

## Supplementary Data 2

### *In silico* decision point qualitative method simulation

All scripts are available for download at [https://www.researchgate.net/profile/Brigitte\\_Desharnais](https://www.researchgate.net/profile/Brigitte_Desharnais).

#### 1 Standard model

```
1 # QUAL_Simulation_Standard.R
2 # In silico decision point qualitative method simulation
3 # By Brigitte Desharnais, last modification 2019-01-23
4
5 #####
6 ##### Parameters to be set by the user #####
7 #####
8
9 # Establishing the concentration at cut-off
10 CO_Conc <- 20
11
12 # Concentrations to be simulated
13 Conc <- c(0, 2, 4, 6, 8, 10, 11, 12, 14, 16, 17, 18, 20, 22, 23, 24, 26, 28, 29, 30,
14           32, 34, 36, 38, 40)
15
16 # Establish the range of possible measurements at cut-off
17 # To be set, for example, based on preliminary experimental values collected.
18 Min_Meas <- 0.008
19 Max_Meas <- 1.050
20
21 # Number of total simulations to be performed
22 # Suggested: >50. Purely guides the number of simulations, can increase this number
23 # a lot.
```

```

22 Nb_Sim <- 100
23
24 # Number of measurements (i.e. virtual "spiked samples") at each concentration
25 # Value will depend on what you are trying to model: an experimental setup? (Use
    experimental value.)
26 # The underlying true value? (Use at least > 50.)
27 Nb_Meas <- 100
28
29 # %RSD at cut-off (will be used to calculate the homoscedastic error)
30 RSD <- 0.15
31
32 #####
33 #####
34 #####
35
36 # Load necessary packages
37 library(dplyr)
38 library(ggplot2)
39 library(extrafont)
40
41 # Initializing an empty data frame to receive the results
42 Data <- data.frame(Conc = double(), Rate = double(), Iter = integer())
43
44 # Perform the specified number of simulations
45 for(i in 1:Nb_Sim){
46   # Setting a known, true measurement at cut-off
47   TR_CO <- runif(1, min = Min_Meas, max = Max_Meas)
48
49   # Calculating the B1 value in y = B1*x
50   B1 <- TR_CO/CO_Conc
51
52   # Calculating the standard deviation value (homoscedastic data is simulated here)
53   SD <- RSD*TR_CO
54
55   # Initializing a vector for the positivity rates to be calculated
56   Rate <- rep(0, times = length(Conc))

```

```

57
58 # For each concentration, calculate the positivity rate
59 for(j in 1:length(Conc)){
60   # Generate a measurements vector under a normal model
61   Meas <- rnorm(Nb_Meas, mean = B1*Conc[j], sd = SD)
62
63   # Calculate the positivity rate and store in the Rate vector
64   # Number of measurements which are above the known, true cut-off value
65   Rate[j] <- sum(Meas > TR_CO)/Nb_Meas*100
66 }
67
68 # Create an iteration vector
69 Iter <- rep(i, times = length(Conc))
70
71 # Bind together Conc, Rate and Iter column and append to Data data.frame.
72 Temp <- cbind(Conc, Rate, Iter)
73 Data <- rbind(Data, Temp)
74 }
75
76 # Convert data.frame to tbl.
77 Data <- tbl_df(Data)
78
79 # Generate a positivity graph (Figure 1(b))
80 ggplot(Data, aes(x = Conc, y = Rate)) +
81   geom_smooth(size = 2, col = "#1824cc") +
82   coord_cartesian(xlim = c(10, 30), ylim = c(0, 100)) +
83   scale_x_continuous(name = "Concentration (ng/mL)") +
84   scale_y_continuous(name = "Positivity Rate (%)") +
85   theme(axis.title = element_text(size = 26, family = "Century Gothic"),
86         axis.title.y = element_text(margin = margin(t = 0, r = 20, b = 0, l = 0)),
87         axis.title.x = element_text(margin = margin(t = 20, r = 0, b = 0, l = 0)),
88         axis.text = element_text(size = 22, family = "Century Gothic")) +
89   geom_vline(xintercept = 20, col = "black", linetype = "longdash", size = 1)

```

## 2 Model corrected for sampled cut-off and heteroscedasticity

```
1 # QUAL_Simulation_Corrected.R
2 # In silico decision point qualitative method simulation
3 # Modified simulation: sampled cut-off, heteroscedasticity
4 # By Brigitte Desharnais, last modificaiton 2019-01-23
5
6 #####
7 ##### Parameters to be set by the user #####
8 #####
9 # Establishing the concentration at cut-off
10 CO_Conc <- 20
11
12 # Concentrations to be simulated
13 Conc <- c(0, 2, 4, 6, 8, 10, 11, 12, 14, 16, 17, 18, 20, 22, 23, 24, 26, 28, 29, 30,
14           32, 34, 36, 38, 40)
15
16 # Establish the range of possible measurements at cut-off
17 # To be set, for example, based on preliminary experimental values collected.
18 Min_Meas <- 0.008
19 Max_Meas <- 1.050
20
21 # Number of total simulations to be performed
22 # Suggested: >50. Purely guides the number of simulations, can increase this number
23 # a lot.
24 Nb_Sim <- 100
25
26 # Number of measurements (i.e. virtual "spiked samples") at each concentration
27 # Value will depend on what you are trying to model: an experimental setup? (Use
28 # experimental value.)
29 # The underlying true value? (Use at least > 50.)
30 Nb_Meas <- 30
31
32 # Number of cut-off measurements to be performed with each batch (sampled cut-off)
33 # Can set it up to reflect actual experimental work, or to explore the impact of
34 # changing its value.
35 Nb_CO <- 2
```

```

32
33 # %RSD at all levels (heteroscedastic data)
34 # Set based on experimental data or hypothesis (e.g. SWGTOX dictates %RSD below 20%)
35 RSD <- 0.15
36
37 #####
38 #####
39 #####
40
41 # Load necessary packages
42 library(dplyr)
43 library(ggplot2)
44 library(extrafont)
45
46 # Initializing an empty data frame to receive the results
47 Data <- data.frame(Conc = double(), Rate = double(), Iter = integer())
48
49 # Perform the specified number of simulations
50 for(i in 1:Nb_Sim){
51   # Setting a known, true measurement at cut-off
52   TR_TR_CO <- runif(1, min = Min_Meas, max = Max_Meas)
53   TR_CO <- mean(rnorm(Nb_CO, mean = TR_TR_CO, sd = RSD*TR_TR_CO))
54
55   # Calculating the B1 value in  $y = B1*x$ 
56   B1 <- TR_CO/CO_Conc
57
58   # Initializing a vector for the positivity rates to be calculated
59   Rate <- rep(0, times = length(Conc))
60
61   # For each concentration, calculate the positivity rate
62   for(j in 1:length(Conc)){
63     # Generate a measurements vector under a normal model
64     Meas <- rnorm(Nb_Meas, mean = B1*Conc[j], sd = RSD*B1*Conc[j])
65
66     # Calculate the positivity rate and store in the Rate vector

```

```

67   # Number of measurements which are above the known, true cut-off value
68   Rate[j] <- sum(Meas > TR_CO)/Nb_Meas*100
69 }
70
71 # Create an iteration vector
72 Iter <- rep(i, times = length(Conc))
73
74 # Bind together Conc, Rate and Iter column and append to Data data.frame.
75 Temp <- cbind(Conc, Rate, Iter)
76 Data <- rbind(Data, Temp)
77 }
78
79 # Convert data.frame to tbl.
80 Data <- tbl_df(Data)
81
82 # Generate a positivity graph
83 ggplot(Data, aes(x = Conc, y = Rate)) +
84   geom_smooth(size = 2, col = "#1824cc") +
85   coord_cartesian(xlim = c(10, 30), ylim = c(0, 100)) +
86   scale_x_continuous(name = "Concentration (ng/mL)") +
87   scale_y_continuous(name = "Positivity Rate (%)") +
88   theme(axis.title = element_text(size = 26, family = "Century Gothic"),
89         axis.title.y = element_text(margin = margin(t = 0, r = 20, b = 0, l = 0)),
90         axis.title.x = element_text(margin = margin(t = 20, r = 0, b = 0, l = 0)),
91         axis.text = element_text(size = 22, family = "Century Gothic")) +
92   geom_vline(xintercept = 20, col = "black", linetype = "longdash", size = 1)
93
94 Summary <- Data %>% group_by(Conc) %>%
95   summarise(min = min(Rate), max = max(Rate), mean = mean(Rate),
96             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 unnecessarily rejecting perfectly adequate methods would be much higher.

```
1 # QUAL_Simulation_Performance.R
2 # In silico decision point qualitative method simulation
3 # Modified simulation & calculation of all performance statistics
```

```

4 # By Brigitte Desharnais, last modifcaiton 2019-01-23
5
6 #####
7 ##### Parameters to be set by the user #####
8 #####
9 # Establishing the concentration at cut-off
10 CO_Conc <- 20
11
12 # Concentrations to be simulated
13 # You can input a series of concentrations to simulate positivity curves.
14 # Or you can input only the LURL, CO and UURL if you want to set validation criteria
15 Conc <- c(11, 20, 29)
16
17 # Establish the range of possible measurements at cut-off
18 Min_Meas <- 0.008
19 Max_Meas <- 1.050
20
21 # Number of total simulations to be performed
22 # Suggested: >50. Purely guides the number of simulations, can increase this number
    a lot.
23 Nb_Sim <- 100
24
25 # Number of measurements (i.e. virtual "spiked samples") at each concentration
26 # Value will depend on what you are trying to model: an experimental setup? (Use
    experimental value.)
27 # The underlying true value? (Use at least > 50.)
28 Nb_Meas <- 30
29
30 # Number of cut-off measurements to be performed with each batch (sampled cut-off)
31 # Can set it up to reflect actual experimental work, or to explore the impact of
    changing its value.
32 Nb_CO <- 2
33
34 # %RSD at all levels (heteroscedastic data)
35 # Set based on experimental data or hypothesis (e.g. SWGTOX dictates %RSD below 20%)

```



```

36 RSD <- 0.15
37
38 #####
39 #####
40 #####
41
42 # Load necessary packages
43 library(dplyr)
44 library(ggplot2)
45 library(extrafont)
46
47 # Initializing an empty data frame to receive the results
48 Data <- data.frame(Conc = double(), Nb_Pos = double(), Iter = integer())
49
50 # Perform the specified number of simulations
51 for(i in 1:Nb_Sim){
52   # Setting a known, true measurement at cut-off
53   TR_TR_CO <- runif(1, min = Min_Meas, max = Max_Meas)
54   TR_CO <- mean(rnorm(Nb_CO, mean = TR_TR_CO, sd = RSD*TR_TR_CO))
55
56   # Calculating the B1 value in y = B1*x
57   B1 <- TR_CO/CO_Conc
58
59   # Initializing a vector for the positivity rates to be calculated
60   Nb_Pos <- rep(0, times = length(Conc))
61
62   # For each concentration, calculate the positivity rate
63   for(j in 1:length(Conc)){
64     # Generate a measurements vector under a normal model
65     Meas <- rnorm(Nb_Meas, mean = B1*Conc[j], sd = RSD*B1*Conc[j])
66
67     # Calculate the positivity rate and store in the Rate vector
68     # Number of measurements which are above the known, true cut-off value
69     Nb_Pos[j] <- sum(Meas > TR_CO)
70   }

```

```

71
72 # Create an iteration vector
73 Iter <- rep(i, times = length(Conc))
74
75 # Bind together Conc, Rate and Iter column and append to Data data.frame.
76 Temp <- cbind(Conc, Nb_Pos, Iter)
77 Data <- rbind(Data, Temp)
78 }
79
80 # Convert data.frame to tbl.
81 Data <- tbl_df(Data)
82
83 # Remove all observations at the cut-off concentration
84 #(which can't be classified into "true" or "false" positive/negative)
85 Data <- Data %>% filter(Conc != CO_Conc)
86
87 # Initialize empty columns for true/false positives/negatives
88 Data$TP <- 0
89 Data$FP <- 0
90 Data$TN <- 0
91 Data$FN <- 0
92
93 # Calculate the number of TP, TN, FP, FN for each concentration
94 for(k in 1:length(Data$Nb_Pos)){
95   if(Data$Conc[k] < CO_Conc){
96     # For samples actually negative, only a TN or FP result is possible.
97     Data$FP[k] <- Data$Nb_Pos[k]
98     Data$TN[k] <- Nb_Meas - Data$Nb_Pos[k]
99   } else {
100     # For samples actually positive, only a TP or FN result is possible.
101     Data$TP[k] <- Data$Nb_Pos[k]
102     Data$FN[k] <- Nb_Meas - Data$Nb_Pos[k]
103   }
104 }
105
106 # Calculation of the performance statistics

```

```

107 Data$FNR <- Data$FN/(Data$FN + Data$TP)*100
108 Data$FPR <- Data$FP/(Data$FP + Data$TN)*100
109 Data$REL <- (Data$TP + Data$TN) / (Data$TP + Data$TN + Data$FP + Data$FN)
110 Data$SEL <- Data$TN / (Data$TN + Data$FP)*100
111 Data$SEN <- Data$TP / (Data$TP +Data$FN)*100
112
113 # Summary table of the performance statistic per concentration level.
114 # Table lists the minimum and maximum value observed, the mean rate for all
      simulations
115 # performed, and the 5% and 95% percentile values observed.
116 # These summary tables were used to set the threshold values stated in the paper.
117 Summary_FNR <- Data %>% group_by(Conc) %>% arrange(FNR) %>%
118   summarise(min = min(FNR), Q5 = nth(FNR, 5), mean = mean(FNR),
119             Q95 = nth(FNR, 95), max = max(FNR))
120
121 Summary_FPR <- Data %>% group_by(Conc) %>% arrange(FPR) %>%
122   summarise(min = min(FPR), Q5 = nth(FPR, 5), mean = mean(FPR),
123             Q95 = nth(FPR, 95), max = max(FPR))
124
125 Summary_REL <- Data %>% group_by(Conc) %>% arrange(REL) %>%
126   summarise(min = min(REL), Q5 = nth(REL, 5), mean = mean(REL),
127             Q95 = nth(REL, 95), max = max(REL))
128
129 Summary_SEL <- Data %>% group_by(Conc) %>% arrange(SEL) %>%
130   summarise(min = min(SEL), Q5 = nth(SEL, 5), mean = mean(SEL),
131             Q95 = nth(SEL, 95), max = max(SEL))
132
133 Summary_SEN <- Data %>% group_by(Conc) %>% arrange(SEN) %>%
134   summarise(min = min(SEN), Q5 = nth(SEN, 5), mean = mean(SEN),
135             Q95 = nth(SEN, 95), max = max(SEN))

```