)*100

# Summary table of the performance statistic per concentration level.
# Table lists the minimum and maximum value observed, the mean rate for all
    simulations
# performed, and the 5% and 95% percentile values observed.
# These summary tables were used to set the threshold values stated in the paper.
Summary_FNR <- Data %>% group_by(Conc) %>% arrange(FNR) %>%
  summarise(min = min(FNR), Q5 = nth(FNR, 5), mean = mean(FNR),
            Q95 = nth(FNR, 95), max = max(FNR))

Summary_FPR <- Data %>% group_by(Conc) %>% arrange(FPR) %>%
  summarise(min = min(FPR), Q5 = nth(FPR, 5), mean = mean(FPR),
            Q95 = nth(FPR, 95), max = max(FPR))

Summary_REL <- Data %>% group_by(Conc) %>% arrange(REL) %>%
  summarise(min = min(REL), Q5 = nth(REL, 5), mean = mean(REL),
            Q95 = nth(REL, 95), max = max(REL))

Summary_SEL <- Data %>% group_by(Conc) %>% arrange(SEL) %>%
  summarise(min = min(SEL), Q5 = nth(SEL, 5), mean = mean(SEL),
            Q95 = nth(SEL, 95), max = max(SEL))

Summary_SEN <- Data %>% group_by(Conc) %>% arrange(SEN) %>%
  summarise(min = min(SEN), Q5 = nth(SEN, 5), mean = mean(SEN),
            Q95 = nth(SEN, 95), max = max(SEN))
```