Supplementary Data 1

Simulations

# Introduction

To assess the performance and mathematical validity of both our *classic* and *adjusted model*, an in silico simulation of the experimental setups was designed, which we describe here. Using these simulations, many virtual digital PCR experiments can be carried out under specified conditions, allowing for an extensive evaluation of the theoretical accuracy and precision of the obtained results.

In order to reproduce the results of **Figure 2A** and **2B** of the manuscript, we first set a random seed.

# Set random seed  
set.seed(12345)

To run the simulations, we make use of our R library digitalPCRsimulations.

# Install library 'digitalPCRsimulations'  
library(devtools)

## Loading required package: usethis

#install\_github("rjnell/digitalPCRsimulations")  
  
# Load library 'digitalPCRsimulations'  
library(digitalPCRsimulations)

# Function definitions

We extend this library with an extra function to calculate the difference between two concentrations. The mathematical rationale is described in detail in the **Materials and Methods** of the manuscript. This function takes the given sample (selection of partitions to include) of the two universes of target1 (e.g. regional corrector RCΔB) and target2 (e.g. T-cell marker ΔB), and then calculates the concentration difference [target1] - [target2].

# Function to calculate the difference between two concentrations,  
# obtained by taking a given sample from two given universes.  
sample\_from\_2\_universes\_diff = function(universe1, universe2, sample, alpha=0.95) {  
   
 # What is the total length of the sample?  
 n = length(sample)  
   
 # As described in the manuscript:  
 # concentration difference target1 - target2 =  
 # [1] - [2] = -log(p\_1\_min / p\_2\_min) / volume  
 p\_1\_min = (n - sum(universe1[sample])) / n  
 p\_2\_min = (n - sum(universe2[sample])) / n  
   
 # Translate this percentage to concentration (difference)...  
 conc = -log(p\_1\_min / p\_2\_min)  
   
 # ... and its CI  
 # As described in manuscript:  
 # Var = (p\_1\_min^-1 + p\_2\_min^-1 - 2) / n  
 var = (p\_1\_min^(-1) + p\_2\_min^(-1) - 2)/n  
   
 # ... and s = sqrt(var), following Dube et al. 2008  
 s = sqrt(var)  
 z = qnorm(1-(1-alpha)/2)  
 conc\_low = conc - z \* s  
 conc\_high = conc + z \* s  
   
 # Volume per droplet  
 volume = 0.00085  
   
 # Return the calculated concentration (difference) with CI  
 conc\_ci = c(conc/volume, conc\_low/volume, conc\_high/volume)   
 names(conc\_ci) = c("concentration", "concentration\_low", "concentration\_high")  
 return(conc\_ci)  
}

The multiplex digital PCR setup described in the manuscript (**Figure 3B**) allows for the simultaneous measurement of the T-cell marker (ΔB), its regional corrector (RCΔB) and the stable reference (REF). Consequently, both the T-cell fraction according to the To simulate this setup, we create another function.

# Function to simulate multiplex adjusted model  
simulate\_multiplex\_adjusted\_model = function(input\_ng,   
 tcf,   
 cnv\_ref,   
 cnv\_region,   
 n\_droplets,   
 n\_simulations,   
 alpha) {  
   
 # The amount of input alleles of the reference depends on the chosen amount of input DNA,   
 # the T-cell fraction and the CNV of the reference.  
 # As in T cells (= healthy cells) a default CNV of 2 is found,   
 # the CNV of the reference can only be altered due to CNA in the non-T cells.  
 input\_ng\_ref = input\_ng \* (tcf \* 2 + (1-tcf) \* cnv\_ref) / 2  
 universe\_ref = universe(input\_ng\_ref)  
   
 # The amount of input alleles of the regional marker depends on the chosen amount of input DNA,   
 # the T-cell fraction and the regional CNV.  
 # In T cells (= healthy cells) a default CNV of 2 is found,   
 # while in non-T cells we observe the regional CNV.  
 input\_ng\_region = input\_ng \* (tcf \* 2 + (1-tcf) \* cnv\_region) / 2  
 universe\_region = universe(input\_ng\_region)  
   
 # The amount of input alleles of the T-cell marker depends on the chosen amount of input DNA,   
 # the T-cell fraction and the regional CNV.  
 # In T-cells the marker is completely lost,   
 # while in non-T cells the regional CNV determines the absolute presence of the T-cell marker.  
 input\_ng\_tcm = input\_ng \* (1-tcf) \* cnv\_region / 2  
 universe\_tcm = universe(input\_ng\_tcm)  
   
 # Initialize the list of results of the simulations  
 results = list()  
 results$tcf\_true\_value = tcf  
 results$tcf\_classic = matrix(NA, nrow = 0, ncol = 3)  
 results$tcf\_adjusted = matrix(NA, nrow = 0, ncol = 3)  
   
 for (simulation in 1:n\_simulations) {  
   
 # Simulate duplex and multiplex  
 sample = simulate\_sample(n\_droplets)  
   
 # Determine the concentration of the reference   
 conc\_ref = sample\_from\_universe(universe\_ref,   
 sample,   
 alpha)  
   
 # Determine the concentration of the regional marker  
 conc\_region = sample\_from\_universe(universe\_region,   
 sample,   
 alpha)  
   
 # Determine the concentration of the T-cell marker  
 conc\_tcm = sample\_from\_universe(universe\_tcm,   
 sample,   
 alpha)  
   
 # Determine the concentration of [regional marker - T-cell marker]  
 conc\_diff\_region\_tcm = sample\_from\_2\_universes\_diff(universe\_region,   
 universe\_tcm,   
 sample,   
 alpha)  
   
 # Calculate the T-cell fraction according the classic model,   
 # using the T-cell marker and the reference  
 tcf\_classic = 1-calc\_ratio(conc\_tcm, conc\_ref)[c(1,3,2)]  
   
 # Calculate the T-cell fraction according the adjusted model,   
 # using the T-cell marker, the regional marker as corrector,  
 # and reference 1.  
 tcf\_adjusted = calc\_ratio(conc\_diff\_region\_tcm, conc\_ref)   
   
 # Bind obtained result to the simulation results list  
 results$tcf\_classic = rbind(results$tcf\_classic, tcf\_classic)   
 results$tcf\_adjusted = rbind(results$tcf\_adjusted, tcf\_adjusted)  
   
 }  
   
 # Return obtained simulation\_results  
 return (results)  
}

# Shortcut function to plot summary of simulations

plot\_simulations\_summary = function(main, simulation\_results, true\_tcf) {  
   
 par(mar=c(5.1, 5, 4.1, 2.1))  
 ylim = c(0,60)  
 xlim = c(-1,22)  
 plot(x = xlim,  
 y = ylim,  
 pch = 16,  
 xlab = "",  
 ylab = "",   
 ylim = ylim,  
 xlim = xlim,   
 bty = "l",   
 type="n",  
 xaxs = "i",   
 yaxs = "i",  
 axes = F)  
   
 yat = seq(from=ylim[1],to=ylim[2],by=10)  
 xat = c(xlim[1],xlim[2])  
 segments(xlim[1],yat,xlim[2],col="#eeeeee",lwd=1.4,xpd=T)  
 segments(xlim[1],50,xlim[2],col="#b1b1b1",lwd=1.4,xpd=T,lty=3)  
 segments(xlim[1],ylim[1],xlim[1],ylim[2]+3.75,xpd=T,col="#B1B1B1",lwd=1.4)  
 axis(side = 2,at=yat,las=2,labels=rep("",length(yat)),col.ticks = "#b1b1b1",col = "#b1b1b1", tck = -0.035,lwd=1.4)  
 axis(side = 2,at=yat,las=2,labels=paste0(yat,"%"), lwd=0, col.axis="#333333",line=-0.23)  
 mtext(text = "T-cell fraction", side = 2, line=3.7,col="#333333")   
   
 r = simulation\_results  
 s = stats(r, true\_value=true\_tcf)  
   
 m = format(round(s$point\_estimate\_mean\*100, 1), nsmall = 1)  
 sd = format(round(s$point\_estimate\_sd\*100, 1), nsmall = 1)  
 cc= format(round(s$coverage\*100, 1), nsmall = 1)  
   
 axis(side = 1, at=xat, labels=rep("",length(xat)),col.ticks = "#b1b1b1",col = "#b1b1b1", tck = 0,lwd=1.4)  
 mtext(text = paste0("Mean (sd) = ", m, "% (", sd,")"), side = 1, line=1.2,col="#333333")   
 mtext(text = paste0("95%-CI coverage = ", cc, "%"), side = 1, line=2.2,col="#333333")   
 mtext(text = main, side = 3, line=1.2,col="#333333", font=4)   
   
 segments(xlim[1],ylim[1],xlim[2],ylim[1],xpd=T,col="#B1B1B1",lwd=1.4)  
   
 results = r[1:20,]\*100  
 results = results[order(results[,1]),]  
   
 lty=rep(1,20)  
 col=rep("#b1b1b1",20)  
 w=which(results[,2]>50 | results[,3] <50)  
 lty[w] = 1  
 col[w] = "#FBB4AF"  
   
 arrows(1:nrow(results), results[,2], 1:nrow(results), results[,3], length=0.05, angle=90, code=3, col=col,lty=lty,lwd=1.4)  
 points(x = 1:nrow(results),  
 y = results[,1],  
 pch = 15, cex=0.9, col="#333333")  
   
}

### SIMULATION 1: 20 ng, copy number stable, 50% TCF

simulation\_1 = simulate\_multiplex\_adjusted\_model(  
   
 # Input in ng genomic DNA  
 input\_ng = 20,   
   
 # T-cell fraction (0-1)  
 tcf = 0.50,   
   
 # Copy number value of the reference in non-tcf (def = 2)  
 cnv\_ref = 2,   
   
 # Copy number value of the T-cell marker region in non-tcf (def = 2)  
 cnv\_region = 2,   
   
 # Number of droplets per experiment  
 n\_droplets = 20000,   
   
 # Number of simulations  
 n\_simulations = 100,   
   
 # Level of significance  
 alpha = 0.95)  
  
# Plot summary of simulation results of classic model  
png("fig-2a-1.png", res=600, width=2000, height=2500)   
plot\_simulations\_summary("Classic model", simulation\_1$tcf\_classic, 0.5)  
dev.off()

## png   
## 2

# Plot summary of simulation results of adjusted model  
png("fig-2a-2.png", res=600, width=2000, height=2500)   
plot\_simulations\_summary("Adjusted model", simulation\_1$tcf\_adjusted, 0.5)  
dev.off()

## png   
## 2

### SIMULATION 2: 20 ng, copy number unstable, 50% TCF

simulation\_2 = simulate\_multiplex\_adjusted\_model(  
   
 # Input in ng genomic DNA  
 input\_ng = 20,   
   
 # T-cell fraction (0-1)  
 tcf = 0.5,   
   
 # Copy number value of the reference in non-tcf (def = 2)  
 cnv\_ref = 2,   
   
 # Copy number value of the T-cell marker region in non-tcf (def = 2)  
 cnv\_region = 3,   
   
 # Number of droplets per experiment  
 n\_droplets = 20000,   
   
 # Number of simulations  
 n\_simulations = 100,   
   
 # Level of significance  
 alpha = 0.95)  
  
# Plot summary of simulation results of classic model  
png("fig-2b-1.png", res=600, width=2000, height=2500)   
plot\_simulations\_summary("Classic model", simulation\_2$tcf\_classic, 0.5)  
dev.off()

## png   
## 2

# Plot summary of simulation results of adjusted model  
png("fig-2b-2.png", res=600, width=2000, height=2500)   
plot\_simulations\_summary("Adjusted model", simulation\_2$tcf\_adjusted, 0.5)  
dev.off()

## png   
## 2