Group level true positives using exGaussian distributions

Guillaume A. Rousselet

2019-01-16

Contents

```
# dependencies
library(ggplot2)
library(tibble)
library(tidyr)
library(cowplot)
library(retimes)
library(rogme)
source("./functions/tests.txt")
source("./functions/skew.txt")
library(beepr)
```

```
sessionInfo()
```

```
## R version 3.4.0 (2017-04-21)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS  10.14.2
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_GB.UTF-8/en_GB.UTF-8/en_GB.UTF-8/C/en_GB.UTF-8/en_GB.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] beepr_1.3     rogme_0.2.0   retimes_0.1-2 cowplot_0.9.1 tidyr_0.7.2
## [6] tibble_1.4.2  ggplot2_3.0.0
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.19     pillar_1.3.0     compiler_3.4.0   plyr_1.8.4
##  [5] bindr_0.1.1      tools_3.4.0      digest_0.6.12    evaluate_0.10.1
##  [9] gtable_0.2.0     pkgconfig_2.0.2  rlang_0.2.2      rstudioapi_0.8
## [13] yaml_2.1.15      bindrcpp_0.2.2   withr_2.1.0      dplyr_0.7.6
```

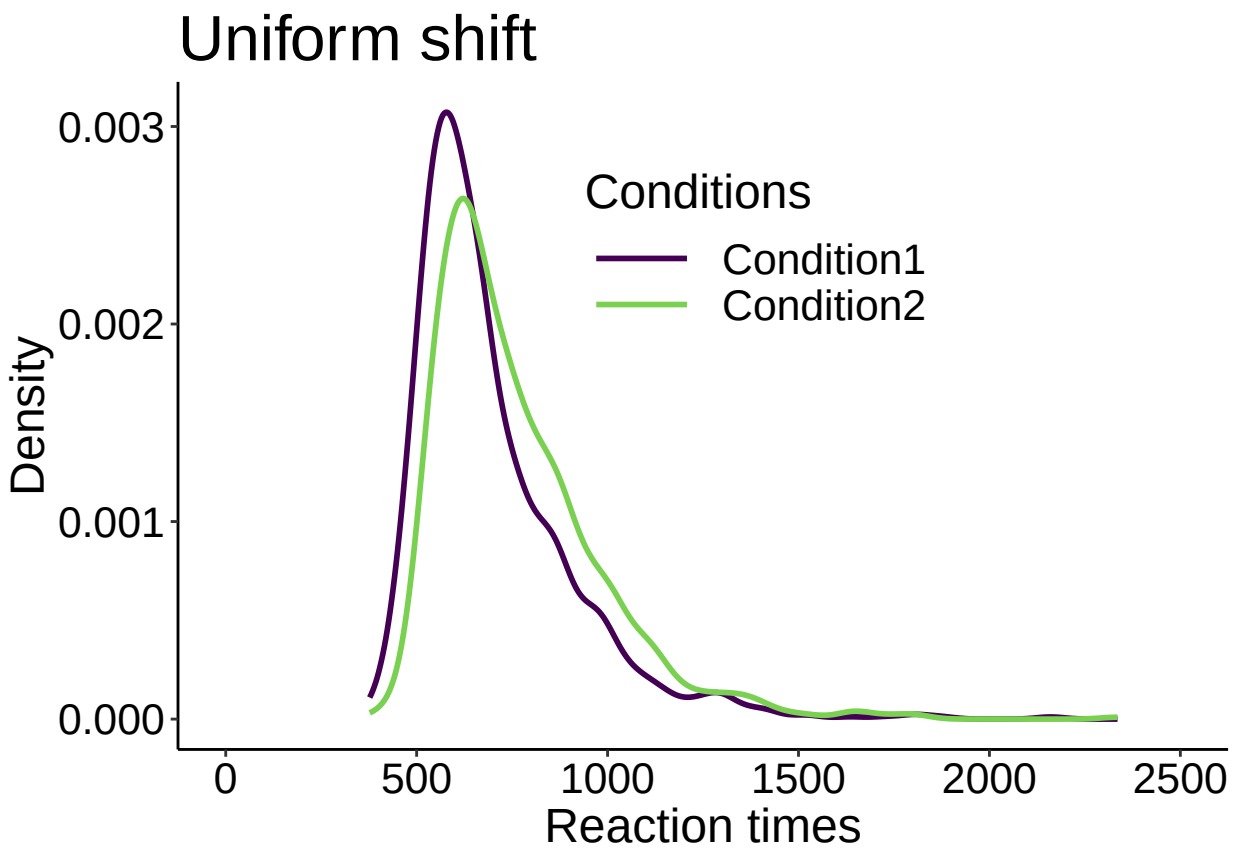
```
## [17] stringr_1.2.0    knitr_1.17       rprojroot_1.2    grid_3.4.0
## [21] tidyselect_0.2.4 glue_1.3.0       R6_2.3.0         rmarkdown_1.8
## [25] purrr_0.2.5      magrittr_1.5     backports_1.1.1  scales_0.5.0
## [29] htmltools_0.3.6  assertthat_0.2.0 colorspace_1.3-2 stringi_1.1.6
## [33] lazyeval_0.2.1   munsell_0.4.3    crayon_1.3.4     audio_0.1-5.1
```

Consider power for changes in exGaussian distributions:
- uniform shift;
- spread;
- tail only (tau component).

Group simulation 1: same number of trials in each group, uniform shift

KDE

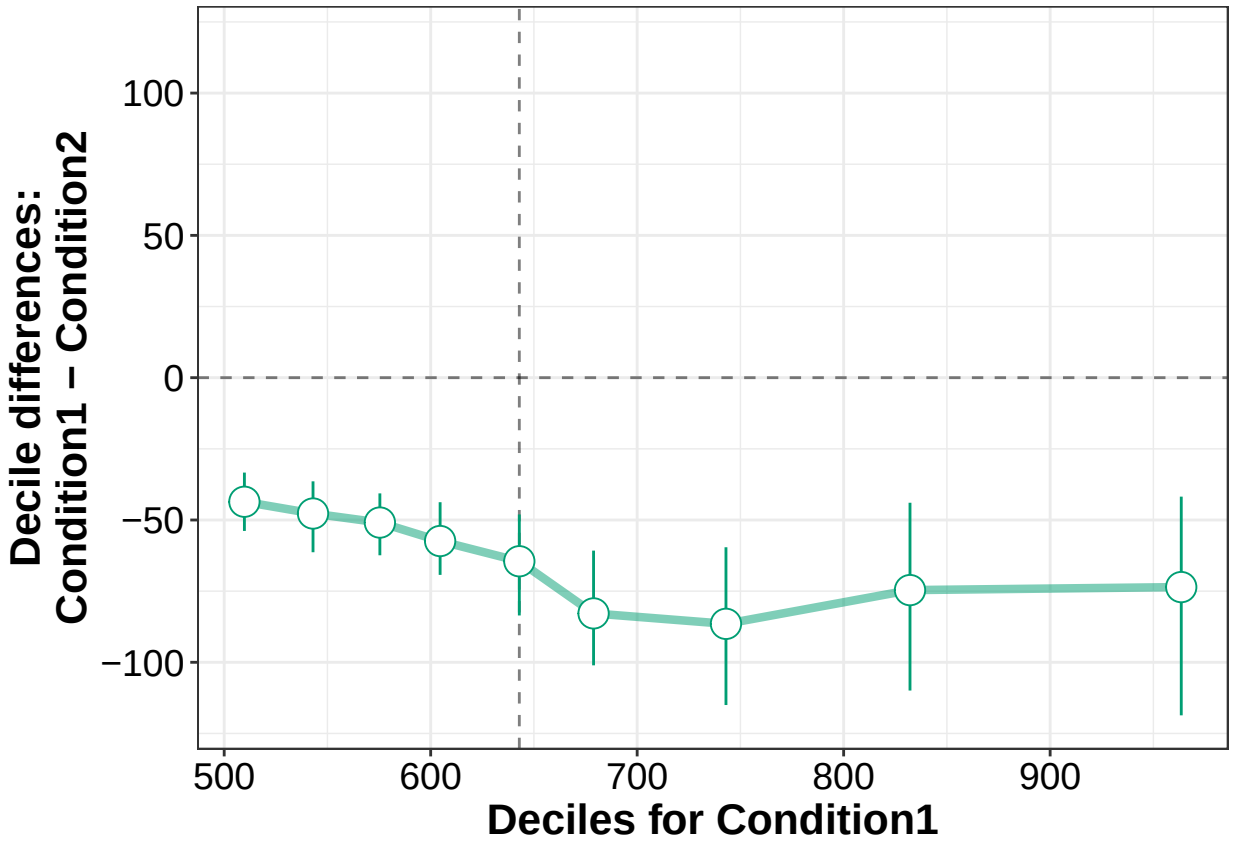
```
nt <- 1000
# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 50

set.seed(21)
g1 <- rexgauss(nt, mu = mu, sigma = sigma, tau = tau)
g2 <- rexgauss(nt, mu = mu, sigma = sigma, tau = tau) + ES
df <- mkt2(g1, g2, group_labels = c("Condition1", "Condition2"))

p <- ggplot(df, aes(x = obs)) + theme_classic() +
  stat_density(aes(colour = gr), geom="line",position="identity", size=1) +
  scale_colour_viridis_d(end = 0.8) +
  coord_cartesian(xlim = c(0, 2500)) +
  theme(axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 16, colour="black"),
        axis.text.y = element_text(size = 16, colour="black"),
        axis.title.y = element_text(size = 18),
        legend.key.width = unit(1.5,"cm"),
        legend.position = c(0.55,0.75),
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        title = element_text(size=20)) +
  labs(x = "Reaction times", y = "Density", colour = "Conditions") +
  ggtitle("Uniform shift")
p
```




```
p.uni.dist <- p
```

Shift function

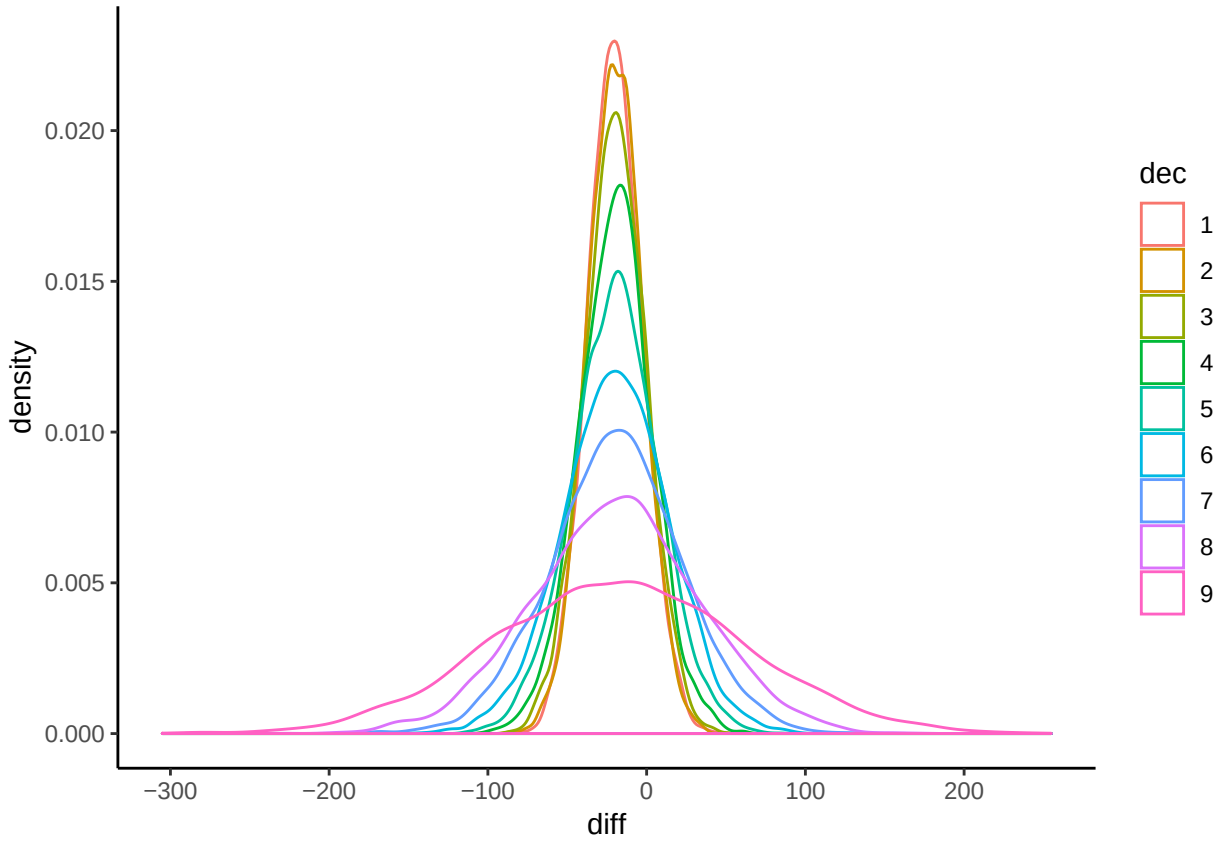
```
out <- shifthd_pbci(df, nboot = 200, adj_ci = FALSE)
p <- plot_sf(out, plot_theme = 1)[[1]] +
     theme(axis.text = element_text(size = 16, colour="black"))
p
```




```
p.uni.sf <- p
```

Decile sampling distributions

```
nt <- 100 # trials
np <- 5000 # participants
# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 20

mc.data1 <- array(rexgauss(nt*np, mu = mu, sigma = sigma, tau = tau),
  dim = c(nt, np))
mc.data2 <- array(rexgauss(nt*np, mu = mu, sigma = sigma, tau = tau),
  dim = c(nt, np)) + ES

# array of differences: 9 deciles x participants
mc.hd <- apply(mc.data1, 2, hdseq) - apply(mc.data2, 2, hdseq)

df <- tibble(diff = as.vector(mc.hd),
             dec = factor(rep(seq(1,9), np)))
ggplot(df, aes(x = diff, colour = dec)) + theme_classic() +
  geom_density()
```



Skewness of sampling distributions

```
apply(mc.hd, 1, skew)
```

```
## [1]  0.054669089 -0.010950193 -0.052160303 -0.037255368 -0.009998445
## [6] -0.048842786 -0.051164577 -0.077623716 -0.001513750
```

Kurtosis of sampling distributions

```
apply(mc.hd, 1, kurt)
```

```
## [1] 3.019325 3.086741 3.039861 3.191546 3.115746 3.154522 3.215416 3.167294
## [9] 3.003582
```

Simulation

```
npseq <- c(25, 50, 100)
ntseq <- c(seq(50, 100, 10), 150, 200)
maxnp <- max(npseq) # max number of participants
maxnt <- max(ntseq) # max number of trials
nsim <- 10000 # simulation samples
qseq <- seq(0.1,0.9,0.1) # define deciles for quantile function
```

```r
# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 10

alpha <- 0.05

# declare matrices of results - save all iterations
res.m.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.md.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.25.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.75.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfhd.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfqt8.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

pvals.0.hd <- vector(mode = "numeric", length = 9)
pvals.10.hd <- vector(mode = "numeric", length = 9)
pvals.20.hd <- vector(mode = "numeric", length = 9)

pvals.0.qt8 <- vector(mode = "numeric", length = 9)
pvals.10.qt8 <- vector(mode = "numeric", length = 9)
pvals.20.qt8 <- vector(mode = "numeric", length = 9)

# create 2 populations
set.seed(21)
npop <- 1000000
pop2 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau)
pop1 <- pop2 + ES

beep(3)

# simulation samples
mc.data1 <- array(sample(pop1, maxnt*maxnp*nsim, replace = TRUE), dim = c(maxnt, maxnp, nsim))
mc.data2 <- array(sample(pop2, maxnt*maxnp*nsim, replace = TRUE), dim = c(maxnt, maxnp, nsim))

for(TR in 1:length(ntseq)){ # number of trials
  print(paste0("number of trials: ",ntseq[TR],"..."))
  beep(2)

  for(PT in 1:length(npseq)){ # number of participants
    print(paste0("number of participants: ",npseq[PT],"..."))

    # compute estimates
    todo1 <- mc.data1[1:ntseq[TR], 1:npseq[PT],]
```

```
    todo2 <- mc.data2[1:ntseq[TR], 1:npseq[PT],]
    mc.m <- apply(todo1, c(2,3), mean) - apply(todo2, c(2,3), mean) # diff in mean RTs
    mc.md <- apply(todo1, c(2,3), median) - apply(todo2, c(2,3), median) # diff in median RTs
    mc.25 <- apply(todo1, c(2,3), quantile, probs = .25) - apply(todo2, c(2,3), quantile, probs = .25)
    mc.75 <- apply(todo1, c(2,3), quantile, probs = .75) - apply(todo2, c(2,3), quantile, probs = .75)

    # tests for each simulation ====================
    # one-sample t-test
    res.m.m.sig[, TR, PT] <- ttest.sig(mc.m)
    # parametric median test
    res.md.md.sig[, TR, PT] <- apply(mc.md, 2, sint.sig)
    res.25.md.sig[, TR, PT] <- apply(mc.25, 2, sint.sig)
    res.75.md.sig[, TR, PT] <- apply(mc.75, 2, sint.sig)

    # shift function
    # array of differences: 9 deciles x participants x simulations
    mc.hd <- apply(todo1, c(2,3), hdseq) - apply(todo2, c(2,3), hdseq)
    mc.qt8 <- apply(todo1, c(2,3), quantile, probs = qseq, type = 8) -
              apply(todo2, c(2,3), quantile, probs = qseq, type = 8)
    # median test for each decile distribution
    pval.hd <- apply(mc.hd, c(1,3), sintv2.pval)
    pval.qt8 <- apply(mc.qt8, c(1,3), sintv2.pval)
    # trimmed mean tests for each decile distribution
    tvals.0.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0)
    tvals.10.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.1)
    tvals.20.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.2)
    tvals.0.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0)
    tvals.10.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.1)
    tvals.20.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.2)
    df0 <- df.yuen(mc.hd[1,,1], tr = 0)
    df10 <- df.yuen(mc.hd[1,,1], tr = 0.1)
    df20 <- df.yuen(mc.hd[1,,1], tr = 0.2)
    for(S in 1:nsim){
      res.sfhd.md.sig[S, TR, PT] <- sum(p.adjust(pval.hd[,S], method = "hochberg") <= alpha)
      res.sfqt8.md.sig[S, TR, PT] <- sum(p.adjust(pval.qt8[,S], method = "hochberg") <= alpha)

      for(Q in 1:9){
        pvals.0.hd[Q] <- cpval(tvals.0.hd[S,Q], df0)
        pvals.10.hd[Q] <- cpval(tvals.10.hd[S,Q], df10)
        pvals.20.hd[Q] <- cpval(tvals.20.hd[S,Q], df20)
        pvals.0.qt8[Q] <- cpval(tvals.0.qt8[S,Q], df0)
        pvals.10.qt8[Q] <- cpval(tvals.10.qt8[S,Q], df10)
        pvals.20.qt8[Q] <- cpval(tvals.20.qt8[S,Q], df20)
      }
      res.sfhd.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.hd, method = "hochberg") <= alpha)
      res.sfhd.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.hd, method = "hochberg") <= alpha)
      res.sfhd.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.hd, method = "hochberg") <= alpha)
      res.sfqt8.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.qt8, method = "hochberg") <= alpha)
      res.sfqt8.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.qt8, method = "hochberg") <= alpha)
      res.sfqt8.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.qt8, method = "hochberg") <= alpha)
    }
  }
}
```

```
save(
  res.m.m.sig,
  res.md.md.sig,
  res.25.md.sig,
  res.75.md.sig,
  res.sfhd.md.sig,
  res.sfhd.tm10.sig,
  res.sfhd.tm20.sig,
  res.sfhd.m.sig,
  res.sfqt8.md.sig,
  res.sfqt8.tm10.sig,
  res.sfqt8.tm20.sig,
  res.sfqt8.m.sig,
  ntseq,
  npseq,
  mu,
  ES,
  sigma,
  tau,
  nsim,
  file=paste0('./data/sim_gp_tp1.RData'))
beep(8)
```

Compute true positive probability

```
load('./data/sim_gp_tp1.RData')
fp.m.m <- apply(res.m.m.sig, c(2,3), mean)
fp.md.md <- apply(res.md.md.sig, c(2,3), mean)
fp.25.md <- apply(res.25.md.sig, c(2,3), mean)
fp.75.md <- apply(res.75.md.sig, c(2,3), mean)
fp.sfhd.md <- apply(res.sfhd.md.sig > 0, c(2,3), mean)
fp.sfhd.m <- apply(res.sfhd.m.sig > 0, c(2,3), mean)
fp.sfhd.tm10 <- apply(res.sfhd.tm10.sig > 0, c(2,3), mean)
fp.sfhd.tm20 <- apply(res.sfhd.tm20.sig > 0, c(2,3), mean)
fp.sfqt8.md <- apply(res.sfqt8.md.sig > 0, c(2,3), mean)
fp.sfqt8.m <- apply(res.sfqt8.m.sig > 0, c(2,3), mean)
fp.sfqt8.tm10 <- apply(res.sfqt8.tm10.sig > 0, c(2,3), mean)
fp.sfqt8.tm20 <- apply(res.sfqt8.tm20.sig > 0, c(2,3), mean)
```

Illustrate results

Make function

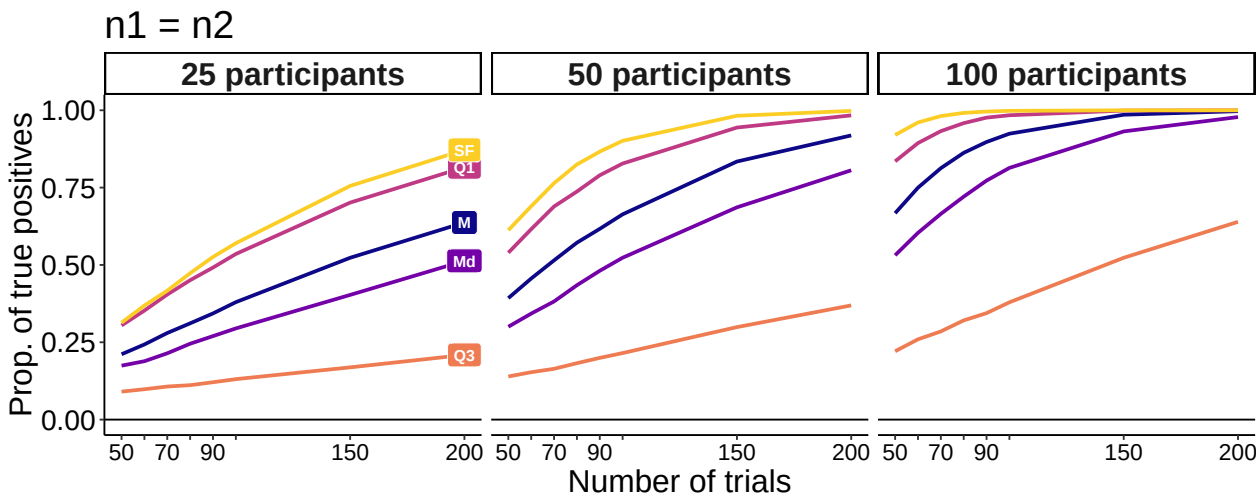
```
plot_tp <- function(FP.in, Est.in, ntseq, npseq){
  df <- tibble(FP = FP.in,
               Estimator = factor(rep(Est.in,
                 each=length(ntseq)*length(npseq))),
               Trials = rep(ntseq,length(npseq)*length(Est.in)),
               Participants = factor(rep(rep(npseq,each=length(ntseq)), length(Est.in)))
               )
```

```
labels <- c("25" = "25 participants", "50" = "50 participants", "100" = "100 participants")
  # make plot
p <- ggplot(df) + theme_classic() +
  geom_line(aes(x=Trials, y=FP, colour = Estimator), size = 1) +
  geom_abline(intercept=0, slope=0, colour="black") +
  scale_colour_viridis_d(option = "plasma", end=0.9) +
  scale_x_continuous(breaks = ntseq,
                     labels = c("50","","70","","90","","150","200")) +
  coord_cartesian(ylim=c(0,1)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 13, colour="black"),
        axis.text.y = element_text(size = 16, colour="black"),
        axis.title.y = element_text(size = 18),
        legend.key.width = unit(1.5,"cm"),
        legend.position = "right",#c(0.6,0.1),
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.text = element_text(size=18, face="bold"),
        strip.background = element_rect(colour="black", fill="white"),
        title = element_text(size = 20)) +
  labs(x = "Number of trials", y = "Prop. of true positives") +
  guides(colour = guide_legend(override.aes = list(size=3))) + # make thicker legend lines
  facet_grid(cols = vars(Participants), labeller=labeller(Participants = labels))
p
}
```

M, Md, Q1, Q3, SF

```
# add labels to facet 1
lab_text <- tibble(FP = c(fp.m.m[length(ntseq), 1],
                          fp.md.md[length(ntseq), 1],
                          fp.25.md[length(ntseq), 1],
                          fp.75.md[length(ntseq), 1],
                          fp.sfqt8.tm20[length(ntseq), 1]),
                   Estimator = factor(c("Mean", "Median", "Q1", "Q3", "SF")),
                   lab = c("M", "Md", "Q1", "Q3", "SF"),
                   Trials = rep(ntseq[length(ntseq)],5),
                   Participants = factor(25,levels = c("25","50","100")))

FP.in <- c(as.vector(fp.m.m),as.vector(fp.md.md),
       as.vector(fp.25.md),as.vector(fp.75.md),as.vector(fp.sfqt8.tm20))
Est.in <- c("Mean", "Median", "Q1", "Q3", "SF")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("n1 = n2") +
     theme(legend.position = "none")

# get colours to re-use in Tau plot + set label colours
g <- ggplot_build(p)
cm <- unique(g$data[[1]]["colour"]) # save colourmap

p <- p +  geom_label(data = lab_text,
```

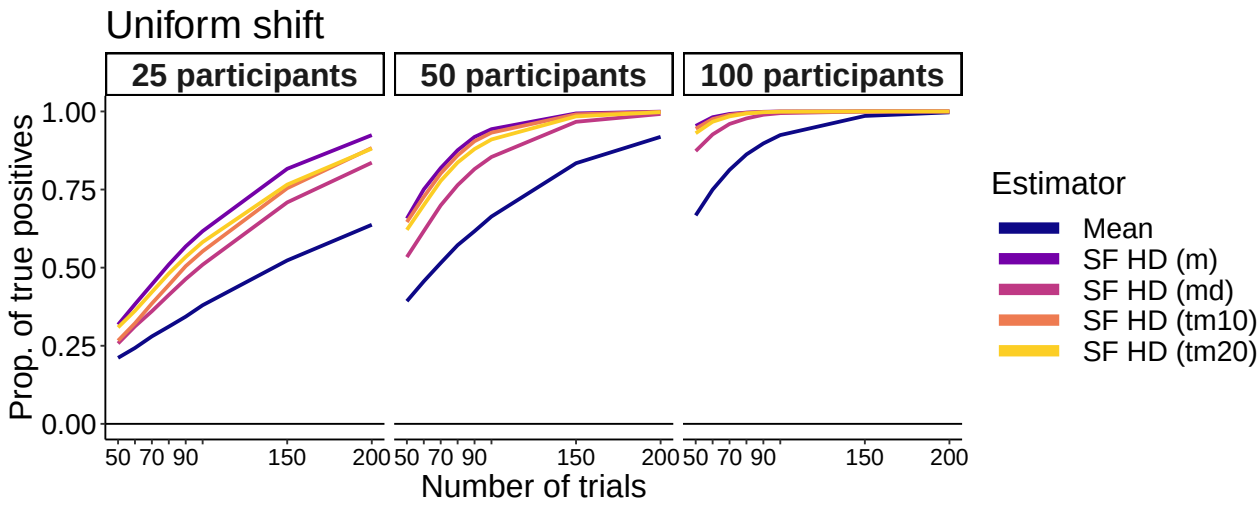
```
                    aes(x = Trials, y = FP, label = lab, fill = Estimator),
                        colour = 'white', fontface = "bold", size = 3) +
          scale_fill_manual(values = cm$colour, guide = FALSE)
p
```




```
p.uni.res <- p
```

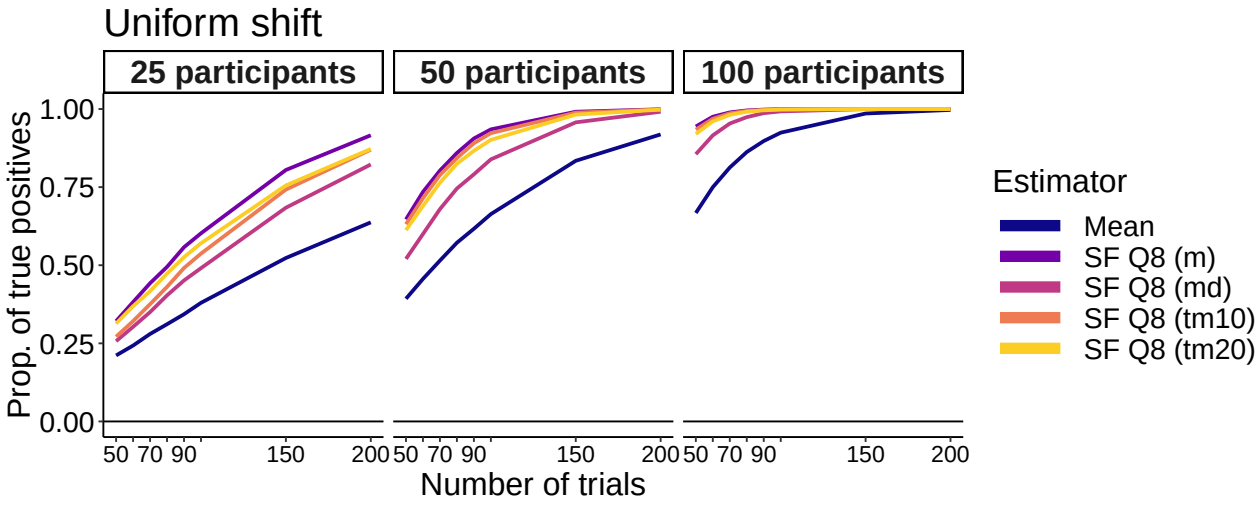
M, SF HD

```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20))
Est.in <- c("Mean","SF HD (m)", "SF HD (md)", "SF HD (tm10)","SF HD (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Uniform shift")
p
```



M, SF QT8

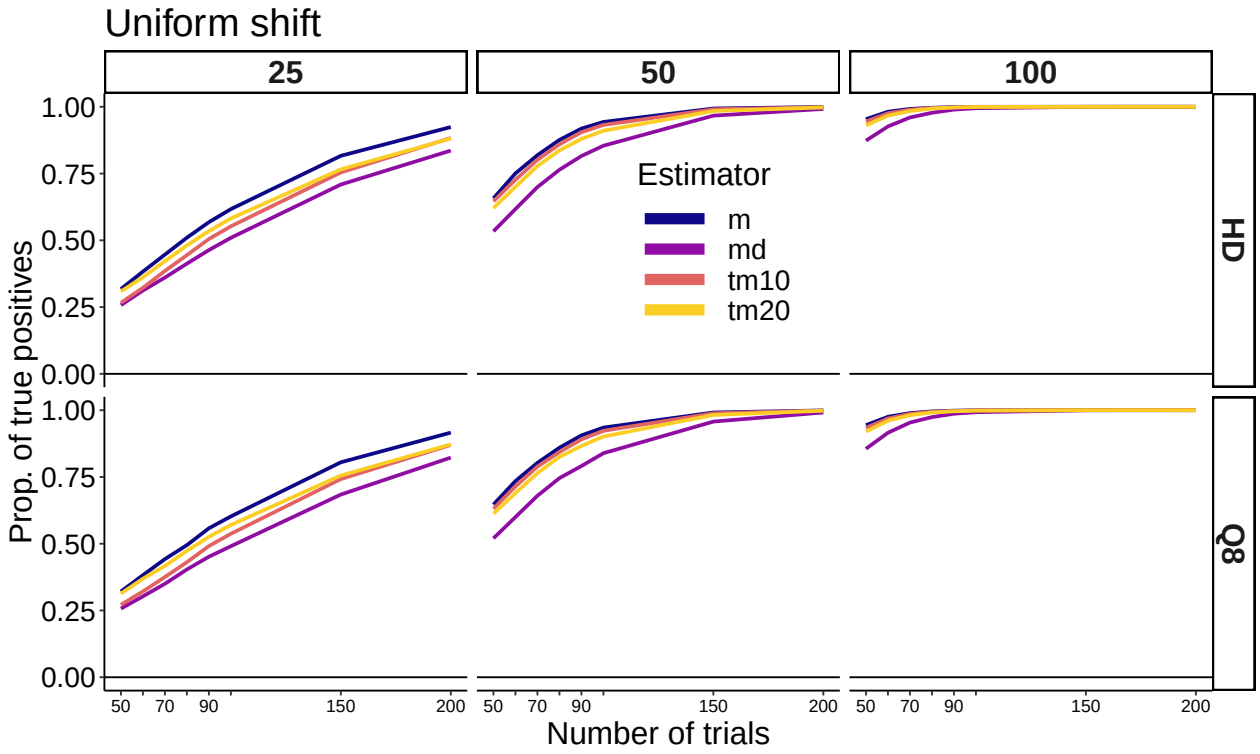
```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20))
Est.in <- c("Mean","SF Q8 (m)", "SF Q8 (md)", "SF Q8 (tm10)","SF Q8 (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Uniform shift")
p
```



Compare SF results

```
df <- tibble(FP = c(as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20)),
             Estimator = factor(rep(rep(c("m", "md", "tm10","tm20"),2),
               each=length(ntseq)*length(npseq))),
             Type = factor(rep(c("Q8","HD"), each = length(ntseq)*length(npseq)*4)),
             Trials = rep(ntseq,length(npseq)*8),
             Participants = factor(rep(rep(npseq,each=length(ntseq)),8))
             )
# make plot
p <- ggplot(df) + theme_classic() +
  geom_line(aes(x=Trials, y=FP, colour = Estimator), size = 1) +
  geom_abline(intercept=0, slope=0, colour="black") +
  scale_colour_viridis_d(option = "plasma", end=0.9) +
  scale_x_continuous(breaks = ntseq,
                     labels = c("50","","70","","90","","150","200")) +
  coord_cartesian(ylim=c(0,1)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 10, colour="black"),
        axis.text.y = element_text(size = 16, colour="black"),
        axis.title.y = element_text(size = 18),
        legend.key.width = unit(1.5,"cm"),
        legend.position = c(0.55,0.75),
```

```
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.text = element_text(size=18, face="bold"),
        strip.background = element_rect(colour="black", fill="white")) +
  labs(x = "Number of trials", y = "Prop. of true positives") +
  guides(colour = guide_legend(override.aes = list(size=3))) + # make thicker legend lines
  ggtitle("Uniform shift") +
  facet_grid(rows = vars(Type),
             cols = vars(Participants))
p
```

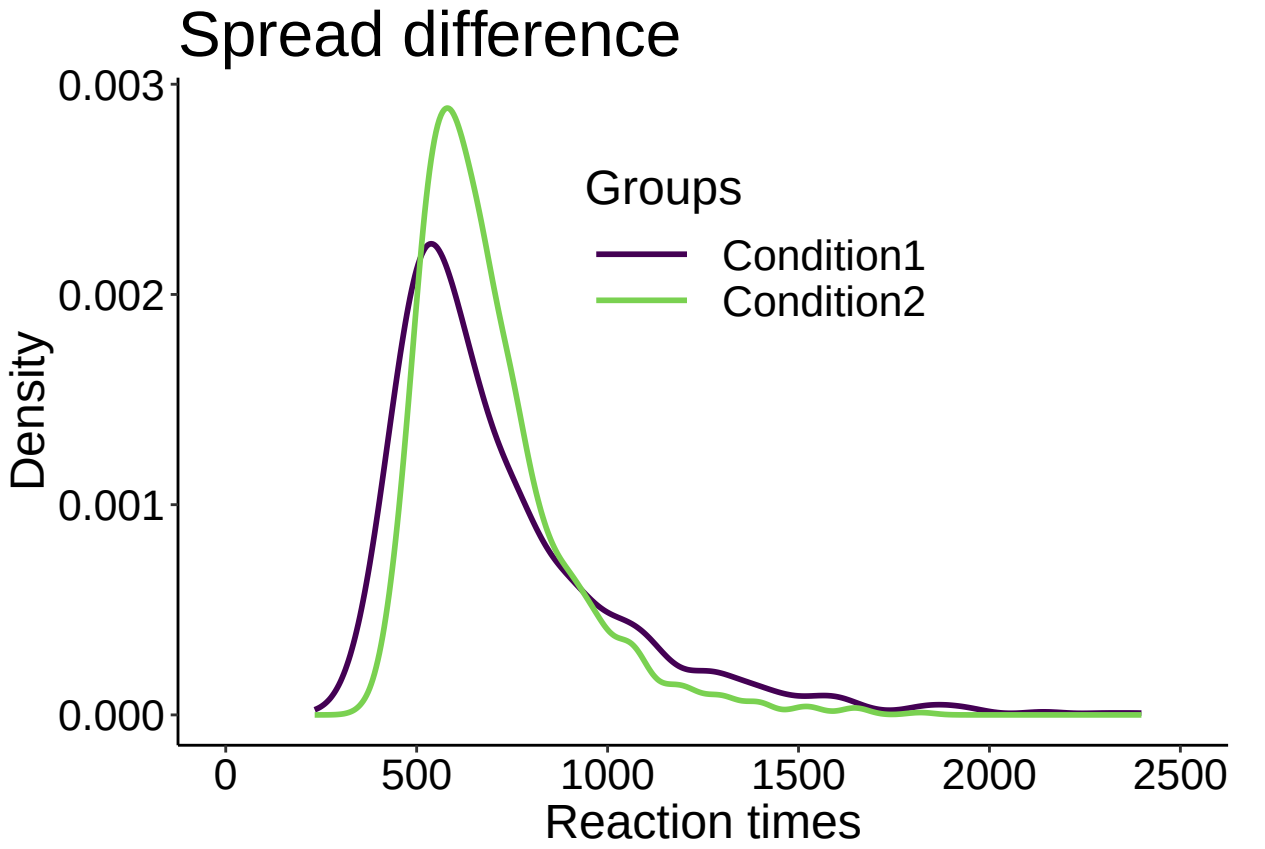


Group simulation 2a: same number of trials in each group, difference in spread

KDE

```
nt <- 1000
# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 1.5

set.seed(777)
g1 <- rexgauss(nt, mu = mu, sigma = sigma, tau = tau)
md.g1 <- median(g1)
```

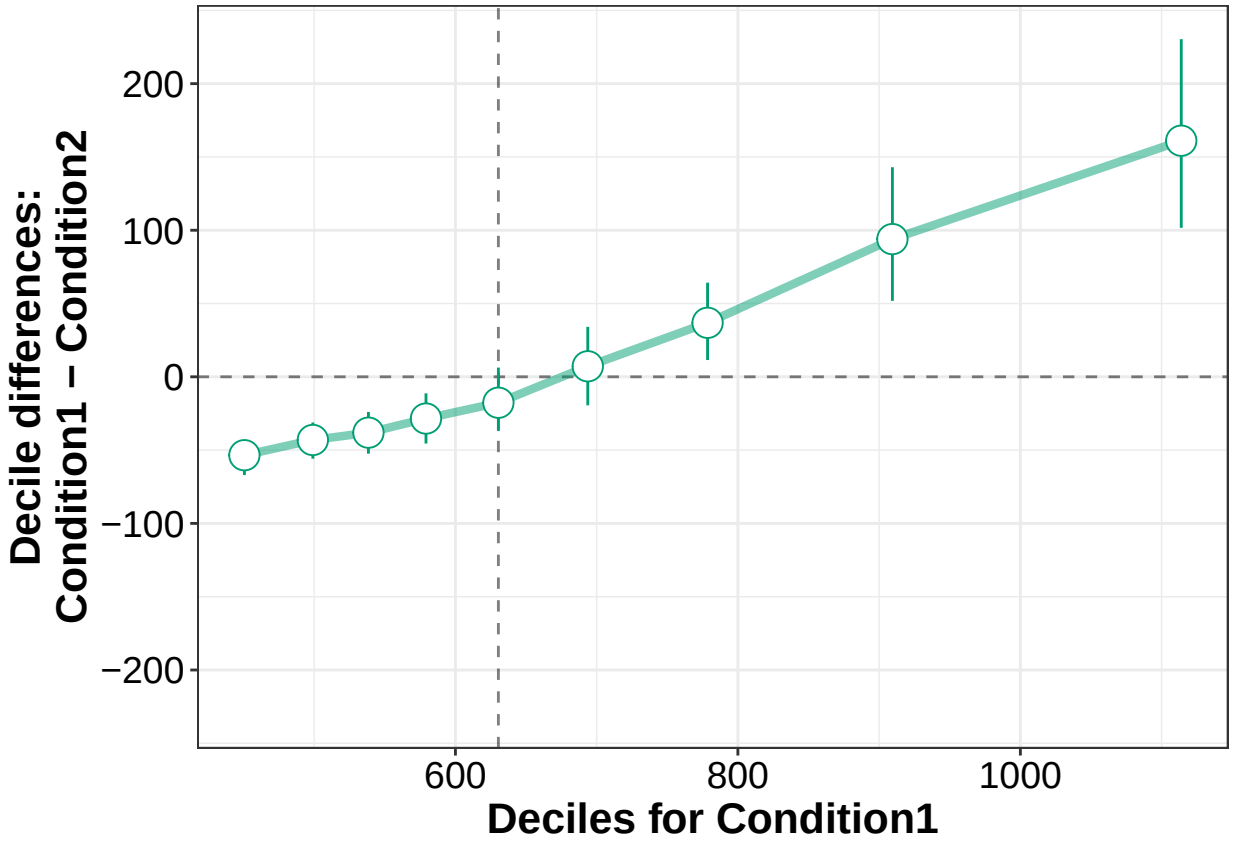
```r
g1 <- (g1 - md.g1) * ES + md.g1 # spread around the median
g2 <- rexgauss(nt, mu = mu, sigma = sigma, tau = tau)
df <- mkt2(g1, g2, group_labels = c("Condition1", "Condition2"))

p <- ggplot(df, aes(x = obs)) + theme_classic() +
  stat_density(aes(colour = gr), geom="line",position="identity", size=1) +
  scale_colour_viridis_d(end = 0.8) +
  coord_cartesian(xlim = c(0, 2500)) +
  theme(axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 16, colour="black"),
        axis.text.y = element_text(size = 16, colour="black"),
        axis.title.y = element_text(size = 18),
        legend.key.width = unit(1.5,"cm"),
        legend.position = c(0.55,0.75),
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        title = element_text(size=20)) +
  labs(x = "Reaction times", y = "Density", colour = "Groups") +
  ggtitle("Spread difference")
p
```




```r
p.spr.dist <- p
```

Shift function

```
out <- shifthd_pbci(df, nboot = 200, adj_ci = FALSE)
p <- plot_sf(out, plot_theme = 1)[[1]] +
     theme(axis.text = element_text(size = 16, colour="black"))
p
```

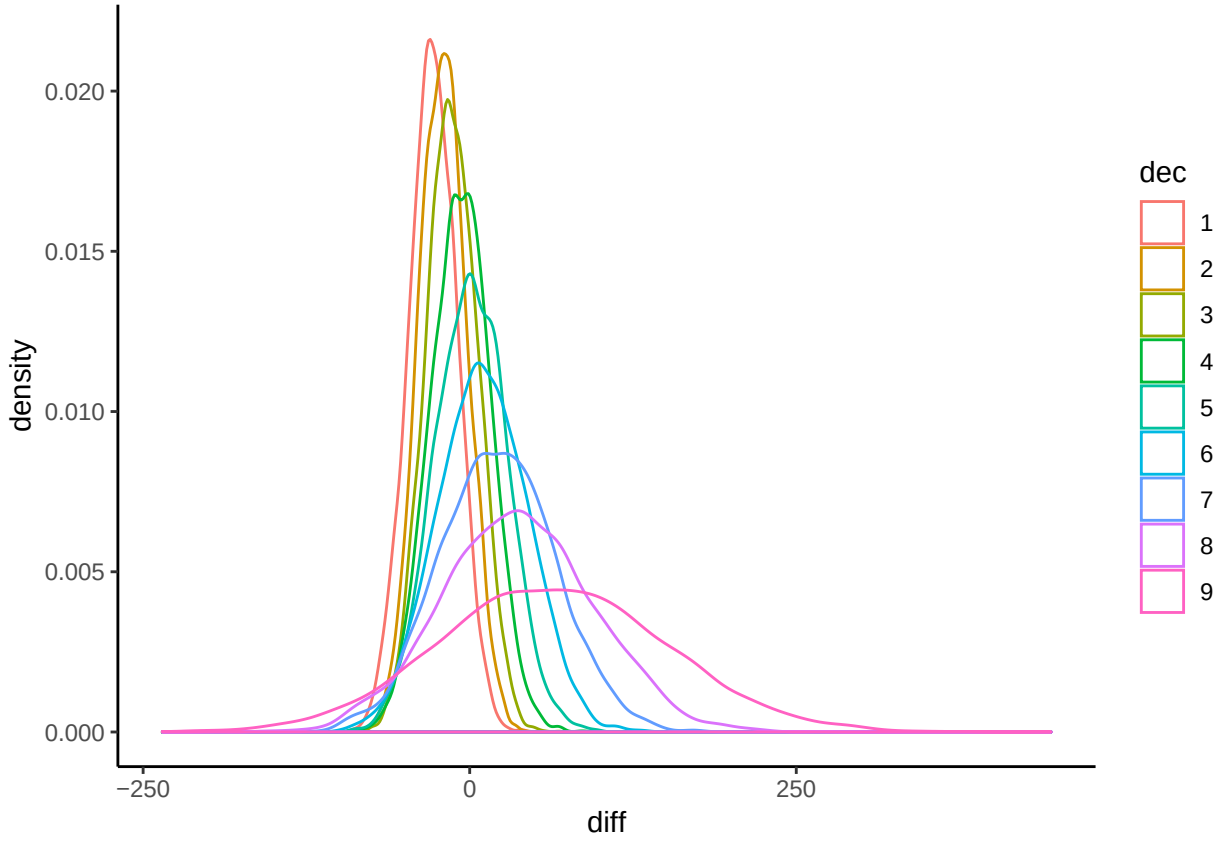



```
p.spr.sf <- p
```

Decile sampling distributions

```
nt <- 100 # trials
np <- 5000 # participants

# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 1.2

mc.data1 <- array(rexgauss(nt*np, mu = mu, sigma = sigma, tau = tau),
  dim = c(nt, np))
md1 <- apply(mc.data1, 2, median)
mc.data1 <- (mc.data1 - matrix(rep(md1,each=nt),nrow = nt)) * ES + matrix(rep(md1,each=nt),nrow = nt)
mc.data2 <- array(rexgauss(nt*np, mu = mu, sigma = sigma, tau = tau),
```

```
  dim = c(nt, np))

# array of differences: 9 deciles x participants
mc.hd <- apply(mc.data1, 2, hdseq) - apply(mc.data2, 2, hdseq)

df <- tibble(diff = as.vector(mc.hd),
             dec = factor(rep(seq(1,9), np)))
ggplot(df, aes(x = diff, colour = dec)) + theme_classic() +
  geom_density()
```



Skewness of sampling distributions

```
apply(mc.hd, 1, skew)
```

```
## [1] -0.01705125  0.03361391  0.02333503  0.01776792  0.01422891  0.03301970
## [7]  0.04040877  0.07363016  0.06918873
```

Kurtosis of sampling distributions

```
apply(mc.hd, 1, kurt)
```

```
## [1] 2.945804 2.869039 2.939383 3.023132 2.988433 2.986641 2.953462 3.037189
## [9] 3.049922
```

Simulation

```
npseq <- c(25, 50, 100)
ntseq <- c(seq(50, 100, 10), 150, 200)
maxnp <- max(npseq) # max number of participants
maxnt <- max(ntseq) # max number of trials
nsim <- 10000 # simulation samples
qseq <- seq(0.1,0.9,0.1) # define deciles for quantile function

# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 1.1

alpha <- 0.05

# declare matrices of results - save all iterations
res.m.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.md.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.25.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.75.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfhd.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfqt8.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

pvals.0.hd <- vector(mode = "numeric", length = 9)
pvals.10.hd <- vector(mode = "numeric", length = 9)
pvals.20.hd <- vector(mode = "numeric", length = 9)

pvals.0.qt8 <- vector(mode = "numeric", length = 9)
pvals.10.qt8 <- vector(mode = "numeric", length = 9)
pvals.20.qt8 <- vector(mode = "numeric", length = 9)

# create 2 populations
set.seed(21)
npop <- 1000000
pop2 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau)
md2 <- median(pop2)
pop1 <- (pop2-md2) * ES + md2

beep(3)

# simulation samples
mc.data1 <- array(sample(pop1, maxnt*maxnp*nsim, replace = TRUE), dim = c(maxnt, maxnp, nsim))
mc.data2 <- array(sample(pop2, maxnt*maxnp*nsim, replace = TRUE), dim = c(maxnt, maxnp, nsim))
```

```r
for(TR in 1:length(ntseq)){ # number of trials
  print(paste0("number of trials: ",ntseq[TR],"..."))
  beep(2)

  for(PT in 1:length(npseq)){ # number of participants
    print(paste0("number of participants: ",npseq[PT],"..."))

    # compute estimates
    todo1 <- mc.data1[1:ntseq[TR], 1:npseq[PT],]
    todo2 <- mc.data2[1:ntseq[TR], 1:npseq[PT],]
    mc.m <- apply(todo1, c(2,3), mean) - apply(todo2, c(2,3), mean) # diff in mean RTs
    mc.md <- apply(todo1, c(2,3), median) - apply(todo2, c(2,3), median) # diff in median RTs
    mc.25 <- apply(todo1, c(2,3), quantile, probs = .25) - apply(todo2, c(2,3), quantile, probs = .25)
    mc.75 <- apply(todo1, c(2,3), quantile, probs = .75) - apply(todo2, c(2,3), quantile, probs = .75)

    # tests for each simulation ====================
    # one-sample t-test
    res.m.m.sig[, TR, PT] <- ttest.sig(mc.m)
    # parametric median test
    res.md.md.sig[, TR, PT] <- apply(mc.md, 2, sint.sig)
    res.25.md.sig[, TR, PT] <- apply(mc.25, 2, sint.sig)
    res.75.md.sig[, TR, PT] <- apply(mc.75, 2, sint.sig)

    # shift function
    # array of differences: 9 deciles x participants x simulations
    mc.hd <- apply(todo1, c(2,3), hdseq) - apply(todo2, c(2,3), hdseq)
    mc.qt8 <- apply(todo1, c(2,3), quantile, probs = qseq, type = 8) -
              apply(todo2, c(2,3), quantile, probs = qseq, type = 8)
    # median test for each decile distribution
    pval.hd <- apply(mc.hd, c(1,3), sintv2.pval)
    pval.qt8 <- apply(mc.qt8, c(1,3), sintv2.pval)
    # trimmed mean tests for each decile distribution
    tvals.0.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0)
    tvals.10.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.1)
    tvals.20.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.2)
    tvals.0.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0)
    tvals.10.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.1)
    tvals.20.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.2)
    df0 <- df.yuen(mc.hd[1,,1], tr = 0)
    df10 <- df.yuen(mc.hd[1,,1], tr = 0.1)
    df20 <- df.yuen(mc.hd[1,,1], tr = 0.2)
    for(S in 1:nsim){
      res.sfhd.md.sig[S, TR, PT] <- sum(p.adjust(pval.hd[,S], method = "hochberg") <= alpha)
      res.sfqt8.md.sig[S, TR, PT] <- sum(p.adjust(pval.qt8[,S], method = "hochberg") <= alpha)

      for(Q in 1:9){
        pvals.0.hd[Q] <- cpval(tvals.0.hd[S,Q], df0)
        pvals.10.hd[Q] <- cpval(tvals.10.hd[S,Q], df10)
        pvals.20.hd[Q] <- cpval(tvals.20.hd[S,Q], df20)
        pvals.0.qt8[Q] <- cpval(tvals.0.qt8[S,Q], df0)
        pvals.10.qt8[Q] <- cpval(tvals.10.qt8[S,Q], df10)
        pvals.20.qt8[Q] <- cpval(tvals.20.qt8[S,Q], df20)
      }
```

```
      res.sfhd.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.hd, method = "hochberg") <= alpha)
      res.sfhd.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.hd, method = "hochberg") <= alpha)
      res.sfhd.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.hd, method = "hochberg") <= alpha)
      res.sfqt8.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.qt8, method = "hochberg") <= alpha)
      res.sfqt8.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.qt8, method = "hochberg") <= alpha)
      res.sfqt8.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.qt8, method = "hochberg") <= alpha)
    }
  }
}

save(
  res.m.m.sig,
  res.md.md.sig,
  res.25.md.sig,
  res.75.md.sig,
  res.sfhd.md.sig,
  res.sfhd.tm10.sig,
  res.sfhd.tm20.sig,
  res.sfhd.m.sig,
  res.sfqt8.md.sig,
  res.sfqt8.tm10.sig,
  res.sfqt8.tm20.sig,
  res.sfqt8.m.sig,
  ntseq,
  npseq,
  mu,
  ES,
  sigma,
  tau,
  nsim,
  file=paste0('./data/sim_gp_tp2.RData'))
beep(8)
```

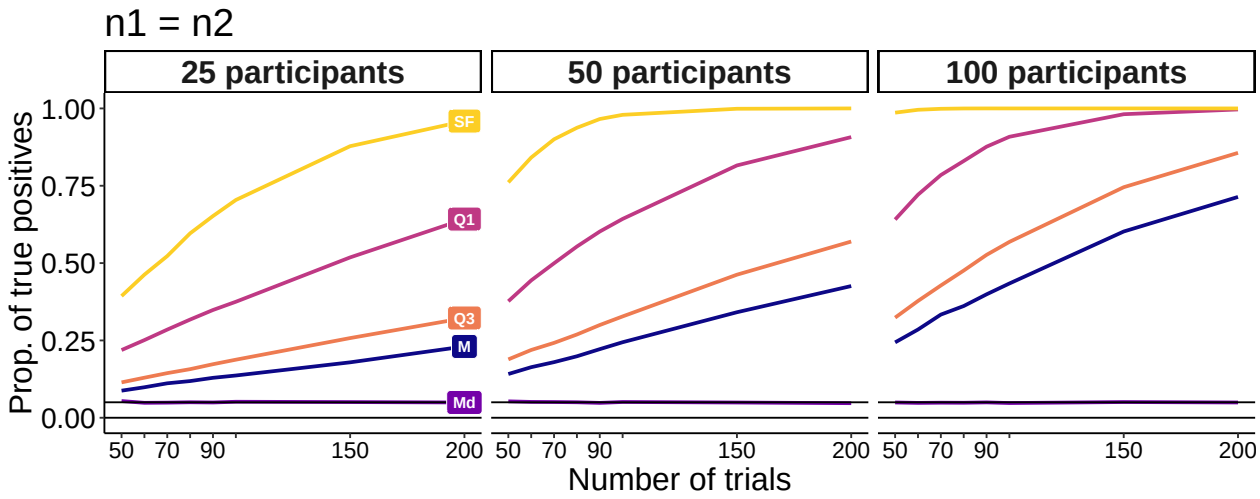
Compute true positive probability

```
load('./data/sim_gp_tp2.RData')
fp.m.m <- apply(res.m.m.sig, c(2,3), mean)
fp.md.md <- apply(res.md.md.sig, c(2,3), mean)
fp.25.md <- apply(res.25.md.sig, c(2,3), mean)
fp.75.md <- apply(res.75.md.sig, c(2,3), mean)
fp.sfhd.md <- apply(res.sfhd.md.sig > 0, c(2,3), mean)
fp.sfhd.m <- apply(res.sfhd.m.sig > 0, c(2,3), mean)
fp.sfhd.tm10 <- apply(res.sfhd.tm10.sig > 0, c(2,3), mean)
fp.sfhd.tm20 <- apply(res.sfhd.tm20.sig > 0, c(2,3), mean)
fp.sfqt8.md <- apply(res.sfqt8.md.sig > 0, c(2,3), mean)
fp.sfqt8.m <- apply(res.sfqt8.m.sig > 0, c(2,3), mean)
fp.sfqt8.tm10 <- apply(res.sfqt8.tm10.sig > 0, c(2,3), mean)
fp.sfqt8.tm20 <- apply(res.sfqt8.tm20.sig > 0, c(2,3), mean)
```

Illustrate results

M, Md, Q1, Q3, SF

```
# add labels to facet 1
lab_text <- tibble(FP = c(fp.m.m[length(ntseq), 1],
                          fp.md.md[length(ntseq), 1],
                          fp.25.md[length(ntseq), 1],
                          fp.75.md[length(ntseq), 1],
                          fp.sfqt8.tm20[length(ntseq), 1]),
                   Estimator = factor(c("Mean", "Median", "Q1", "Q3", "SF")),
                   lab = c("M", "Md", "Q1", "Q3", "SF"),
                   Trials = rep(ntseq[length(ntseq)],5),
                   Participants = factor(25,levels = c("25","50","100")))

FP.in <- c(as.vector(fp.m.m),as.vector(fp.md.md),
        as.vector(fp.25.md),as.vector(fp.75.md),as.vector(fp.sfqt8.tm20))
Est.in <- c("Mean", "Median", "Q1", "Q3", "SF")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("n1 = n2") +
      theme(legend.position = "none") +
      geom_hline(yintercept = 0.05)

p <- p +  geom_label(data = lab_text,
                     aes(x = Trials, y = FP, label = lab, fill = Estimator),
                         colour = 'white', fontface = "bold", size = 3) +
          scale_fill_manual(values = cm$colour, guide = FALSE)
p
```

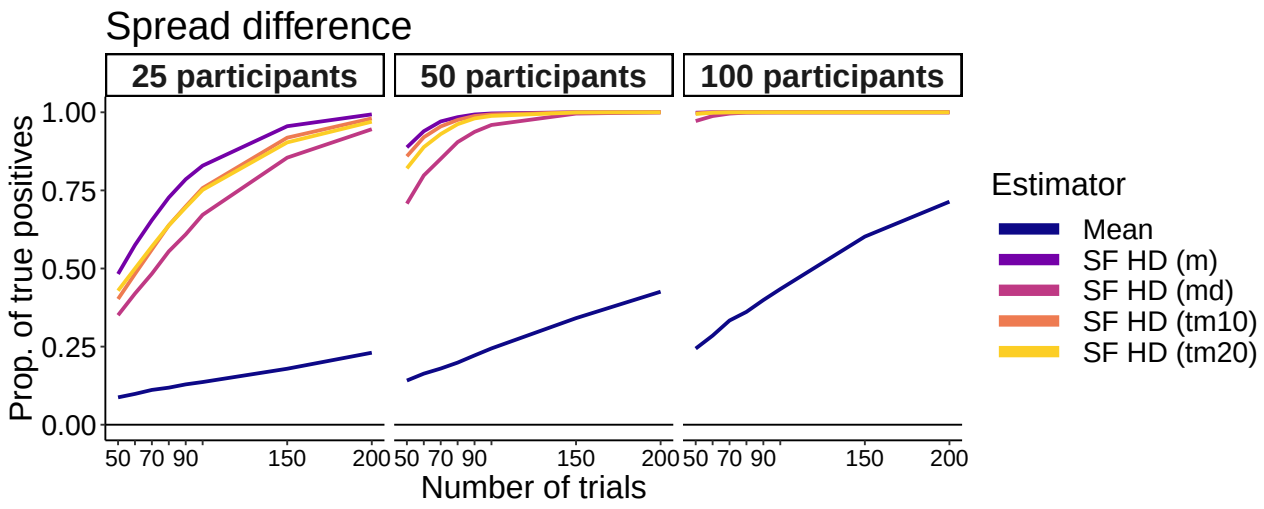



```
p.spr.res <- p
```

M, SF HD

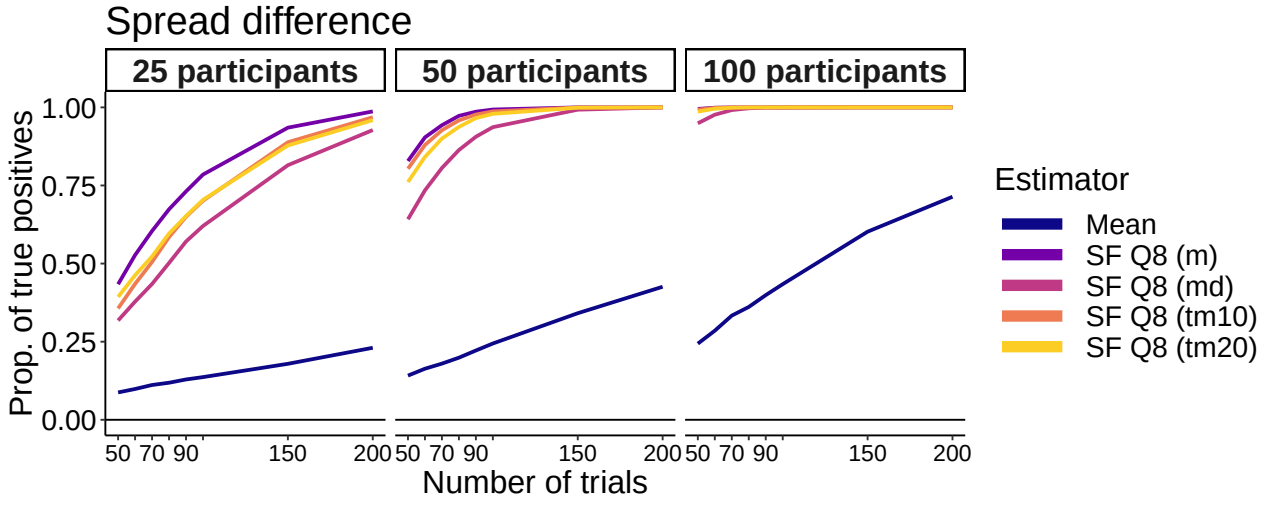
```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20))
Est.in <- c("Mean","SF HD (m)", "SF HD (md)", "SF HD (tm10)","SF HD (tm20)")
```

```
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Spread difference")
p
```



M, SF QT8

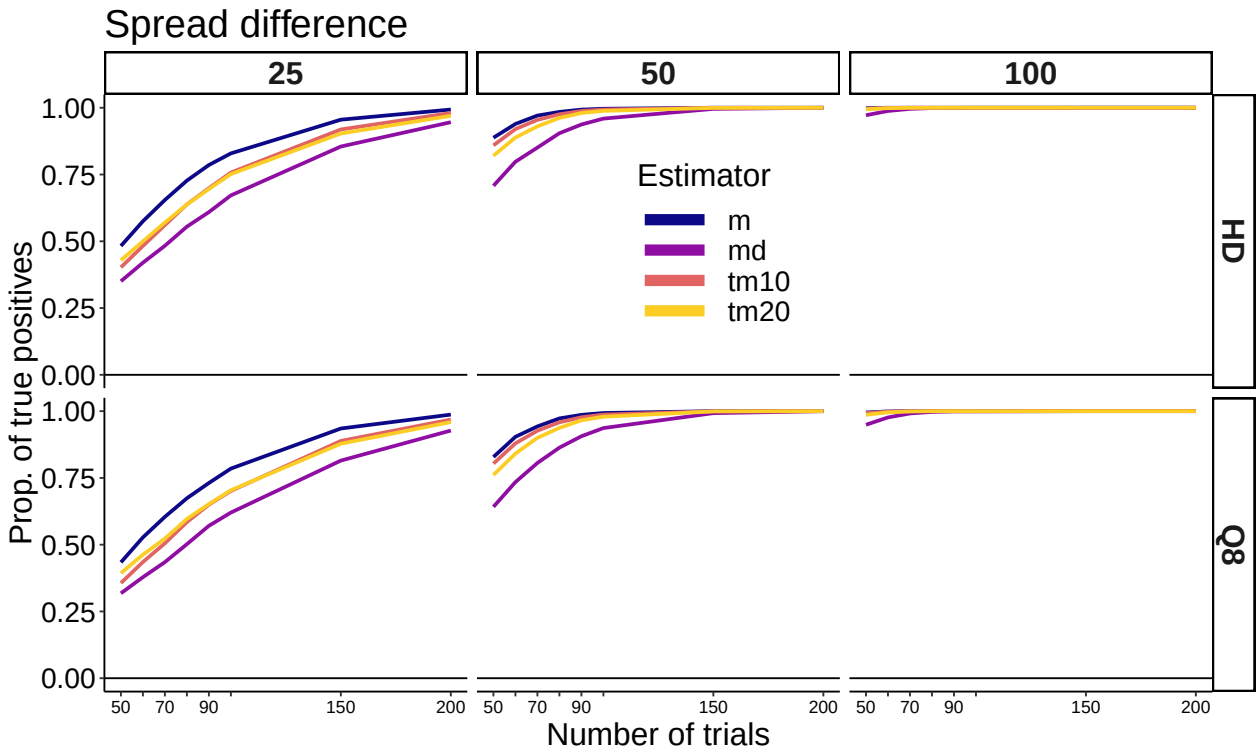
```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20))
Est.in <- c("Mean","SF Q8 (m)", "SF Q8 (md)", "SF Q8 (tm10)","SF Q8 (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Spread difference")
p
```



Compare SF results

```
df <- tibble(FP = c(as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20)),
```

```
            Estimator = factor(rep(rep(c("m", "md", "tm10","tm20"),2),
              each=length(ntseq)*length(npseq))),
            Type = factor(rep(c("Q8","HD"), each = length(ntseq)*length(npseq)*4)),
            Trials = rep(ntseq,length(npseq)*8),
            Participants = factor(rep(rep(npseq,each=length(ntseq)),8))
            )
# make plot
p <- ggplot(df) + theme_classic() +
  geom_line(aes(x=Trials, y=FP, colour = Estimator), size = 1) +
  geom_abline(intercept=0, slope=0, colour="black") +
  scale_colour_viridis_d(option = "plasma", end=0.9) +
  scale_x_continuous(breaks = ntseq,
                     labels = c("50","","70","","90","","150","200")) +
  coord_cartesian(ylim=c(0,1)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 10, colour="black"),
        axis.text.y = element_text(size = 16, colour="black"),
        axis.title.y = element_text(size = 18),
        legend.key.width = unit(1.5,"cm"),
        legend.position = c(0.55,0.75),
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.text = element_text(size=18, face="bold"),
        strip.background = element_rect(colour="black", fill="white")) +
  labs(x = "Number of trials", y = "Prop. of true positives") +
  guides(colour = guide_legend(override.aes = list(size=3))) + # make thicker legend lines
  ggtitle("Spread difference") +
  facet_grid(rows = vars(Type),
             cols = vars(Participants))
p
```

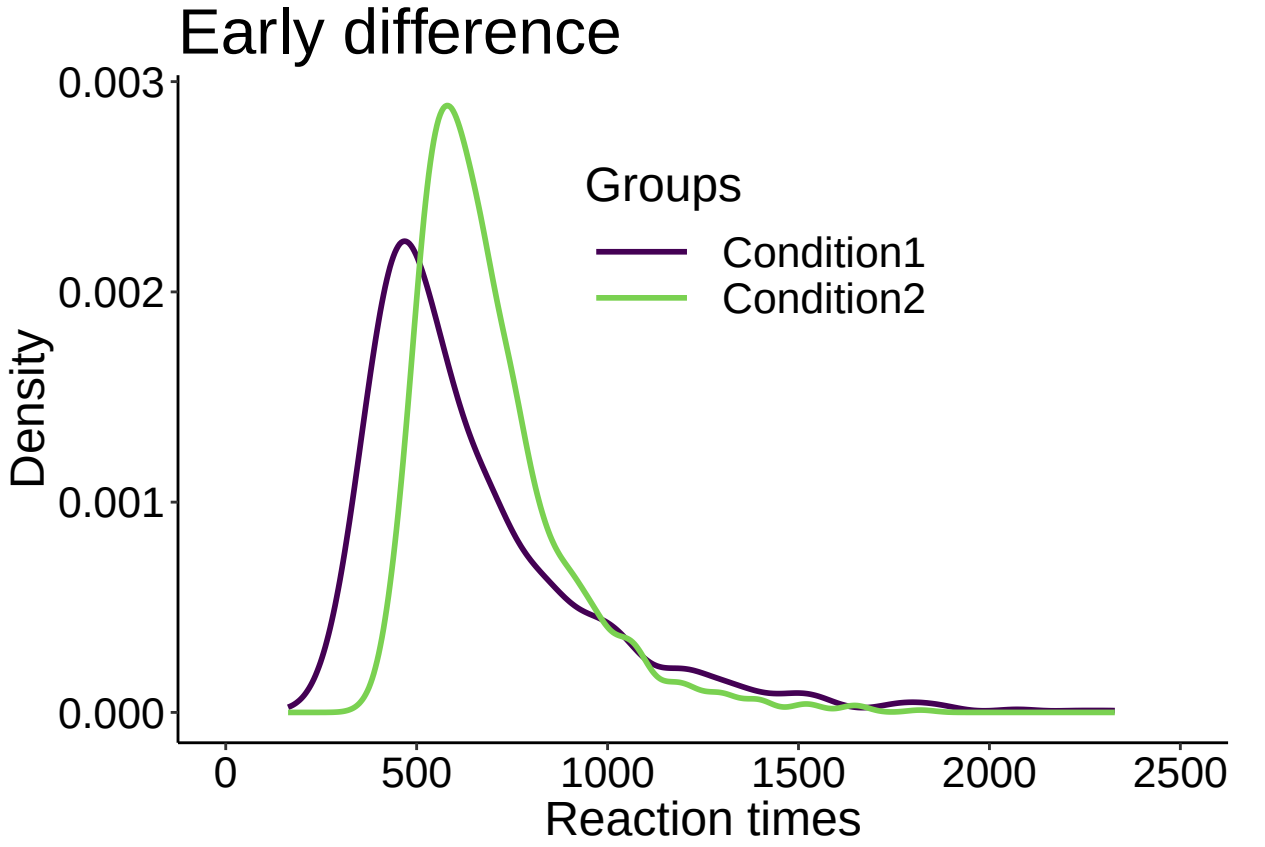


Group simulation 2b: same number of trials in each group, early difference

KDE

```
nt <- 1000
# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 1.5

set.seed(777)
g1 <- rexgauss(nt, mu = mu, sigma = sigma, tau = tau)
md.g1 <- median(g1)
q3.g1 <- quantile(g1, probs = 0.75)
# g1 <- (g1 - md.g1) * ES + md.g1 # spread around the median
g1 <- (g1 - q3.g1) * ES + q3.g1 # spread around the 3rd quartile
g2 <- rexgauss(nt, mu = mu, sigma = sigma, tau = tau)
df <- mkt2(g1, g2, group_labels = c("Condition1", "Condition2"))

p <- ggplot(df, aes(x = obs)) + theme_classic() +
  stat_density(aes(colour = gr), geom="line",position="identity", size=1) +
  scale_colour_viridis_d(end = 0.8) +
  coord_cartesian(xlim = c(0, 2500)) +
  theme(axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 16, colour="black"),
```

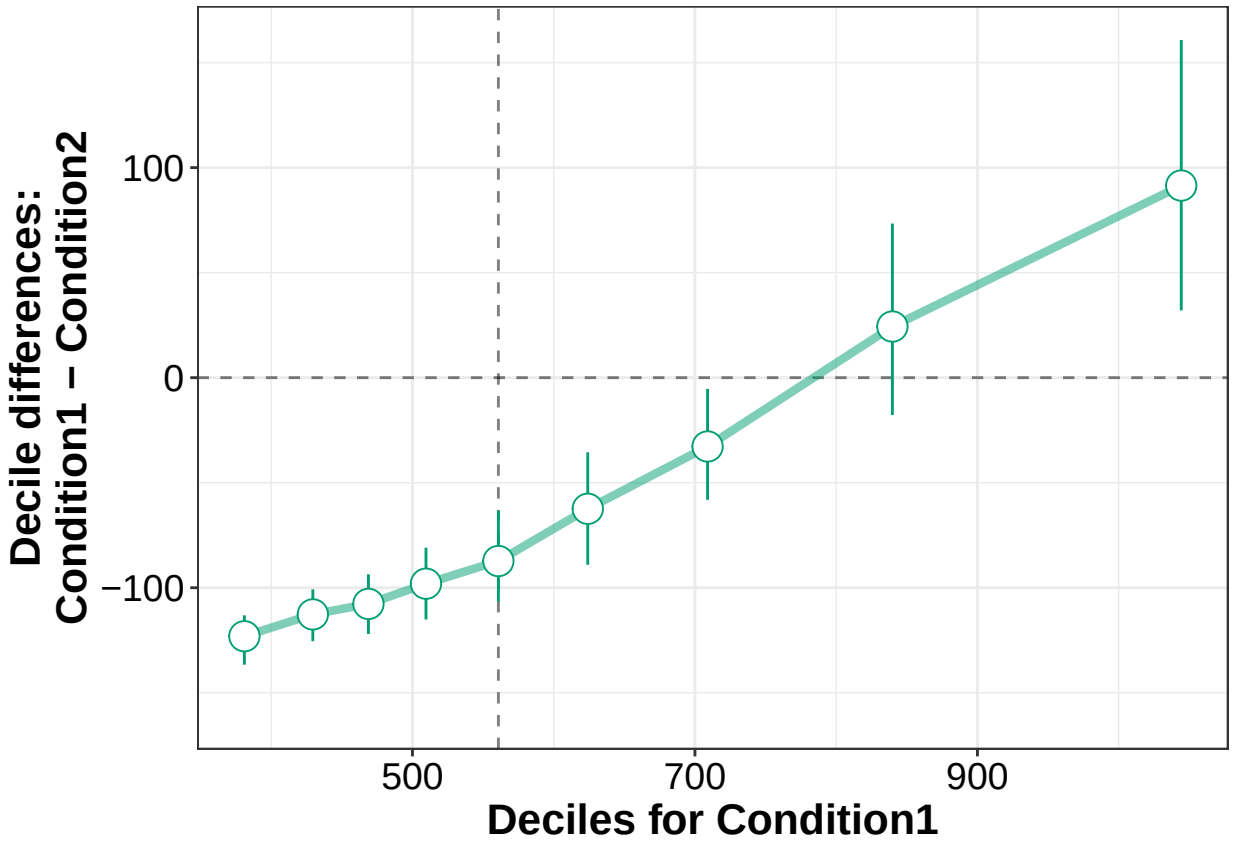
```
        axis.text.y = element_text(size = 16, colour="black"),
        axis.title.y = element_text(size = 18),
        legend.key.width = unit(1.5,"cm"),
        legend.position = c(0.55,0.75),
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        title = element_text(size=20)) +
  labs(x = "Reaction times", y = "Density", colour = "Groups") +
  ggtitle("Early difference")
p
```




```
p.early.dist <- p
```

Shift function

```
out <- shifthd_pbci(df, nboot = 200, adj_ci = FALSE)
p <- plot_sf(out, plot_theme = 1)[[1]] +
     theme(axis.text = element_text(size = 16, colour="black"))
p
```

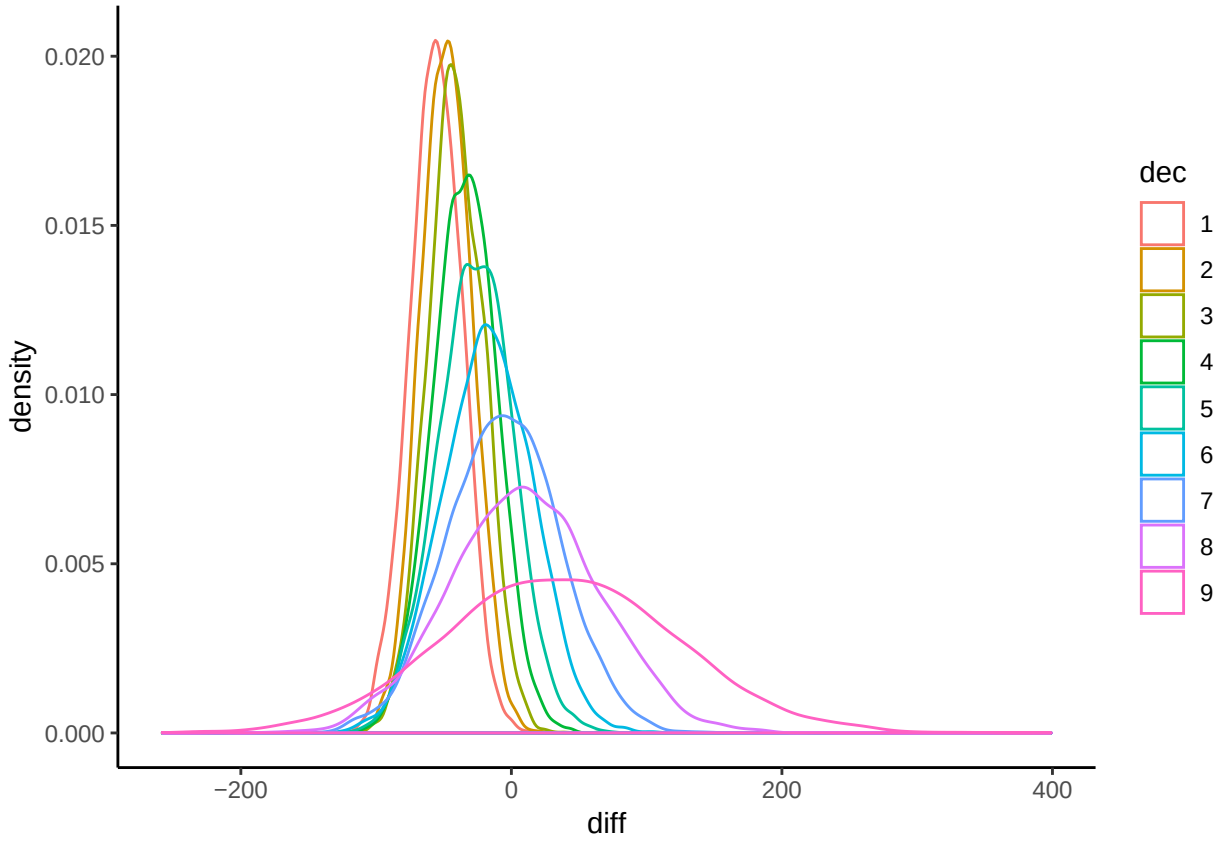



```
p.early.sf <- p
```

Decile sampling distributions

```
nt <- 100 # trials
np <- 5000 # participants

# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 1.2

mc.data1 <- array(rexgauss(nt*np, mu = mu, sigma = sigma, tau = tau),
  dim = c(nt, np))
md1 <- apply(mc.data1, 2, quantile, probs = 0.75)
mc.data1 <- (mc.data1 - matrix(rep(md1,each=nt),nrow = nt)) * ES + matrix(rep(md1,each=nt),nrow = nt)
mc.data2 <- array(rexgauss(nt*np, mu = mu, sigma = sigma, tau = tau),
  dim = c(nt, np))

# array of differences: 9 deciles x participants
mc.hd <- apply(mc.data1, 2, hdseq) - apply(mc.data2, 2, hdseq)

df <- tibble(diff = as.vector(mc.hd),
             dec = factor(rep(seq(1,9), np)))
ggplot(df, aes(x = diff, colour = dec)) + theme_classic() +
```

```
  geom_density()
```



Skewness of sampling distributions

```
apply(mc.hd, 1, skew)
```

```
## [1] -0.027046719  0.036533460  0.026097716  0.011898479 -0.007940779
## [6] -0.002785055 -0.002978714  0.042967179  0.060749018
```

Kurtosis of sampling distributions

```
apply(mc.hd, 1, kurt)
```

```
## [1] 2.925838 2.945597 3.015089 3.070177 3.007128 3.002834 2.943905 3.026318
## [9] 3.042560
```

Simulation

```
npseq <- c(25, 50, 100)
ntseq <- c(seq(50, 100, 10), 150, 200)
maxnp <- max(npseq) # max number of participants
maxnt <- max(ntseq) # max number of trials
nsim <- 10000 # simulation samples
```

```r
qseq <- seq(0.1,0.9,0.1) # define deciles for quantile function

# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 1.1

alpha <- 0.05

# declare matrices of results - save all iterations
res.m.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.md.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.25.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.75.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfhd.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfqt8.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

pvals.0.hd <- vector(mode = "numeric", length = 9)
pvals.10.hd <- vector(mode = "numeric", length = 9)
pvals.20.hd <- vector(mode = "numeric", length = 9)

pvals.0.qt8 <- vector(mode = "numeric", length = 9)
pvals.10.qt8 <- vector(mode = "numeric", length = 9)
pvals.20.qt8 <- vector(mode = "numeric", length = 9)

# create 2 populations
set.seed(21)
npop <- 1000000
pop2 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau)
q3.2 <- quantile(pop2, probs = 0.75)
pop1 <- (pop2-q3.2) * ES + q3.2

beep(3)

# simulation samples
mc.data1 <- array(sample(pop1, maxnt*maxnp*nsim, replace = TRUE), dim = c(maxnt, maxnp, nsim))
mc.data2 <- array(sample(pop2, maxnt*maxnp*nsim, replace = TRUE), dim = c(maxnt, maxnp, nsim))

for(TR in 1:length(ntseq)){ # number of trials
  print(paste0("number of trials: ",ntseq[TR],"..."))
  beep(2)

  for(PT in 1:length(npseq)){ # number of participants
    print(paste0("number of participants: ",npseq[PT],"..."))
```

```r
  # compute estimates
  todo1 <- mc.data1[1:ntseq[TR], 1:npseq[PT],]
  todo2 <- mc.data2[1:ntseq[TR], 1:npseq[PT],]
  mc.m <- apply(todo1, c(2,3), mean) - apply(todo2, c(2,3), mean) # diff in mean RTs
  mc.md <- apply(todo1, c(2,3), median) - apply(todo2, c(2,3), median) # diff in median RTs
  mc.25 <- apply(todo1, c(2,3), quantile, probs = .25) - apply(todo2, c(2,3), quantile, probs = .25)
  mc.75 <- apply(todo1, c(2,3), quantile, probs = .75) - apply(todo2, c(2,3), quantile, probs = .75)

  # tests for each simulation ====================
  # one-sample t-test
  res.m.m.sig[, TR, PT] <- ttest.sig(mc.m)
  # parametric median test
  res.md.md.sig[, TR, PT] <- apply(mc.md, 2, sint.sig)
  res.25.md.sig[, TR, PT] <- apply(mc.25, 2, sint.sig)
  res.75.md.sig[, TR, PT] <- apply(mc.75, 2, sint.sig)

  # shift function
  # array of differences: 9 deciles x participants x simulations
  mc.hd <- apply(todo1, c(2,3), hdseq) - apply(todo2, c(2,3), hdseq)
  mc.qt8 <- apply(todo1, c(2,3), quantile, probs = qseq, type = 8) -
            apply(todo2, c(2,3), quantile, probs = qseq, type = 8)
  # median test for each decile distribution
  pval.hd <- apply(mc.hd, c(1,3), sintv2.pval)
  pval.qt8 <- apply(mc.qt8, c(1,3), sintv2.pval)
  # trimmed mean tests for each decile distribution
  tvals.0.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0)
  tvals.10.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.1)
  tvals.20.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.2)
  tvals.0.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0)
  tvals.10.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.1)
  tvals.20.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.2)
  df0 <- df.yuen(mc.hd[1,,1], tr = 0)
  df10 <- df.yuen(mc.hd[1,,1], tr = 0.1)
  df20 <- df.yuen(mc.hd[1,,1], tr = 0.2)
  for(S in 1:nsim){
    res.sfhd.md.sig[S, TR, PT] <- sum(p.adjust(pval.hd[,S], method = "hochberg") <= alpha)
    res.sfqt8.md.sig[S, TR, PT] <- sum(p.adjust(pval.qt8[,S], method = "hochberg") <= alpha)

    for(Q in 1:9){
      pvals.0.hd[Q] <- cpval(tvals.0.hd[S,Q], df0)
      pvals.10.hd[Q] <- cpval(tvals.10.hd[S,Q], df10)
      pvals.20.hd[Q] <- cpval(tvals.20.hd[S,Q], df20)
      pvals.0.qt8[Q] <- cpval(tvals.0.qt8[S,Q], df0)
      pvals.10.qt8[Q] <- cpval(tvals.10.qt8[S,Q], df10)
      pvals.20.qt8[Q] <- cpval(tvals.20.qt8[S,Q], df20)
    }
    res.sfhd.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.hd, method = "hochberg") <= alpha)
    res.sfhd.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.hd, method = "hochberg") <= alpha)
    res.sfhd.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.hd, method = "hochberg") <= alpha)
    res.sfqt8.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.qt8, method = "hochberg") <= alpha)
    res.sfqt8.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.qt8, method = "hochberg") <= alpha)
    res.sfqt8.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.qt8, method = "hochberg") <= alpha)
  }
```

```
  }
}

save(
  res.m.m.sig,
  res.md.md.sig,
  res.25.md.sig,
  res.75.md.sig,
  res.sfhd.md.sig,
  res.sfhd.tm10.sig,
  res.sfhd.tm20.sig,
  res.sfhd.m.sig,
  res.sfqt8.md.sig,
  res.sfqt8.tm10.sig,
  res.sfqt8.tm20.sig,
  res.sfqt8.m.sig,
  ntseq,
  npseq,
  mu,
  ES,
  sigma,
  tau,
  nsim,
  file=paste0('./data/sim_gp_tp2b.RData'))
beep(8)
```

Compute true positive probability

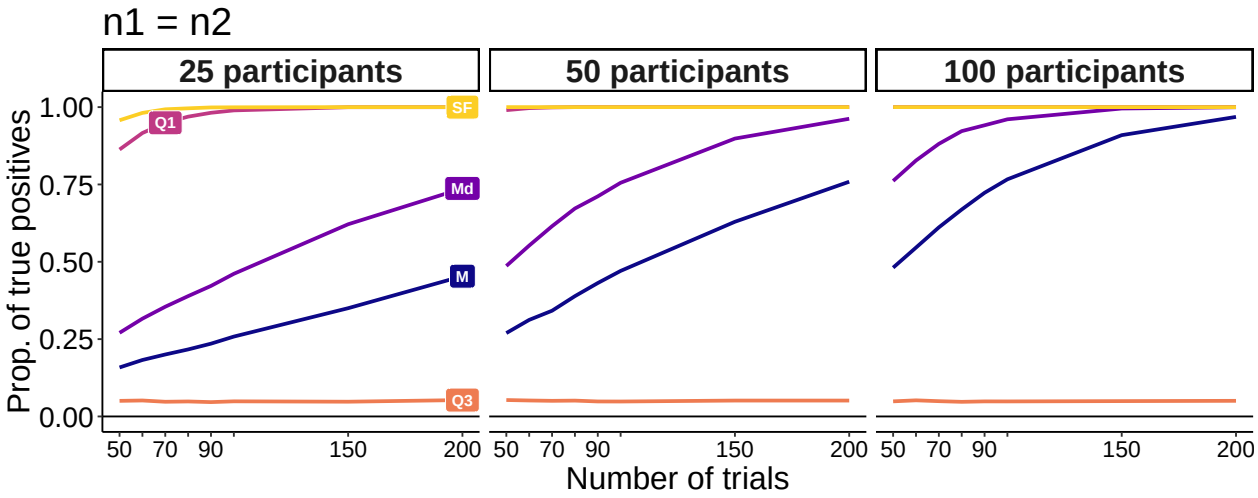
```
load('./data/sim_gp_tp2b.RData')
fp.m.m <- apply(res.m.m.sig, c(2,3), mean)
fp.md.md <- apply(res.md.md.sig, c(2,3), mean)
fp.25.md <- apply(res.25.md.sig, c(2,3), mean)
fp.75.md <- apply(res.75.md.sig, c(2,3), mean)
fp.sfhd.md <- apply(res.sfhd.md.sig > 0, c(2,3), mean)
fp.sfhd.m <- apply(res.sfhd.m.sig > 0, c(2,3), mean)
fp.sfhd.tm10 <- apply(res.sfhd.tm10.sig > 0, c(2,3), mean)
fp.sfhd.tm20 <- apply(res.sfhd.tm20.sig > 0, c(2,3), mean)
fp.sfqt8.md <- apply(res.sfqt8.md.sig > 0, c(2,3), mean)
fp.sfqt8.m <- apply(res.sfqt8.m.sig > 0, c(2,3), mean)
fp.sfqt8.tm10 <- apply(res.sfqt8.tm10.sig > 0, c(2,3), mean)
fp.sfqt8.tm20 <- apply(res.sfqt8.tm20.sig > 0, c(2,3), mean)
```

Illustrate results

M, Md, Q1, Q3, SF

```
# add labels to facet 1
lab_text <- tibble(FP = c(fp.m.m[length(ntseq), 1],
                          fp.md.md[length(ntseq), 1],
                          fp.25.md[3, 1],
```

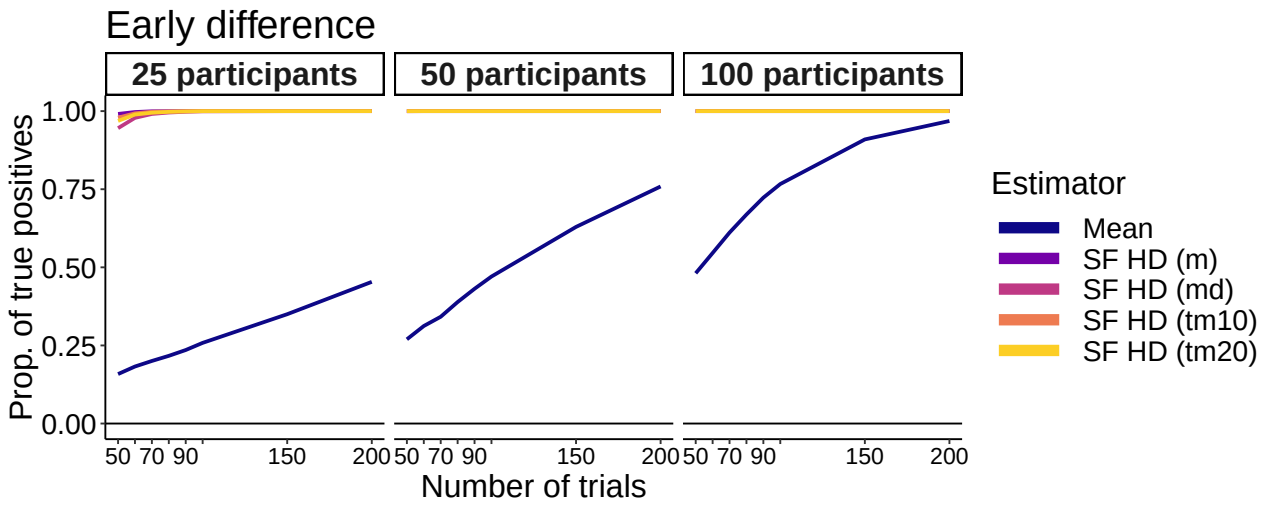
```
                        fp.75.md[length(ntseq), 1],
                        fp.sfqt8.tm20[length(ntseq), 1]),
                    Estimator = factor(c("Mean", "Median", "Q1", "Q3", "SF")),
                    lab = c("M", "Md", "Q1", "Q3", "SF"),
                    Trials = c(200, 200, 70, 200, 200),#rep(ntseq[length(ntseq)],5),
                    Participants = factor(25,levels = c("25","50","100")))

FP.in <- c(as.vector(fp.m.m),as.vector(fp.md.md),
        as.vector(fp.25.md),as.vector(fp.75.md),as.vector(fp.sfqt8.tm20))
Est.in <- c("Mean", "Median", "Q1", "Q3", "SF")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("n1 = n2")

# get colours to re-use in Tau plot + set label colours
g <- ggplot_build(p)
cm <- unique(g$data[[1]]["colour"]) # save colourmap

p <- p +  geom_label(data = lab_text,
                     aes(x = Trials, y = FP, label = lab, fill = Estimator),
                         colour = 'white', fontface = "bold", size = 3) +
          scale_fill_manual(values = cm$colour, guide = FALSE) +
          theme(legend.position = "none")
p
```




```
p.early.res <- p
```

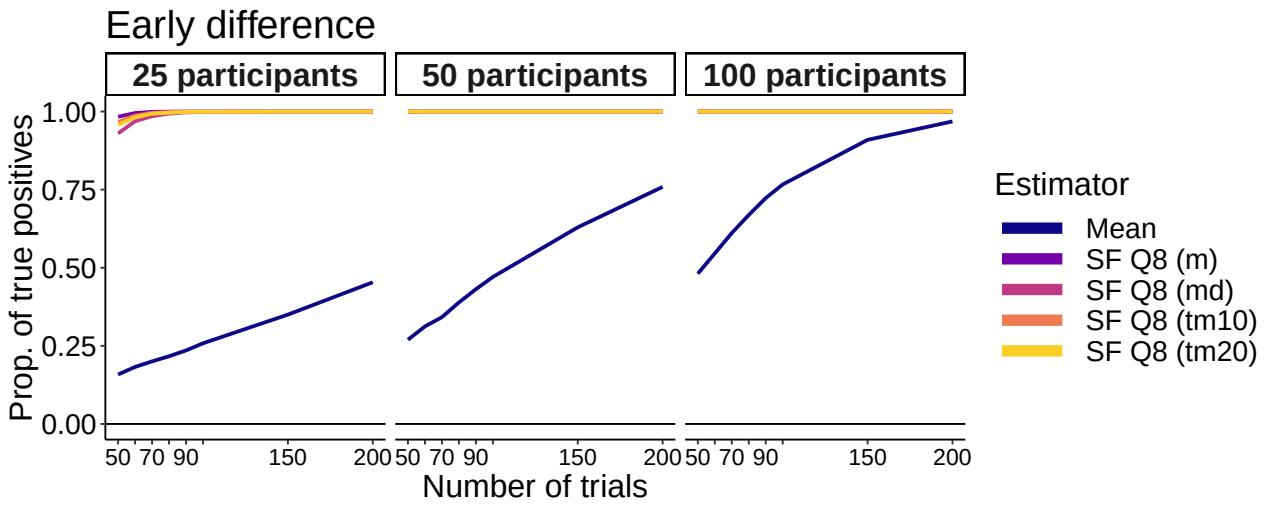
M, SF HD

```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20))
Est.in <- c("Mean","SF HD (m)", "SF HD (md)", "SF HD (tm10)","SF HD (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Early difference")
p
```



M, SF QT8

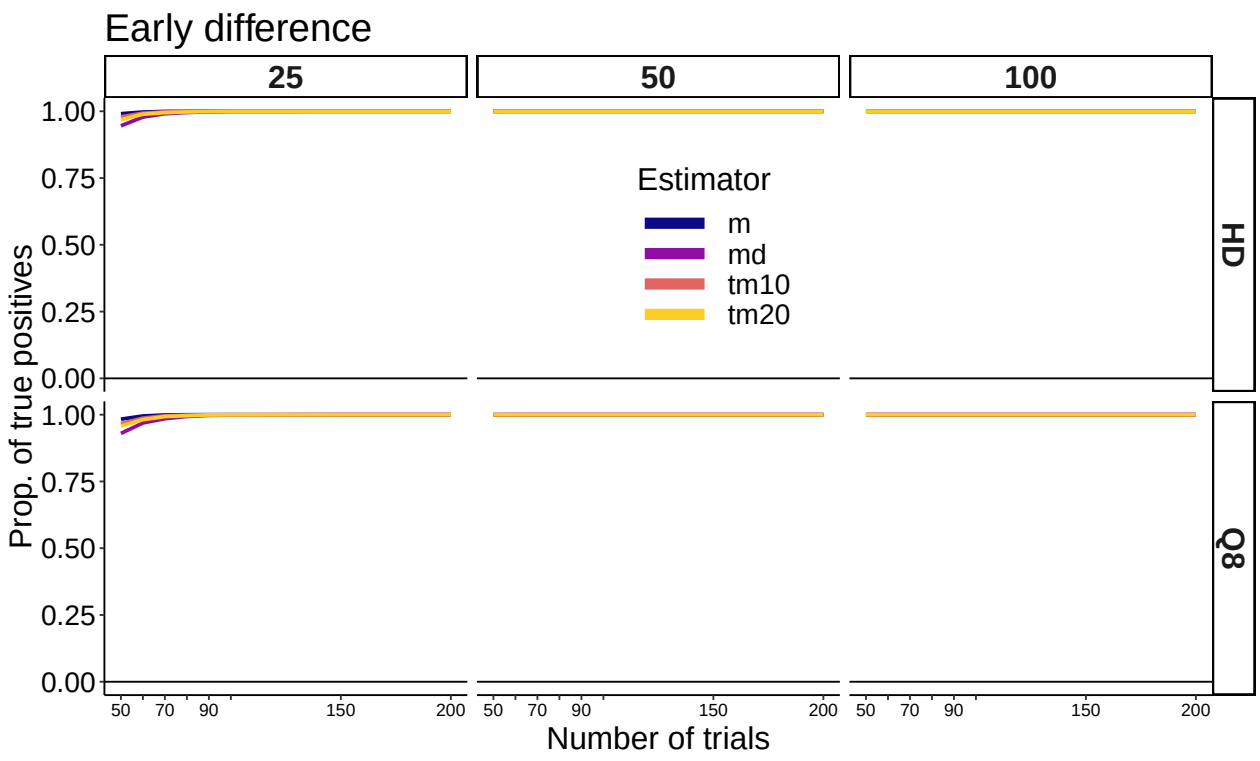
```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20))
Est.in <- c("Mean","SF Q8 (m)", "SF Q8 (md)", "SF Q8 (tm10)","SF Q8 (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Early difference")
p
```



Compare SF results

```
df <- tibble(FP = c(as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20)),
             Estimator = factor(rep(rep(c("m", "md", "tm10","tm20"),2),
               each=length(ntseq)*length(npseq))),
             Type = factor(rep(c("Q8","HD"), each = length(ntseq)*length(npseq)*4)),
             Trials = rep(ntseq,length(npseq)*8),
```

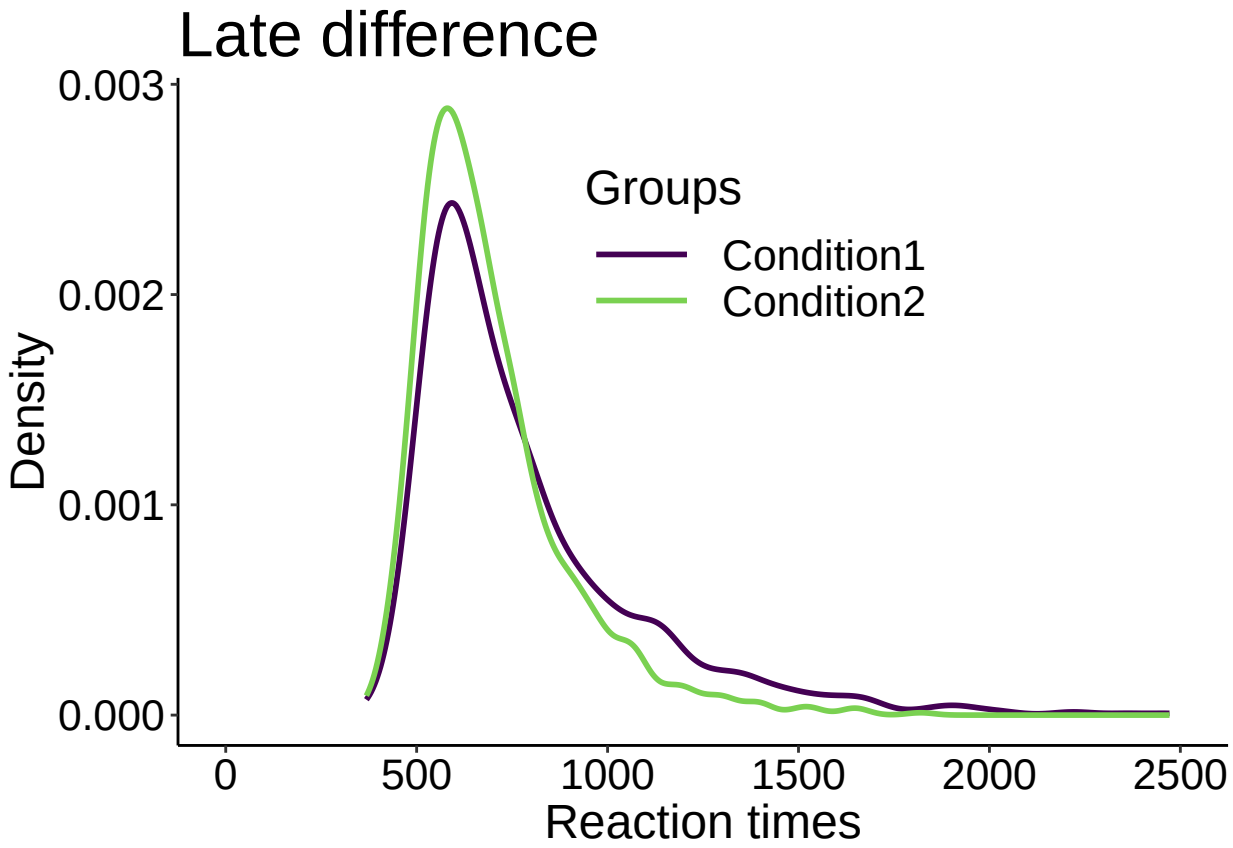
```
            Participants = factor(rep(rep(npseq,each=length(ntseq)),8))
            )
# make plot
p <- ggplot(df) + theme_classic() +
  geom_line(aes(x=Trials, y=FP, colour = Estimator), size = 1) +
  geom_abline(intercept=0, slope=0, colour="black") +
  scale_colour_viridis_d(option = "plasma", end=0.9) +
  scale_x_continuous(breaks = ntseq,
                     labels = c("50","","70","","90","","150","200")) +
  coord_cartesian(ylim=c(0,1)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 10, colour="black"),
        axis.text.y = element_text(size = 16, colour="black"),
        axis.title.y = element_text(size = 18),
        legend.key.width = unit(1.5,"cm"),
        legend.position = c(0.55,0.75),
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.text = element_text(size=18, face="bold"),
        strip.background = element_rect(colour="black", fill="white")) +
  labs(x = "Number of trials", y = "Prop. of true positives") +
  guides(colour = guide_legend(override.aes = list(size=3))) + # make thicker legend lines
  ggtitle("Early difference") +
  facet_grid(rows = vars(Type),
             cols = vars(Participants))
p
```



Group simulation 3: same number of trials in each group, vary Tau component

KDE

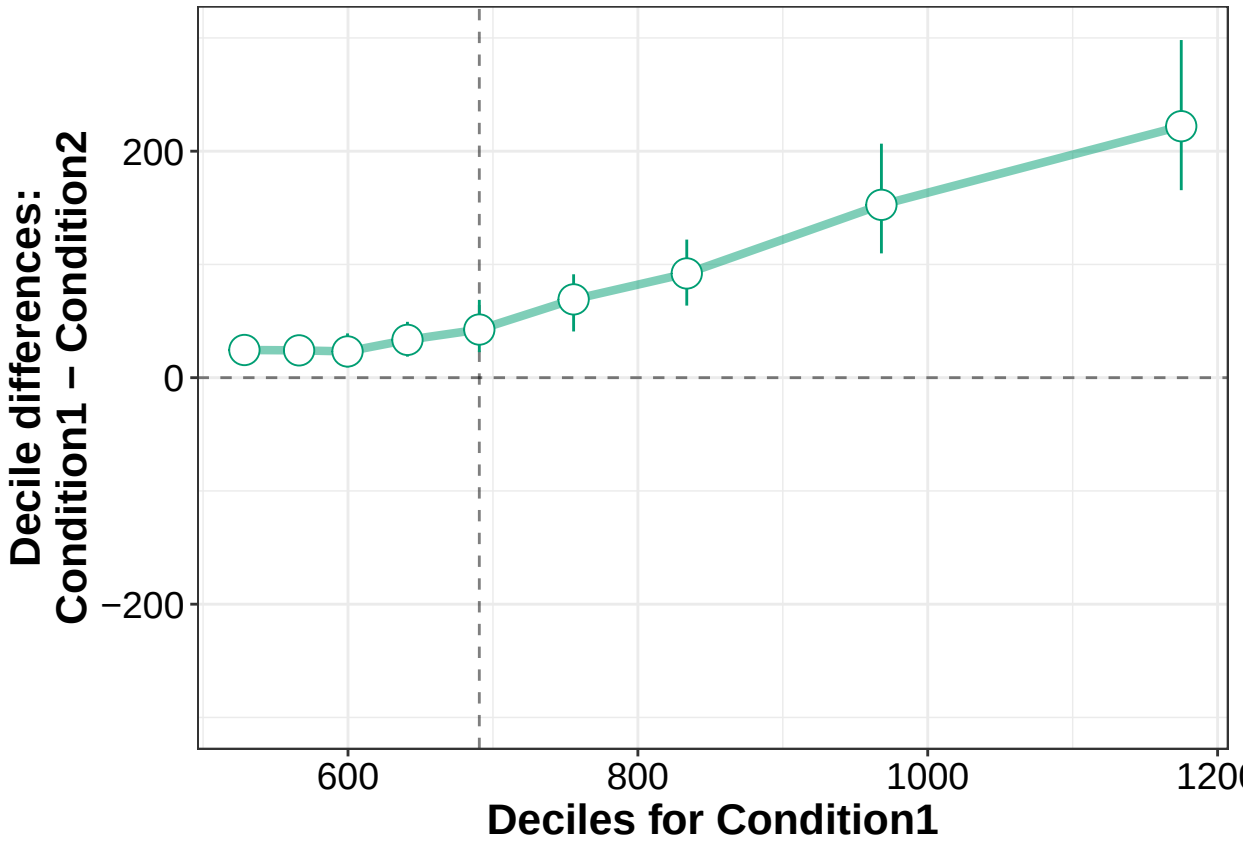
```
nt <- 1000
# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 100

set.seed(777)
g1 <- rexgauss(nt, mu = mu, sigma = sigma, tau = tau + ES)
g2 <- rexgauss(nt, mu = mu, sigma = sigma, tau = tau)
df <- mkt2(g1, g2, group_labels = c("Condition1", "Condition2"))

p <- ggplot(df, aes(x = obs)) + theme_classic() +
  stat_density(aes(colour = gr), geom="line",position="identity", size=1) +
  scale_colour_viridis_d(end = 0.8) +
  coord_cartesian(xlim = c(0, 2500)) +
  theme(axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 16, colour="black"),
        axis.text.y = element_text(size = 16, colour="black"),
        axis.title.y = element_text(size = 18),
        legend.key.width = unit(1.5,"cm"),
        legend.position = c(0.55,0.75),
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        title = element_text(size=20)) +
  labs(x = "Reaction times", y = "Density", colour = "Groups") +
  ggtitle("Late difference")
p
```




```
p.tau.dist <- p
```

Shift function

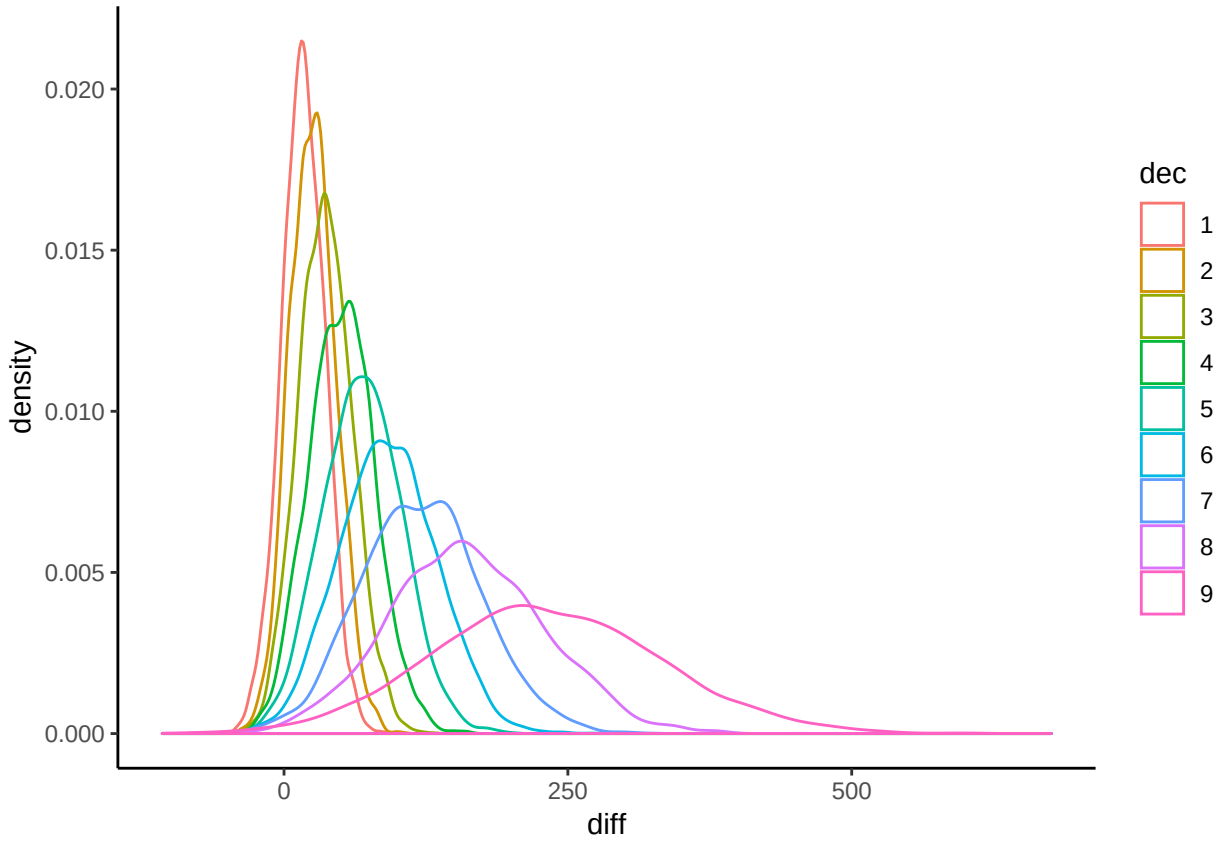
```
out <- shifthd_pbci(df, nboot = 200, adj_ci = FALSE)
p <- plot_sf(out, plot_theme = 1)[[1]] +
     theme(axis.text = element_text(size = 16, colour="black"))
p
```




```
p.tau.sf <- p
```

Decile sampling distributions

```
nt <- 100 # trials
np <- 5000 # participants
# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 100

mc.data1 <- array(rexgauss(nt*np, mu = mu, sigma = sigma, tau = tau + ES),
  dim = c(nt, np))
mc.data2 <- array(rexgauss(nt*np, mu = mu, sigma = sigma, tau = tau),
  dim = c(nt, np))

# array of differences: 9 deciles x participants
mc.hd <- apply(mc.data1, 2, hdseq) - apply(mc.data2, 2, hdseq)

df <- tibble(diff = as.vector(mc.hd),
             dec = factor(rep(seq(1,9), np)))
ggplot(df, aes(x = diff, colour = dec)) + theme_classic() +
  geom_density()
```



Skewness of sampling distributions

```
apply(mc.hd, 1, skew)
```

```
## [1] 0.02656926 0.12372885 0.13160041 0.12285270 0.09935977 0.11025065
## [7] 0.10154940 0.13198260 0.13416154
```

Kurtosis of sampling distributions

```
apply(mc.hd, 1, kurt)
```

```
## [1] 3.051358 2.968536 3.003442 3.011747 2.991509 3.016459 2.995255 3.075438
## [9] 3.077020
```

Simulation

```
npseq <- c(25, 50, 100)
ntseq <- c(seq(50, 100, 10), 150, 200)
maxnp <- max(npseq) # max number of participants
maxnt <- max(ntseq) # max number of trials
nsim <- 10000 # simulation samples
qseq <- seq(0.1,0.9,0.1) # define deciles for quantile function
```

```r
# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 15

alpha <- 0.05

# declare matrices of results - save all iterations
res.m.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.md.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.25.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.75.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfhd.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfqt8.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

pvals.0.hd <- vector(mode = "numeric", length = 9)
pvals.10.hd <- vector(mode = "numeric", length = 9)
pvals.20.hd <- vector(mode = "numeric", length = 9)

pvals.0.qt8 <- vector(mode = "numeric", length = 9)
pvals.10.qt8 <- vector(mode = "numeric", length = 9)
pvals.20.qt8 <- vector(mode = "numeric", length = 9)

# create 2 populations
set.seed(21)
npop <- 1000000
pop1 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau + ES)
pop2 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau)

beep(3)

# simulation samples
mc.data1 <- array(sample(pop1, maxnt*maxnp*nsim, replace = TRUE), dim = c(maxnt, maxnp, nsim))
mc.data2 <- array(sample(pop2, maxnt*maxnp*nsim, replace = TRUE), dim = c(maxnt, maxnp, nsim))

for(TR in 1:length(ntseq)){ # number of trials
  print(paste0("number of trials: ",ntseq[TR],"..."))
  beep(2)

  for(PT in 1:length(npseq)){ # number of participants
    print(paste0("number of participants: ",npseq[PT],"..."))

    # compute estimates
    todo1 <- mc.data1[1:ntseq[TR], 1:npseq[PT],]
```

```
    todo2 <- mc.data2[1:ntseq[TR], 1:npseq[PT],]
    mc.m <- apply(todo1, c(2,3), mean) - apply(todo2, c(2,3), mean) # diff in mean RTs
    mc.md <- apply(todo1, c(2,3), median) - apply(todo2, c(2,3), median) # diff in median RTs
    mc.25 <- apply(todo1, c(2,3), quantile, probs = .25) - apply(todo2, c(2,3), quantile, probs = .25)
    mc.75 <- apply(todo1, c(2,3), quantile, probs = .75) - apply(todo2, c(2,3), quantile, probs = .75)

    # tests for each simulation ====================
    # one-sample t-test
    res.m.m.sig[, TR, PT] <- ttest.sig(mc.m)
    # parametric median test
    res.md.md.sig[, TR, PT] <- apply(mc.md, 2, sint.sig)
    res.25.md.sig[, TR, PT] <- apply(mc.25, 2, sint.sig)
    res.75.md.sig[, TR, PT] <- apply(mc.75, 2, sint.sig)

    # shift function
    # array of differences: 9 deciles x participants x simulations
    mc.hd <- apply(todo1, c(2,3), hdseq) - apply(todo2, c(2,3), hdseq)
    mc.qt8 <- apply(todo1, c(2,3), quantile, probs = qseq, type = 8) -
              apply(todo2, c(2,3), quantile, probs = qseq, type = 8)
    # median test for each decile distribution
    pval.hd <- apply(mc.hd, c(1,3), sintv2.pval)
    pval.qt8 <- apply(mc.qt8, c(1,3), sintv2.pval)
    # trimmed mean tests for each decile distribution
    tvals.0.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0)
    tvals.10.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.1)
    tvals.20.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.2)
    tvals.0.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0)
    tvals.10.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.1)
    tvals.20.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.2)
    df0 <- df.yuen(mc.hd[1,,1], tr = 0)
    df10 <- df.yuen(mc.hd[1,,1], tr = 0.1)
    df20 <- df.yuen(mc.hd[1,,1], tr = 0.2)
    for(S in 1:nsim){
      res.sfhd.md.sig[S, TR, PT] <- sum(p.adjust(pval.hd[,S], method = "hochberg") <= alpha)
      res.sfqt8.md.sig[S, TR, PT] <- sum(p.adjust(pval.qt8[,S], method = "hochberg") <= alpha)

      for(Q in 1:9){
        pvals.0.hd[Q] <- cpval(tvals.0.hd[S,Q], df0)
        pvals.10.hd[Q] <- cpval(tvals.10.hd[S,Q], df10)
        pvals.20.hd[Q] <- cpval(tvals.20.hd[S,Q], df20)
        pvals.0.qt8[Q] <- cpval(tvals.0.qt8[S,Q], df0)
        pvals.10.qt8[Q] <- cpval(tvals.10.qt8[S,Q], df10)
        pvals.20.qt8[Q] <- cpval(tvals.20.qt8[S,Q], df20)
      }
      res.sfhd.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.hd, method = "hochberg") <= alpha)
      res.sfhd.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.hd, method = "hochberg") <= alpha)
      res.sfhd.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.hd, method = "hochberg") <= alpha)
      res.sfqt8.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.qt8, method = "hochberg") <= alpha)
      res.sfqt8.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.qt8, method = "hochberg") <= alpha)
      res.sfqt8.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.qt8, method = "hochberg") <= alpha)
    }
  }
}
```

```
save(
  res.m.m.sig,
  res.md.md.sig,
  res.25.md.sig,
  res.75.md.sig,
  res.sfhd.md.sig,
  res.sfhd.tm10.sig,
  res.sfhd.tm20.sig,
  res.sfhd.m.sig,
  res.sfqt8.md.sig,
  res.sfqt8.tm10.sig,
  res.sfqt8.tm20.sig,
  res.sfqt8.m.sig,
  ntseq,
  npseq,
  mu,
  ES,
  sigma,
  tau,
  nsim,
  file=paste0('./data/sim_gp_tp3.RData'))
beep(8)
```

Simulation: estimate tau

```
npseq <- c(25, 50, 100)
ntseq <- c(seq(50, 100, 10), 150, 200)
maxnp <- max(npseq) # max number of participants
maxnt <- max(ntseq) # max number of trials
nsim <- 10000 # simulation samples

# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 15

alpha <- 0.05

# declare matrices of results - save all iterations
# res.mu.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
# res.sigma.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.tau.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

# create 2 populations
set.seed(21)
npop <- 1000000
pop1 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau + ES)
pop2 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau)

beep(3)
```

```
# simulation samples
mc.data1 <- array(sample(pop1, maxnt*maxnp*nsim, replace = TRUE), dim = c(maxnt, maxnp, nsim))
mc.data2 <- array(sample(pop2, maxnt*maxnp*nsim, replace = TRUE), dim = c(maxnt, maxnp, nsim))

for(TR in 1:length(ntseq)){ # number of trials
  print(paste0("number of trials: ",ntseq[TR],"..."))
  beep(2)
  tau_diff <- matrix(NA, nrow = max(npseq), ncol = nsim)
  # Fit Ex-Gaussians
    tofit1 <- mc.data1[1:ntseq[TR],,]
    tofit2 <- mc.data2[1:ntseq[TR],,]

    for(iter.s in 1:nsim){
      for(iter.p in 1:max(npseq)){
       out1 <- timefit(tofit1[,iter.p,iter.s])
       out2 <- timefit(tofit2[,iter.p,iter.s])
       tau_diff[iter.p,iter.s] <- out1@par[3] - out2@par[3]
      }
    }

  for(PT in 1:length(npseq)){ # number of participants
    # one-sample t-test
    res.tau.m.sig[, TR, PT] <- ttest.sig(tau_diff[1:npseq[PT],])
  }
}

save(
  res.tau.m.sig,
  ntseq,
  npseq,
  mu,
  ES,
  sigma,
  tau,
  nsim,
  file=paste0('./data/sim_gp_tp3_exgfit.RData'))
beep(8)
```

Compute true positive probability

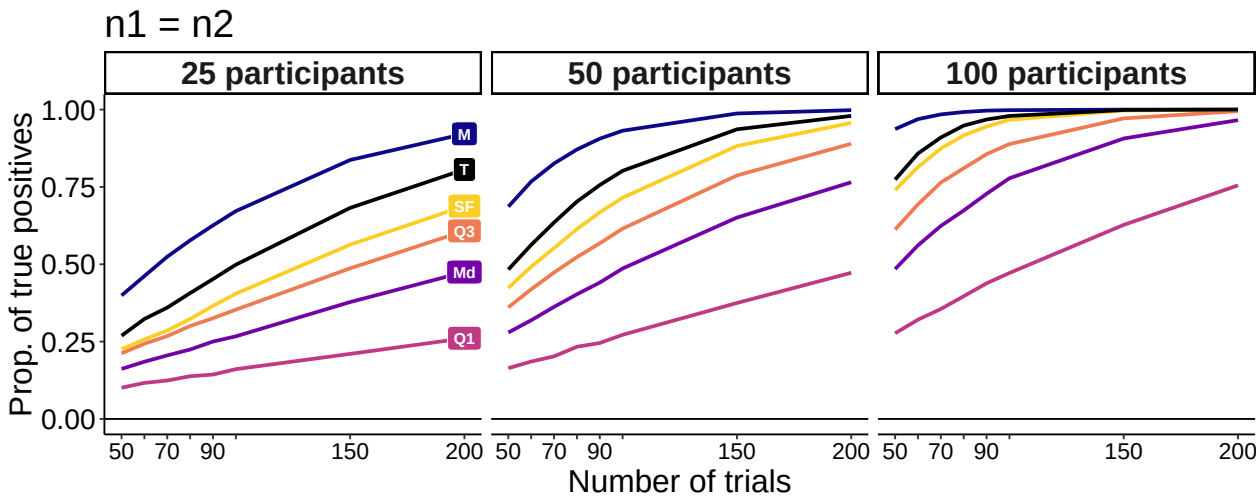
```
load('./data/sim_gp_tp3.RData')
fp.m.m <- apply(res.m.m.sig, c(2,3), mean)
fp.md.md <- apply(res.md.md.sig, c(2,3), mean)
fp.25.md <- apply(res.25.md.sig, c(2,3), mean)
fp.75.md <- apply(res.75.md.sig, c(2,3), mean)
fp.sfhd.md <- apply(res.sfhd.md.sig > 0, c(2,3), mean)
fp.sfhd.m <- apply(res.sfhd.m.sig > 0, c(2,3), mean)
fp.sfhd.tm10 <- apply(res.sfhd.tm10.sig > 0, c(2,3), mean)
fp.sfhd.tm20 <- apply(res.sfhd.tm20.sig > 0, c(2,3), mean)
fp.sfqt8.md <- apply(res.sfqt8.md.sig > 0, c(2,3), mean)
fp.sfqt8.m <- apply(res.sfqt8.m.sig > 0, c(2,3), mean)
```

```
fp.sfqt8.tm10 <- apply(res.sfqt8.tm10.sig > 0, c(2,3), mean)
fp.sfqt8.tm20 <- apply(res.sfqt8.tm20.sig > 0, c(2,3), mean)
load('./data/sim_gp_tp3_exgfit.RData')
fp.tau.m <- apply(res.tau.m.sig, c(2,3), mean)
```

Illustrate results

M, Md, Q1, Q3, SF, tau

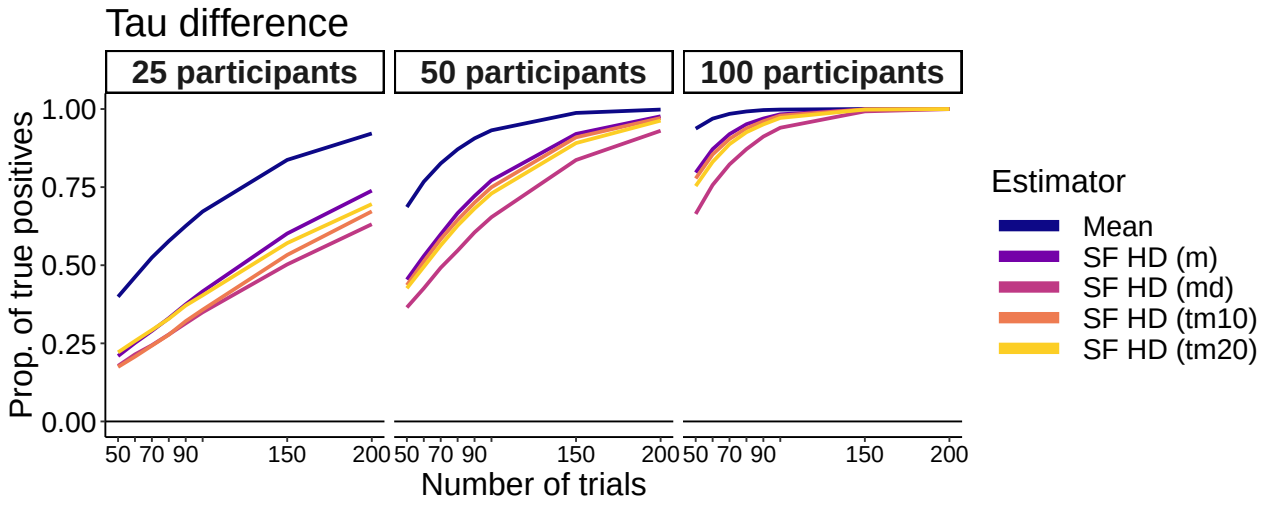
```
# add labels to facet 1
lab_text <- tibble(FP = c(fp.m.m[length(ntseq), 1],
                          fp.md.md[length(ntseq), 1],
                          fp.25.md[length(ntseq), 1],
                          fp.75.md[length(ntseq), 1],
                          fp.sfqt8.tm20[length(ntseq), 1],
                          fp.tau.m[length(ntseq), 1]),
                   Estimator = factor(c("Mean", "Median", "Q1", "Q3", "SF", "Tau")),
                   lab = c("M", "Md", "Q1", "Q3", "SF", "T"),
                   Trials = rep(ntseq[length(ntseq)],6),
                   Participants = factor(25,levels = c("25","50","100")))

FP.in <- c(as.vector(fp.m.m), as.vector(fp.md.md), as.vector(fp.25.md),
           as.vector(fp.75.md), as.vector(fp.sfqt8.tm20), as.vector(fp.tau.m))
Est.in <- c("Mean", "Median", "Q1", "Q3", "SF", "Tau")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("n1 = n2") +
           scale_colour_manual(values = c(cm$colour, "black"))  +
           theme(legend.position = "none")

p <- p +  geom_label(data = lab_text,
                     aes(x = Trials, y = FP, label = lab, fill = Estimator),
                         colour = 'white', fontface = "bold", size = 3) +
          scale_fill_manual(values = c(cm$colour, "black"), guide = FALSE)
p
```




```
p.tau.res <- p
```

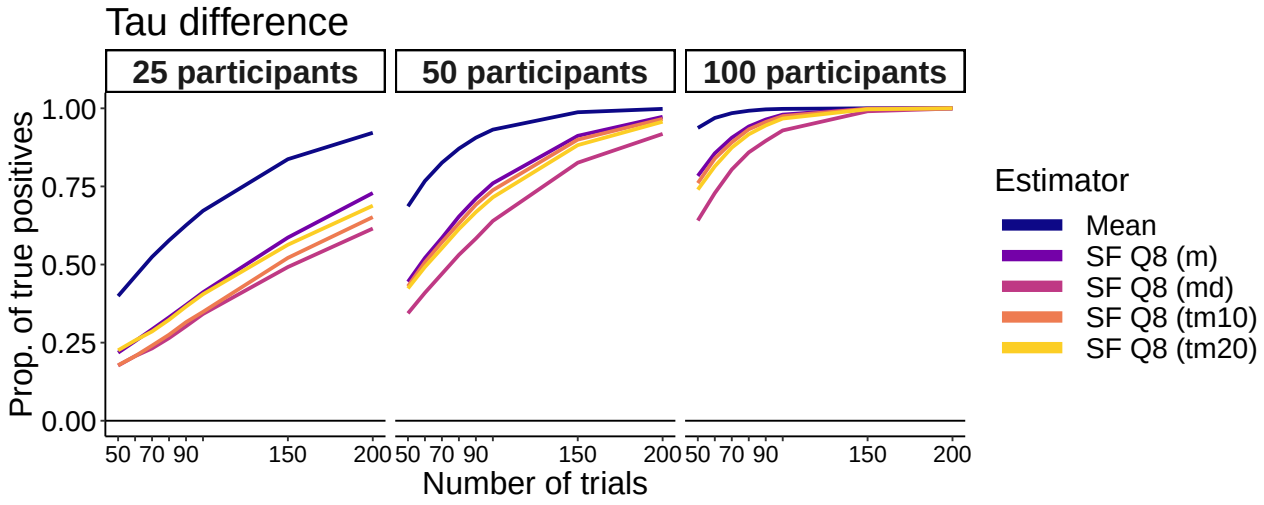
M, SF HD

```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20))
Est.in <- c("Mean","SF HD (m)", "SF HD (md)", "SF HD (tm10)","SF HD (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Tau difference")
p
```



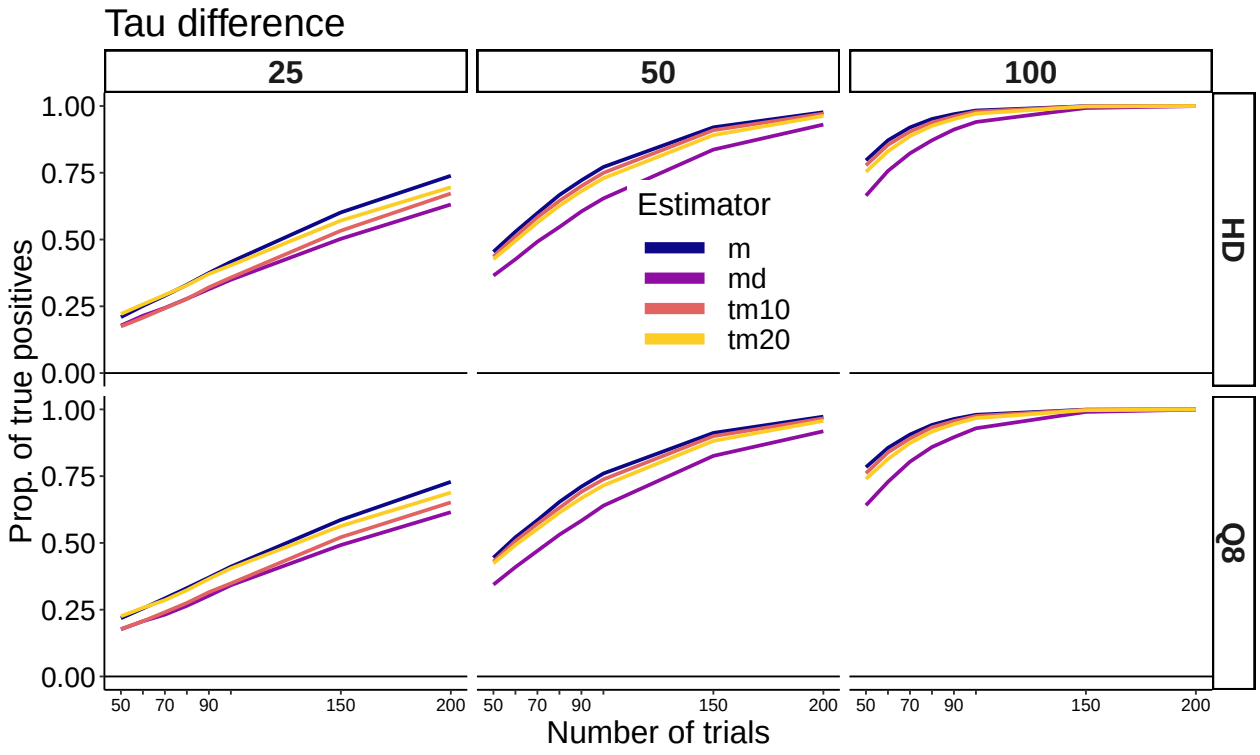
M, SF QT8

```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20))
Est.in <- c("Mean","SF Q8 (m)", "SF Q8 (md)", "SF Q8 (tm10)","SF Q8 (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Tau difference")
p
```



Compare SF results

```
df <- tibble(FP = c(as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20)),
             Estimator = factor(rep(rep(c("m", "md", "tm10","tm20"),2),
               each=length(ntseq)*length(npseq))),
             Type = factor(rep(c("Q8","HD"), each = length(ntseq)*length(npseq)*4)),
             Trials = rep(ntseq,length(npseq)*8),
             Participants = factor(rep(rep(npseq,each=length(ntseq)),8))
             )
# make plot
p <- ggplot(df) + theme_classic() +
  geom_line(aes(x=Trials, y=FP, colour = Estimator), size = 1) +
  geom_abline(intercept=0, slope=0, colour="black") +
  scale_colour_viridis_d(option = "plasma", end=0.9) +
  scale_x_continuous(breaks = ntseq,
                     labels = c("50","","70","","90","","150","200")) +
  coord_cartesian(ylim=c(0,1)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 10, colour="black"),
        axis.text.y = element_text(size = 16, colour="black"),
        axis.title.y = element_text(size = 18),
        legend.key.width = unit(1.5,"cm"),
        legend.position = c(0.55,0.70),
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.text = element_text(size=18, face="bold"),
        strip.background = element_rect(colour="black", fill="white")) +
  labs(x = "Number of trials", y = "Prop. of true positives") +
  guides(colour = guide_legend(override.aes = list(size=3))) + # make thicker legend lines
  ggtitle("Tau difference") +
  facet_grid(rows = vars(Type),
             cols = vars(Participants))
p
```



Summary figure

```
# combine panels into one figure
cowplot::plot_grid(p.uni.res, p.spr.res, p.early.res, p.tau.res,
                           labels = c("A", "B", "C", "D"),
                           ncol = 1,
                           nrow = 4,
                           # rel_widths = c(1, 1, 1),
                           label_size = 20,
                           hjust = -0.5,
                           scale=.95,
                           align = "h")
# save figure
ggsave(filename=('./figures/figure_sim_gp_tp1_summary.pdf'),width=10,height=15)
```

Group simulation 4 (2.1): uniform shift, one group = always 200 trials, other group n trials

Simulation

```
npseq <- c(25, 50, 100)
ntseq1 <- c(seq(50, 100, 10), 150, 200)
ntseq2 <- rep(200, length(ntseq1))
maxnp <- max(npseq) # max number of participants
maxnt1 <- max(ntseq1) # max number of trials
```

```r
maxnt2 <- max(ntseq2) # max number of trials
nsim <- 10000 # simulation samples
qseq <- seq(0.1,0.9,0.1) # define deciles for quantile function
ntseq <- ntseq1

# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 10

alpha <- 0.05

# declare matrices of results - save all iterations
res.m.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.md.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.25.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.75.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfhd.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfqt8.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

pvals.0.hd <- vector(mode = "numeric", length = 9)
pvals.10.hd <- vector(mode = "numeric", length = 9)
pvals.20.hd <- vector(mode = "numeric", length = 9)

pvals.0.qt8 <- vector(mode = "numeric", length = 9)
pvals.10.qt8 <- vector(mode = "numeric", length = 9)
pvals.20.qt8 <- vector(mode = "numeric", length = 9)

# create 2 populations
set.seed(21)
npop <- 1000000
pop2 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau)
pop1 <- pop2 + ES

beep(3)

# simulation samples
mc.data1 <- array(sample(pop1, maxnt1*maxnp*nsim, replace = TRUE), dim = c(maxnt1, maxnp, nsim))
mc.data2 <- array(sample(pop2, maxnt2*maxnp*nsim, replace = TRUE), dim = c(maxnt2, maxnp, nsim))

for(TR in 1:length(ntseq)){ # number of trials
  print(paste0("number of trials: ",ntseq[TR],"..."))
  beep(2)
```

```r
for(PT in 1:length(npseq)){ # number of participants
  print(paste0("number of participants: ",npseq[PT],"..."))

  # compute estimates
  todo1 <- mc.data1[1:ntseq1[TR], 1:npseq[PT],]
  todo2 <- mc.data2[1:ntseq2[TR], 1:npseq[PT],]
  mc.m <- apply(todo1, c(2,3), mean) - apply(todo2, c(2,3), mean) # diff in mean RTs
  mc.md <- apply(todo1, c(2,3), median) - apply(todo2, c(2,3), median) # diff in median RTs
  mc.25 <- apply(todo1, c(2,3), quantile, probs = .25) - apply(todo2, c(2,3), quantile, probs = .25)
  mc.75 <- apply(todo1, c(2,3), quantile, probs = .75) - apply(todo2, c(2,3), quantile, probs = .75)

  # tests for each simulation ====================
  # one-sample t-test
  res.m.m.sig[, TR, PT] <- ttest.sig(mc.m)
  # parametric median test
  res.md.md.sig[, TR, PT] <- apply(mc.md, 2, sint.sig)
  res.25.md.sig[, TR, PT] <- apply(mc.25, 2, sint.sig)
  res.75.md.sig[, TR, PT] <- apply(mc.75, 2, sint.sig)

  # shift function
  # array of differences: 9 deciles x participants x simulations
  mc.hd <- apply(todo1, c(2,3), hdseq) - apply(todo2, c(2,3), hdseq)
  mc.qt8 <- apply(todo1, c(2,3), quantile, probs = qseq, type = 8) -
            apply(todo2, c(2,3), quantile, probs = qseq, type = 8)
  # median test for each decile distribution
  pval.hd <- apply(mc.hd, c(1,3), sintv2.pval)
  pval.qt8 <- apply(mc.qt8, c(1,3), sintv2.pval)
  # trimmed mean tests for each decile distribution
  tvals.0.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0)
  tvals.10.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.1)
  tvals.20.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.2)
  tvals.0.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0)
  tvals.10.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.1)
  tvals.20.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.2)
  df0 <- df.yuen(mc.hd[1,,1], tr = 0)
  df10 <- df.yuen(mc.hd[1,,1], tr = 0.1)
  df20 <- df.yuen(mc.hd[1,,1], tr = 0.2)
  for(S in 1:nsim){
    res.sfhd.md.sig[S, TR, PT] <- sum(p.adjust(pval.hd[,S], method = "hochberg") <= alpha)
    res.sfqt8.md.sig[S, TR, PT] <- sum(p.adjust(pval.qt8[,S], method = "hochberg") <= alpha)

    for(Q in 1:9){
      pvals.0.hd[Q] <- cpval(tvals.0.hd[S,Q], df0)
      pvals.10.hd[Q] <- cpval(tvals.10.hd[S,Q], df10)
      pvals.20.hd[Q] <- cpval(tvals.20.hd[S,Q], df20)
      pvals.0.qt8[Q] <- cpval(tvals.0.qt8[S,Q], df0)
      pvals.10.qt8[Q] <- cpval(tvals.10.qt8[S,Q], df10)
      pvals.20.qt8[Q] <- cpval(tvals.20.qt8[S,Q], df20)
    }
    res.sfhd.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.hd, method = "hochberg") <= alpha)
    res.sfhd.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.hd, method = "hochberg") <= alpha)
    res.sfhd.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.hd, method = "hochberg") <= alpha)
    res.sfqt8.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.qt8, method = "hochberg") <= alpha)
```

```
      res.sfqt8.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.qt8, method = "hochberg") <= alpha)
      res.sfqt8.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.qt8, method = "hochberg") <= alpha)
    }
  }
}

save(
  res.m.m.sig,
  res.md.md.sig,
  res.25.md.sig,
  res.75.md.sig,
  res.sfhd.md.sig,
  res.sfhd.tm10.sig,
  res.sfhd.tm20.sig,
  res.sfhd.m.sig,
  res.sfqt8.md.sig,
  res.sfqt8.tm10.sig,
  res.sfqt8.tm20.sig,
  res.sfqt8.m.sig,
  ntseq1,
  ntseq2,
  ntseq,
  npseq,
  mu,
  ES,
  sigma,
  tau,
  nsim,
  file=paste0('./data/sim_gp_tp4.RData'))
beep(8)
```

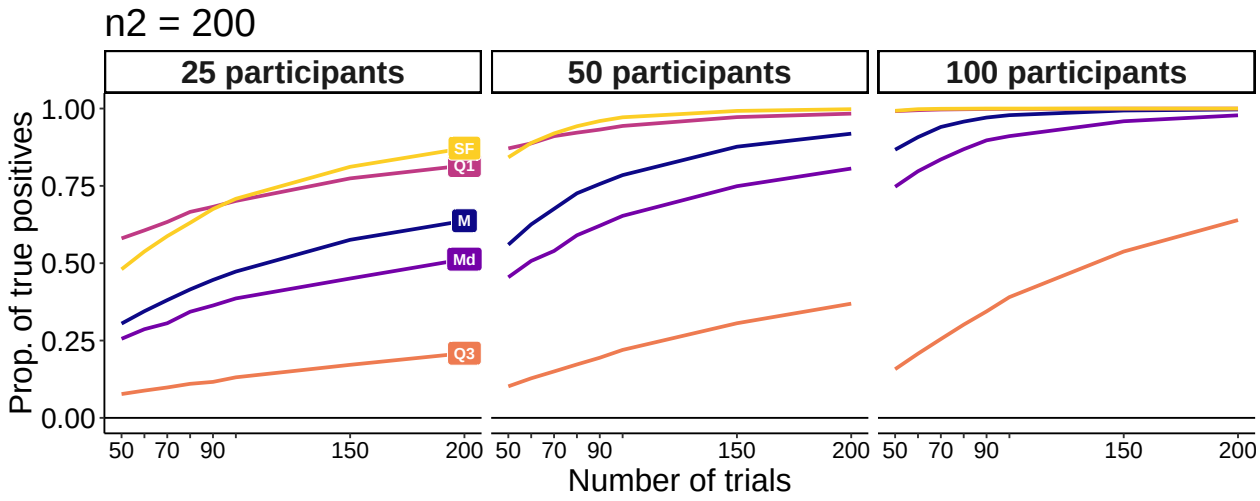
Compute true positive probability

```
load('./data/sim_gp_tp4.RData')
fp.m.m <- apply(res.m.m.sig, c(2,3), mean)
fp.md.md <- apply(res.md.md.sig, c(2,3), mean)
fp.25.md <- apply(res.25.md.sig, c(2,3), mean)
fp.75.md <- apply(res.75.md.sig, c(2,3), mean)
fp.sfhd.md <- apply(res.sfhd.md.sig > 0, c(2,3), mean)
fp.sfhd.m <- apply(res.sfhd.m.sig > 0, c(2,3), mean)
fp.sfhd.tm10 <- apply(res.sfhd.tm10.sig > 0, c(2,3), mean)
fp.sfhd.tm20 <- apply(res.sfhd.tm20.sig > 0, c(2,3), mean)
fp.sfqt8.md <- apply(res.sfqt8.md.sig > 0, c(2,3), mean)
fp.sfqt8.m <- apply(res.sfqt8.m.sig > 0, c(2,3), mean)
fp.sfqt8.tm10 <- apply(res.sfqt8.tm10.sig > 0, c(2,3), mean)
fp.sfqt8.tm20 <- apply(res.sfqt8.tm20.sig > 0, c(2,3), mean)
```

Illustrate results

M, Md, Q1, Q3, SF

```
# add labels to facet 1
lab_text <- tibble(FP = c(fp.m.m[length(ntseq), 1],
                          fp.md.md[length(ntseq), 1],
                          fp.25.md[length(ntseq), 1],
                          fp.75.md[length(ntseq), 1],
                          fp.sfqt8.tm20[length(ntseq), 1]),
                   Estimator = factor(c("Mean", "Median", "Q1", "Q3", "SF")),
                   lab = c("M", "Md", "Q1", "Q3", "SF"),
                   Trials = rep(ntseq[length(ntseq)],5),
                   Participants = factor(25,levels = c("25","50","100")))

FP.in <- c(as.vector(fp.m.m),as.vector(fp.md.md),
        as.vector(fp.25.md),as.vector(fp.75.md),as.vector(fp.sfqt8.tm20))
Est.in <- c("Mean", "Median", "Q1", "Q3", "SF")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("n2 = 200") +
      theme(legend.position = "none")

p <- p +  geom_label(data = lab_text,
                     aes(x = Trials, y = FP, label = lab, fill = Estimator),
                         colour = 'white', fontface = "bold", size = 3) +
          scale_fill_manual(values = cm$colour, guide = FALSE)
p
```

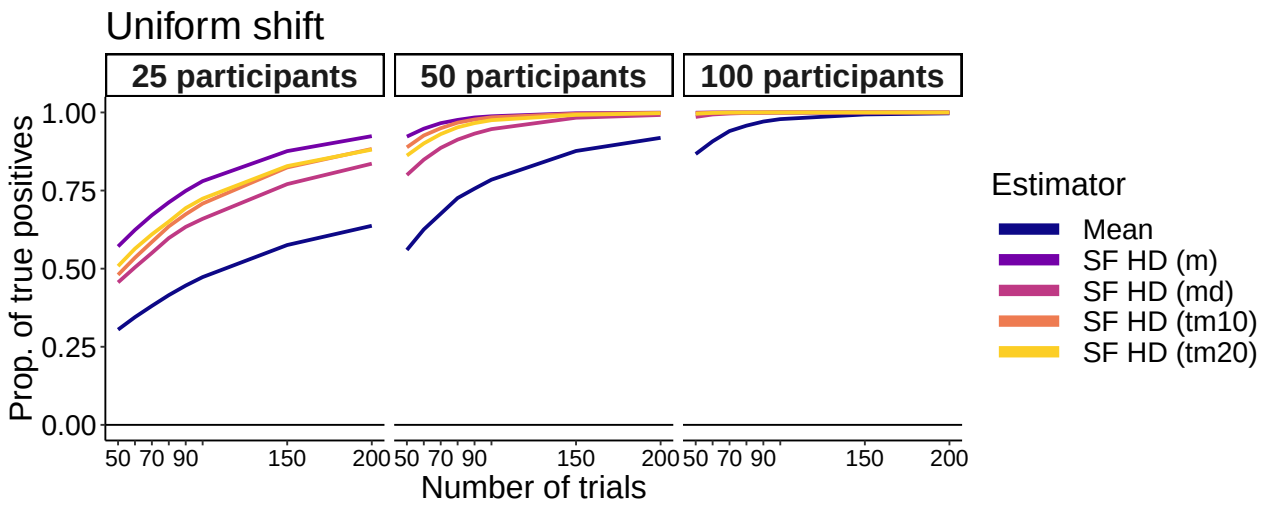



```
p.uni.res2 <- p
```

M, SF HD

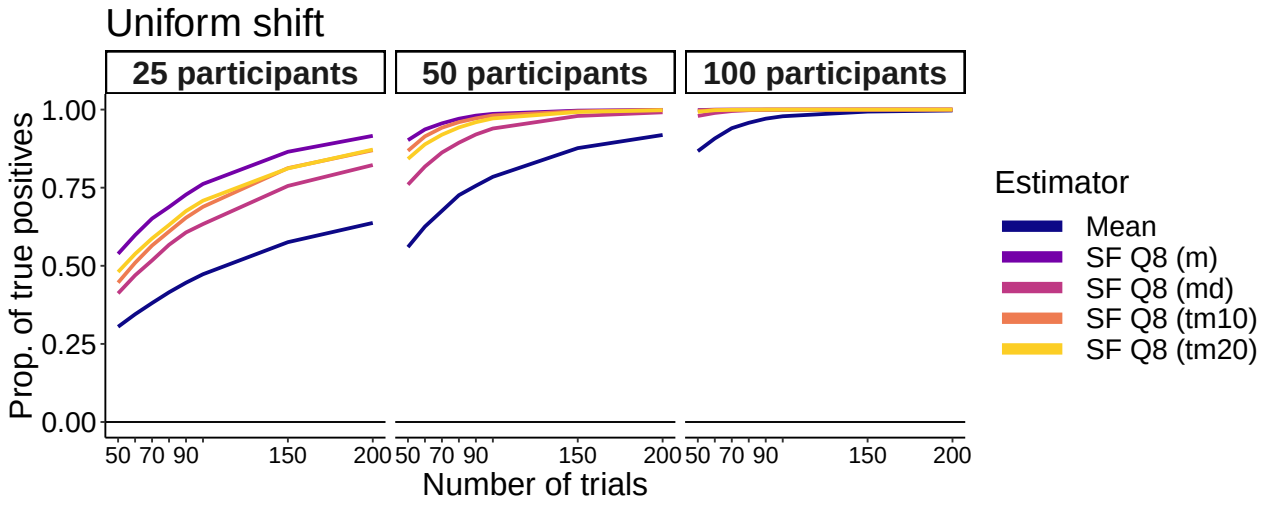
```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20))
Est.in <- c("Mean","SF HD (m)", "SF HD (md)", "SF HD (tm10)","SF HD (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Uniform shift")
```

```
p
```



M, SF QT8

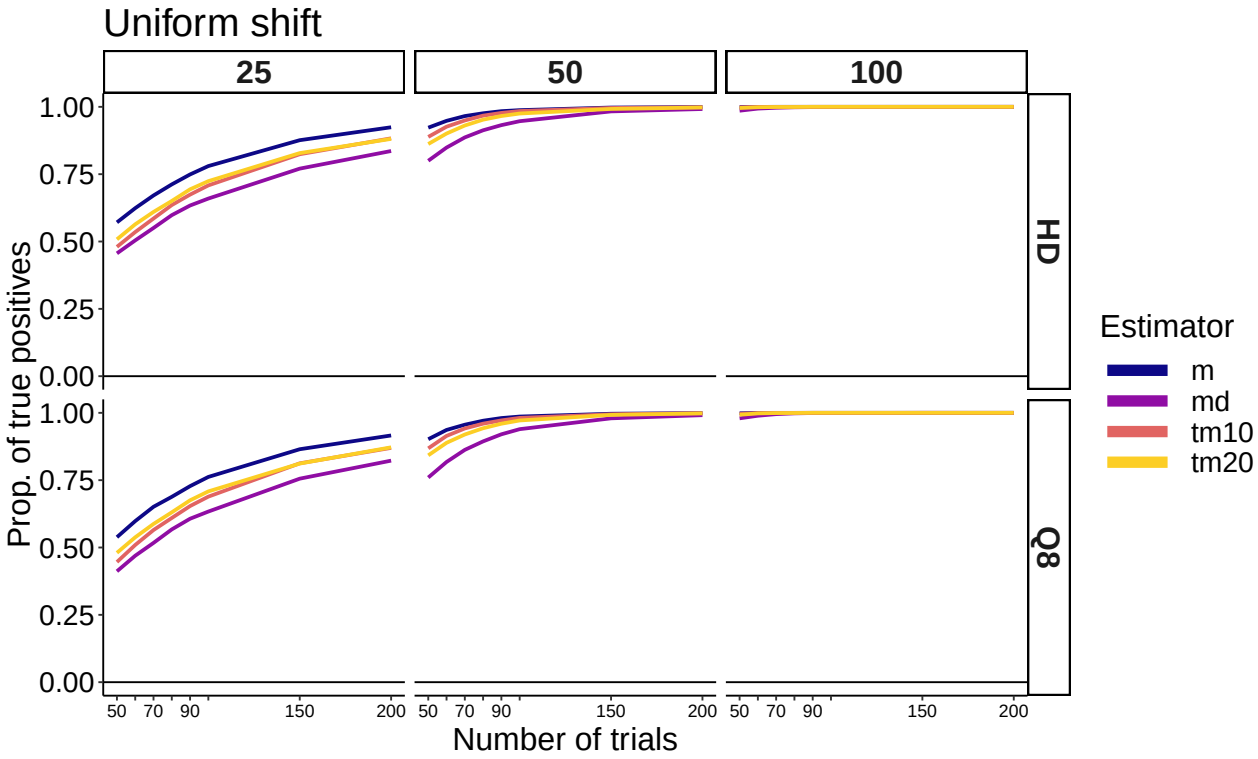
```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20))
Est.in <- c("Mean","SF Q8 (m)", "SF Q8 (md)", "SF Q8 (tm10)","SF Q8 (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Uniform shift")
p
```



Compare SF results

```
df <- tibble(FP = c(as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20)),
             Estimator = factor(rep(rep(c("m", "md", "tm10","tm20"),2),
               each=length(ntseq)*length(npseq))),
```

```
            Type = factor(rep(c("Q8","HD"), each = length(ntseq)*length(npseq)*4)),
            Trials = rep(ntseq,length(npseq)*8),
            Participants = factor(rep(rep(npseq,each=length(ntseq)),8))
            )
# make plot
p <- ggplot(df) + theme_classic() +
  geom_line(aes(x=Trials, y=FP, colour = Estimator), size = 1) +
  geom_abline(intercept=0, slope=0, colour="black") +
  scale_colour_viridis_d(option = "plasma", end=0.9) +
  scale_x_continuous(breaks = ntseq,
                     labels = c("50","","70","","90","","150","200")) +
  coord_cartesian(ylim=c(0,1)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 10, colour="black"),
        axis.text.y = element_text(size = 16, colour="black"),
        axis.title.y = element_text(size = 18),
        legend.key.width = unit(1.5,"cm"),
        legend.position = "right",#c(0.55,0.85),
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.text = element_text(size=18, face="bold"),
        strip.background = element_rect(colour="black", fill="white")) +
  labs(x = "Number of trials", y = "Prop. of true positives") +
  guides(colour = guide_legend(override.aes = list(size=3))) + # make thicker legend lines
  ggtitle("Uniform shift") +
  facet_grid(rows = vars(Type),
             cols = vars(Participants))
p
```



Group simulation 5a (2.2a): spread difference, one group = always 200 trials, other group n trials

Simulation

```
npseq <- c(25, 50, 100)
ntseq1 <- c(seq(50, 100, 10), 150, 200)
ntseq2 <- rep(200, length(ntseq1))
maxnp <- max(npseq) # max number of participants
maxnt1 <- max(ntseq1) # max number of trials
maxnt2 <- max(ntseq2) # max number of trials
nsim <- 10000 # simulation samples
qseq <- seq(0.1,0.9,0.1) # define deciles for quantile function
ntseq <- ntseq1

# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 1.1

alpha <- 0.05

# declare matrices of results - save all iterations
res.m.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.md.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.25.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
```

```r
res.75.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfhd.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfqt8.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

pvals.0.hd <- vector(mode = "numeric", length = 9)
pvals.10.hd <- vector(mode = "numeric", length = 9)
pvals.20.hd <- vector(mode = "numeric", length = 9)

pvals.0.qt8 <- vector(mode = "numeric", length = 9)
pvals.10.qt8 <- vector(mode = "numeric", length = 9)
pvals.20.qt8 <- vector(mode = "numeric", length = 9)

# create 2 populations
set.seed(21)
npop <- 1000000
pop2 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau)
md2 <- median(pop2)
pop1 <- (pop2-md2) * ES + md2

beep(3)

# simulation samples
mc.data1 <- array(sample(pop1, maxnt1*maxnp*nsim, replace = TRUE), dim = c(maxnt1, maxnp, nsim))
mc.data2 <- array(sample(pop2, maxnt2*maxnp*nsim, replace = TRUE), dim = c(maxnt2, maxnp, nsim))

for(TR in 1:length(ntseq)){ # number of trials
  print(paste0("number of trials: ",ntseq[TR],"..."))
  beep(2)

  for(PT in 1:length(npseq)){ # number of participants
    print(paste0("number of participants: ",npseq[PT],"..."))

    # compute estimates
    todo1 <- mc.data1[1:ntseq1[TR], 1:npseq[PT],]
    todo2 <- mc.data2[1:ntseq2[TR], 1:npseq[PT],]
    mc.m <- apply(todo1, c(2,3), mean) - apply(todo2, c(2,3), mean) # diff in mean RTs
    mc.md <- apply(todo1, c(2,3), median) - apply(todo2, c(2,3), median) # diff in median RTs
    mc.25 <- apply(todo1, c(2,3), quantile, probs = .25) - apply(todo2, c(2,3), quantile, probs = .25)
    mc.75 <- apply(todo1, c(2,3), quantile, probs = .75) - apply(todo2, c(2,3), quantile, probs = .75)

    # tests for each simulation ====================
    # one-sample t-test
    res.m.m.sig[, TR, PT] <- ttest.sig(mc.m)
    # parametric median test
    res.md.md.sig[, TR, PT] <- apply(mc.md, 2, sint.sig)
```

```
    res.25.md.sig[, TR, PT] <- apply(mc.25, 2, sint.sig)
    res.75.md.sig[, TR, PT] <- apply(mc.75, 2, sint.sig)

    # shift function
    # array of differences: 9 deciles x participants x simulations
    mc.hd <- apply(todo1, c(2,3), hdseq) - apply(todo2, c(2,3), hdseq)
    mc.qt8 <- apply(todo1, c(2,3), quantile, probs = qseq, type = 8) -
              apply(todo2, c(2,3), quantile, probs = qseq, type = 8)
    # median test for each decile distribution
    pval.hd <- apply(mc.hd, c(1,3), sintv2.pval)
    pval.qt8 <- apply(mc.qt8, c(1,3), sintv2.pval)
    # trimmed mean tests for each decile distribution
    tvals.0.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0)
    tvals.10.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.1)
    tvals.20.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.2)
    tvals.0.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0)
    tvals.10.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.1)
    tvals.20.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.2)
    df0 <- df.yuen(mc.hd[1,,1], tr = 0)
    df10 <- df.yuen(mc.hd[1,,1], tr = 0.1)
    df20 <- df.yuen(mc.hd[1,,1], tr = 0.2)
    for(S in 1:nsim){
      res.sfhd.md.sig[S, TR, PT] <- sum(p.adjust(pval.hd[,S], method = "hochberg") <= alpha)
      res.sfqt8.md.sig[S, TR, PT] <- sum(p.adjust(pval.qt8[,S], method = "hochberg") <= alpha)

      for(Q in 1:9){
        pvals.0.hd[Q] <- cpval(tvals.0.hd[S,Q], df0)
        pvals.10.hd[Q] <- cpval(tvals.10.hd[S,Q], df10)
        pvals.20.hd[Q] <- cpval(tvals.20.hd[S,Q], df20)
        pvals.0.qt8[Q] <- cpval(tvals.0.qt8[S,Q], df0)
        pvals.10.qt8[Q] <- cpval(tvals.10.qt8[S,Q], df10)
        pvals.20.qt8[Q] <- cpval(tvals.20.qt8[S,Q], df20)
      }
      res.sfhd.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.hd, method = "hochberg") <= alpha)
      res.sfhd.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.hd, method = "hochberg") <= alpha)
      res.sfhd.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.hd, method = "hochberg") <= alpha)
      res.sfqt8.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.qt8, method = "hochberg") <= alpha)
      res.sfqt8.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.qt8, method = "hochberg") <= alpha)
      res.sfqt8.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.qt8, method = "hochberg") <= alpha)
    }
  }
}

save(
  res.m.m.sig,
  res.md.md.sig,
  res.25.md.sig,
  res.75.md.sig,
  res.sfhd.md.sig,
  res.sfhd.tm10.sig,
  res.sfhd.tm20.sig,
  res.sfhd.m.sig,
  res.sfqt8.md.sig,
```

```
  res.sfqt8.tm10.sig,
  res.sfqt8.tm20.sig,
  res.sfqt8.m.sig,
  ntseq1,
  ntseq2,
  ntseq,
  npseq,
  mu,
  ES,
  sigma,
  tau,
  nsim,
  file=paste0('./data/sim_gp_tp5.RData'))
beep(8)
```

Compute true positive probability

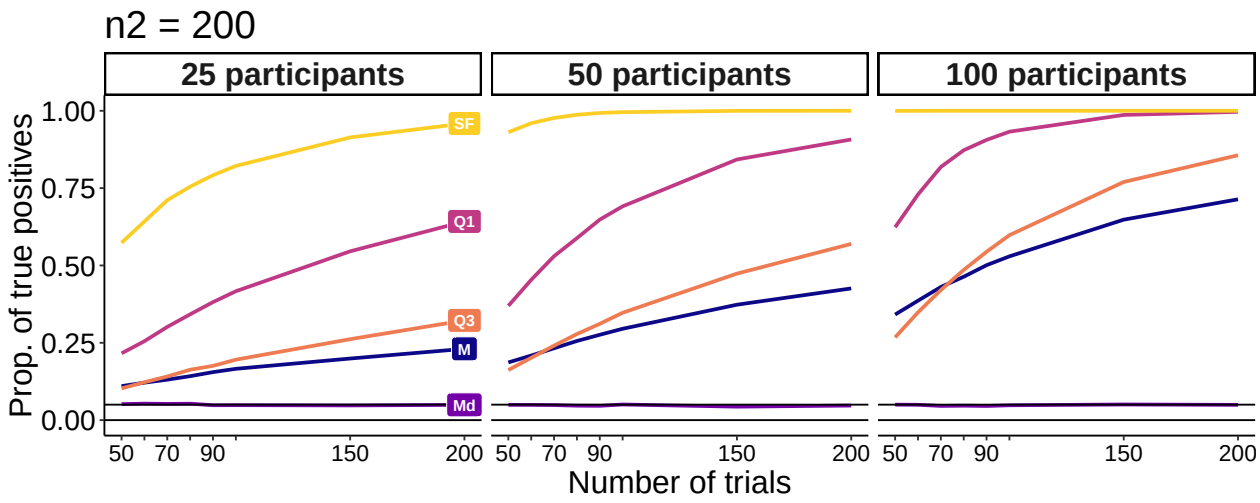
```
load('./data/sim_gp_tp5.RData')
fp.m.m <- apply(res.m.m.sig, c(2,3), mean)
fp.md.md <- apply(res.md.md.sig, c(2,3), mean)
fp.25.md <- apply(res.25.md.sig, c(2,3), mean)
fp.75.md <- apply(res.75.md.sig, c(2,3), mean)
fp.sfhd.md <- apply(res.sfhd.md.sig > 0, c(2,3), mean)
fp.sfhd.m <- apply(res.sfhd.m.sig > 0, c(2,3), mean)
fp.sfhd.tm10 <- apply(res.sfhd.tm10.sig > 0, c(2,3), mean)
fp.sfhd.tm20 <- apply(res.sfhd.tm20.sig > 0, c(2,3), mean)
fp.sfqt8.md <- apply(res.sfqt8.md.sig > 0, c(2,3), mean)
fp.sfqt8.m <- apply(res.sfqt8.m.sig > 0, c(2,3), mean)
fp.sfqt8.tm10 <- apply(res.sfqt8.tm10.sig > 0, c(2,3), mean)
fp.sfqt8.tm20 <- apply(res.sfqt8.tm20.sig > 0, c(2,3), mean)
```

Illustrate results

M, Md, Q1, Q3, SF

```
# add labels to facet 1
lab_text <- tibble(FP = c(fp.m.m[length(ntseq), 1],
                          fp.md.md[length(ntseq), 1],
                          fp.25.md[length(ntseq), 1],
                          fp.75.md[length(ntseq), 1],
                          fp.sfqt8.tm20[length(ntseq), 1]),
                   Estimator = factor(c("Mean", "Median", "Q1", "Q3", "SF")),
                   lab = c("M", "Md", "Q1", "Q3", "SF"),
                   Trials = rep(ntseq[length(ntseq)],5),
                   Participants = factor(25,levels = c("25","50","100")))

FP.in <- c(as.vector(fp.m.m),as.vector(fp.md.md),
        as.vector(fp.25.md),as.vector(fp.75.md),as.vector(fp.sfqt8.tm20))
Est.in <- c("Mean", "Median", "Q1", "Q3", "SF")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("n2 = 200") +
```

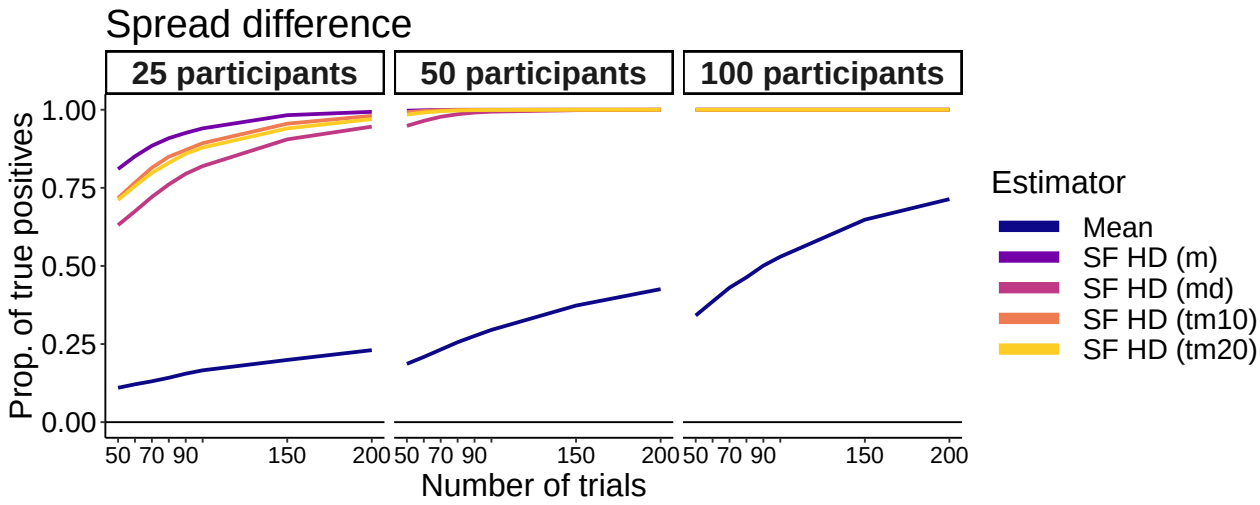
```
      theme(legend.position = "none")  +
      geom_hline(yintercept = 0.05)

p <- p +  geom_label(data = lab_text,
                     aes(x = Trials, y = FP, label = lab, fill = Estimator),
                         colour = 'white', fontface = "bold", size = 3) +
          scale_fill_manual(values = cm$colour, guide = FALSE)
p
```




```
p.spr.res2 <- p
```

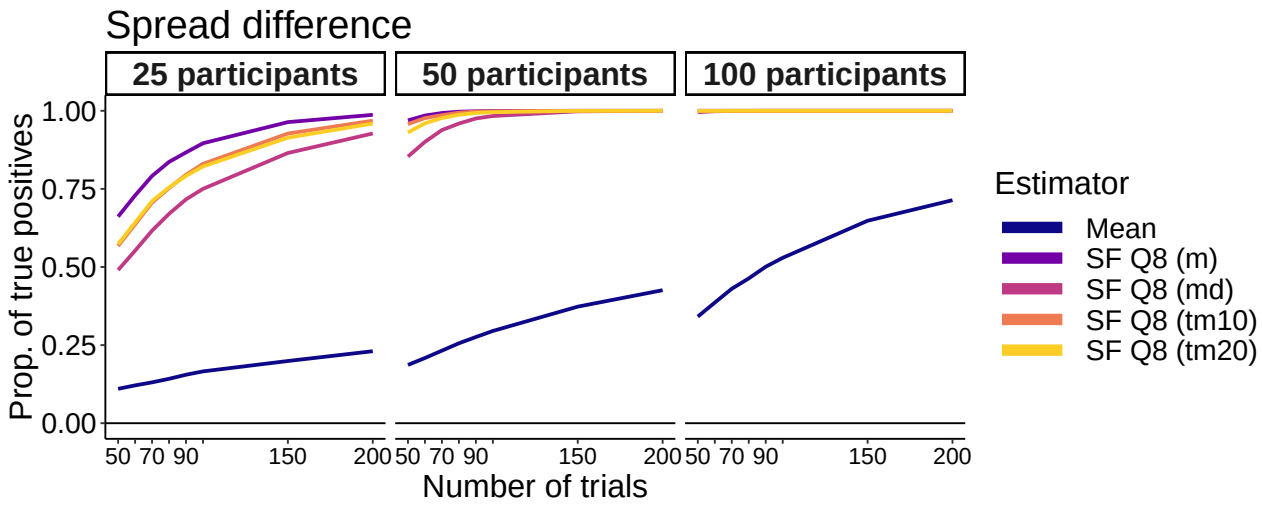
M, SF HD

```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20))
Est.in <- c("Mean","SF HD (m)", "SF HD (md)", "SF HD (tm10)","SF HD (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Spread difference")
p
```



M, SF QT8

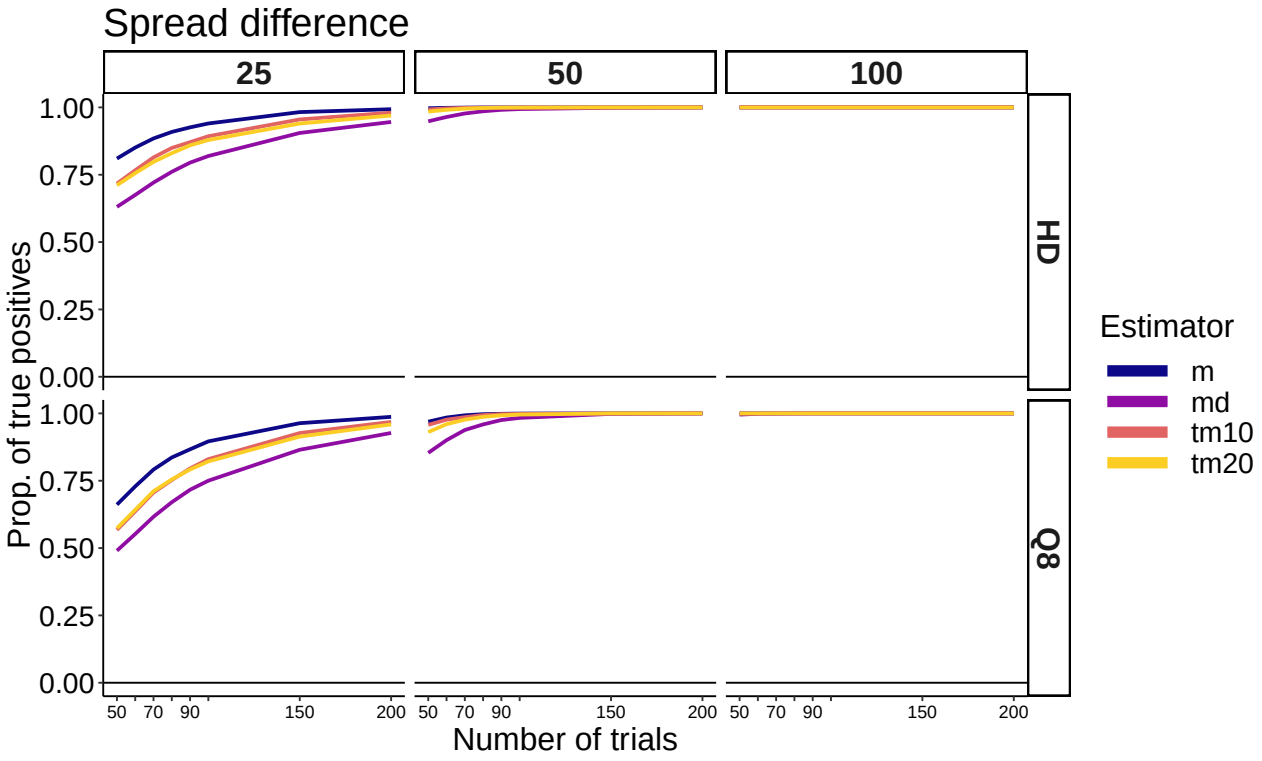
```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20))
Est.in <- c("Mean","SF Q8 (m)", "SF Q8 (md)", "SF Q8 (tm10)","SF Q8 (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Spread difference")
p
```



Compare SF results

```
df <- tibble(FP = c(as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20)),
             Estimator = factor(rep(rep(c("m", "md", "tm10","tm20"),2),
               each=length(ntseq)*length(npseq))),
             Type = factor(rep(c("Q8","HD"), each = length(ntseq)*length(npseq)*4)),
             Trials = rep(ntseq,length(npseq)*8),
             Participants = factor(rep(rep(npseq,each=length(ntseq)),8))
             )
# make plot
p <- ggplot(df) + theme_classic() +
  geom_line(aes(x=Trials, y=FP, colour = Estimator), size = 1) +
  geom_abline(intercept=0, slope=0, colour="black") +
  scale_colour_viridis_d(option = "plasma", end=0.9) +
  scale_x_continuous(breaks = ntseq,
                     labels = c("50","","70","","90","","150","200")) +
  coord_cartesian(ylim=c(0,1)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 10, colour="black"),
        axis.text.y = element_text(size = 16, colour="black"),
        axis.title.y = element_text(size = 18),
        legend.key.width = unit(1.5,"cm"),
        legend.position = "right",#c(0.55,0.85),
```

```
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.text = element_text(size=18, face="bold"),
        strip.background = element_rect(colour="black", fill="white")) +
  labs(x = "Number of trials", y = "Prop. of true positives") +
  guides(colour = guide_legend(override.aes = list(size=3))) + # make thicker legend lines
  ggtitle("Spread difference") +
  facet_grid(rows = vars(Type),
             cols = vars(Participants))
p
```



Group simulation 5b (2.2b): early difference, one group = always 200 trials, other group n trials

Simulation

```
npseq <- c(25, 50, 100)
ntseq1 <- c(seq(50, 100, 10), 150, 200)
ntseq2 <- rep(200, length(ntseq1))
maxnp <- max(npseq) # max number of participants
maxnt1 <- max(ntseq1) # max number of trials
maxnt2 <- max(ntseq2) # max number of trials
nsim <- 10000 # simulation samples
qseq <- seq(0.1,0.9,0.1) # define deciles for quantile function
ntseq <- ntseq1
```

```r
# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 1.1

alpha <- 0.05

# declare matrices of results - save all iterations
res.m.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.md.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.25.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.75.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfhd.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfqt8.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

pvals.0.hd <- vector(mode = "numeric", length = 9)
pvals.10.hd <- vector(mode = "numeric", length = 9)
pvals.20.hd <- vector(mode = "numeric", length = 9)

pvals.0.qt8 <- vector(mode = "numeric", length = 9)
pvals.10.qt8 <- vector(mode = "numeric", length = 9)
pvals.20.qt8 <- vector(mode = "numeric", length = 9)

# create 2 populations
set.seed(21)
npop <- 1000000
pop2 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau)
q3.2 <- quantile(pop2, probs = 0.75)
pop1 <- (pop2-q3.2) * ES + q3.2

beep(3)

# simulation samples
mc.data1 <- array(sample(pop1, maxnt1*maxnp*nsim, replace = TRUE), dim = c(maxnt1, maxnp, nsim))
mc.data2 <- array(sample(pop2, maxnt2*maxnp*nsim, replace = TRUE), dim = c(maxnt2, maxnp, nsim))

for(TR in 1:length(ntseq)){ # number of trials
  print(paste0("number of trials: ",ntseq[TR],"..."))
  beep(2)

  for(PT in 1:length(npseq)){ # number of participants
    print(paste0("number of participants: ",npseq[PT],"..."))

    # compute estimates
```

```
    todo1 <- mc.data1[1:ntseq1[TR], 1:npseq[PT],]
    todo2 <- mc.data2[1:ntseq2[TR], 1:npseq[PT],]
    mc.m <- apply(todo1, c(2,3), mean) - apply(todo2, c(2,3), mean) # diff in mean RTs
    mc.md <- apply(todo1, c(2,3), median) - apply(todo2, c(2,3), median) # diff in median RTs
    mc.25 <- apply(todo1, c(2,3), quantile, probs = .25) - apply(todo2, c(2,3), quantile, probs = .25)
    mc.75 <- apply(todo1, c(2,3), quantile, probs = .75) - apply(todo2, c(2,3), quantile, probs = .75)

    # tests for each simulation ====================
    # one-sample t-test
    res.m.m.sig[, TR, PT] <- ttest.sig(mc.m)
    # parametric median test
    res.md.md.sig[, TR, PT] <- apply(mc.md, 2, sint.sig)
    res.25.md.sig[, TR, PT] <- apply(mc.25, 2, sint.sig)
    res.75.md.sig[, TR, PT] <- apply(mc.75, 2, sint.sig)

    # shift function
    # array of differences: 9 deciles x participants x simulations
    mc.hd <- apply(todo1, c(2,3), hdseq) - apply(todo2, c(2,3), hdseq)
    mc.qt8 <- apply(todo1, c(2,3), quantile, probs = qseq, type = 8) -
              apply(todo2, c(2,3), quantile, probs = qseq, type = 8)
    # median test for each decile distribution
    pval.hd <- apply(mc.hd, c(1,3), sintv2.pval)
    pval.qt8 <- apply(mc.qt8, c(1,3), sintv2.pval)
    # trimmed mean tests for each decile distribution
    tvals.0.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0)
    tvals.10.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.1)
    tvals.20.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.2)
    tvals.0.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0)
    tvals.10.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.1)
    tvals.20.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.2)
    df0 <- df.yuen(mc.hd[1,,1], tr = 0)
    df10 <- df.yuen(mc.hd[1,,1], tr = 0.1)
    df20 <- df.yuen(mc.hd[1,,1], tr = 0.2)
    for(S in 1:nsim){
      res.sfhd.md.sig[S, TR, PT] <- sum(p.adjust(pval.hd[,S], method = "hochberg") <= alpha)
      res.sfqt8.md.sig[S, TR, PT] <- sum(p.adjust(pval.qt8[,S], method = "hochberg") <= alpha)

      for(Q in 1:9){
        pvals.0.hd[Q] <- cpval(tvals.0.hd[S,Q], df0)
        pvals.10.hd[Q] <- cpval(tvals.10.hd[S,Q], df10)
        pvals.20.hd[Q] <- cpval(tvals.20.hd[S,Q], df20)
        pvals.0.qt8[Q] <- cpval(tvals.0.qt8[S,Q], df0)
        pvals.10.qt8[Q] <- cpval(tvals.10.qt8[S,Q], df10)
        pvals.20.qt8[Q] <- cpval(tvals.20.qt8[S,Q], df20)
      }
      res.sfhd.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.hd, method = "hochberg") <= alpha)
      res.sfhd.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.hd, method = "hochberg") <= alpha)
      res.sfhd.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.hd, method = "hochberg") <= alpha)
      res.sfqt8.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.qt8, method = "hochberg") <= alpha)
      res.sfqt8.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.qt8, method = "hochberg") <= alpha)
      res.sfqt8.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.qt8, method = "hochberg") <= alpha)
    }
  }
```

```
}

save(
  res.m.m.sig,
  res.md.md.sig,
  res.25.md.sig,
  res.75.md.sig,
  res.sfhd.md.sig,
  res.sfhd.tm10.sig,
  res.sfhd.tm20.sig,
  res.sfhd.m.sig,
  res.sfqt8.md.sig,
  res.sfqt8.tm10.sig,
  res.sfqt8.tm20.sig,
  res.sfqt8.m.sig,
  ntseq1,
  ntseq2,
  ntseq,
  npseq,
  mu,
  ES,
  sigma,
  tau,
  nsim,
  file=paste0('./data/sim_gp_tp5b.RData'))
beep(8)
```

Compute true positive probability

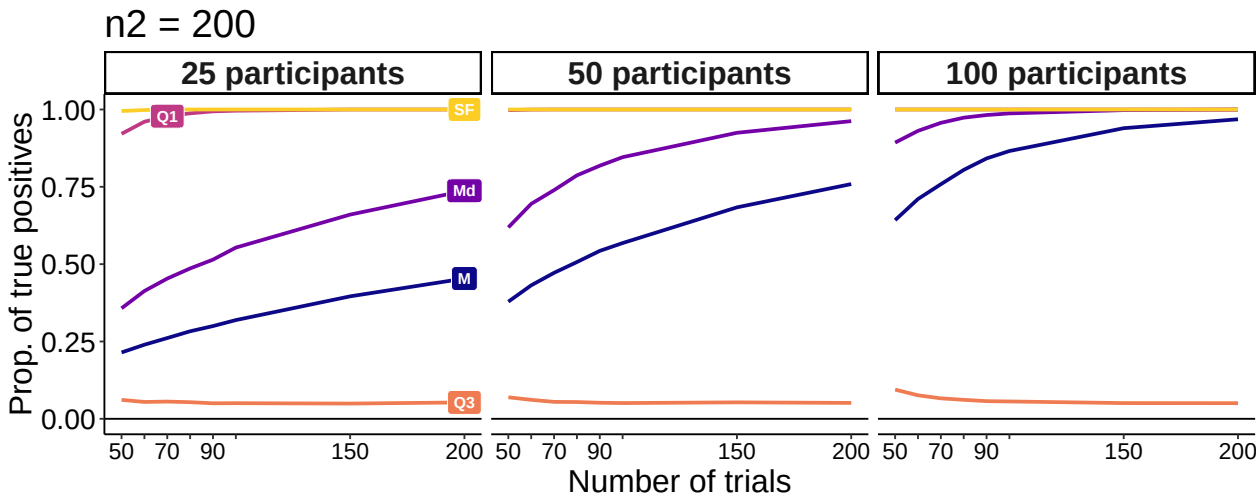
```
load('./data/sim_gp_tp5b.RData')
fp.m.m <- apply(res.m.m.sig, c(2,3), mean)
fp.md.md <- apply(res.md.md.sig, c(2,3), mean)
fp.25.md <- apply(res.25.md.sig, c(2,3), mean)
fp.75.md <- apply(res.75.md.sig, c(2,3), mean)
fp.sfhd.md <- apply(res.sfhd.md.sig > 0, c(2,3), mean)
fp.sfhd.m <- apply(res.sfhd.m.sig > 0, c(2,3), mean)
fp.sfhd.tm10 <- apply(res.sfhd.tm10.sig > 0, c(2,3), mean)
fp.sfhd.tm20 <- apply(res.sfhd.tm20.sig > 0, c(2,3), mean)
fp.sfqt8.md <- apply(res.sfqt8.md.sig > 0, c(2,3), mean)
fp.sfqt8.m <- apply(res.sfqt8.m.sig > 0, c(2,3), mean)
fp.sfqt8.tm10 <- apply(res.sfqt8.tm10.sig > 0, c(2,3), mean)
fp.sfqt8.tm20 <- apply(res.sfqt8.tm20.sig > 0, c(2,3), mean)
```

Illustrate results

M, Md, Q1, Q3, SF

```
# add labels to facet 1
lab_text <- tibble(FP = c(fp.m.m[length(ntseq), 1],
                          fp.md.md[length(ntseq), 1],
```

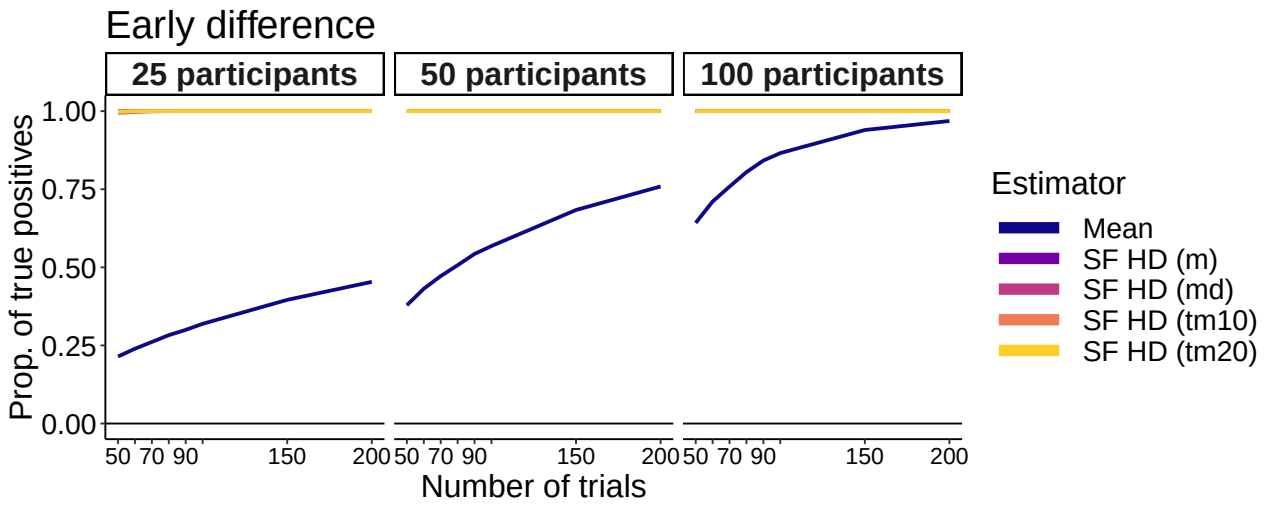
```
                          fp.25.md[3, 1],
                          fp.75.md[length(ntseq), 1],
                          fp.sfqt8.tm20[length(ntseq), 1]),
                    Estimator = factor(c("Mean", "Median", "Q1", "Q3", "SF")),
                    lab = c("M", "Md", "Q1", "Q3", "SF"),
                    Trials = c(200, 200, 70, 200, 200),
                    Participants = factor(25,levels = c("25","50","100")))

FP.in <- c(as.vector(fp.m.m),as.vector(fp.md.md),
        as.vector(fp.25.md),as.vector(fp.75.md),as.vector(fp.sfqt8.tm20))
Est.in <- c("Mean", "Median", "Q1", "Q3", "SF")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("n2 = 200") +
      theme(legend.position = "none")

p <- p +  geom_label(data = lab_text,
                     aes(x = Trials, y = FP, label = lab, fill = Estimator),
                         colour = 'white', fontface = "bold", size = 3) +
          scale_fill_manual(values = cm$colour, guide = FALSE)
p
```




```
p.early.res2 <- p
```

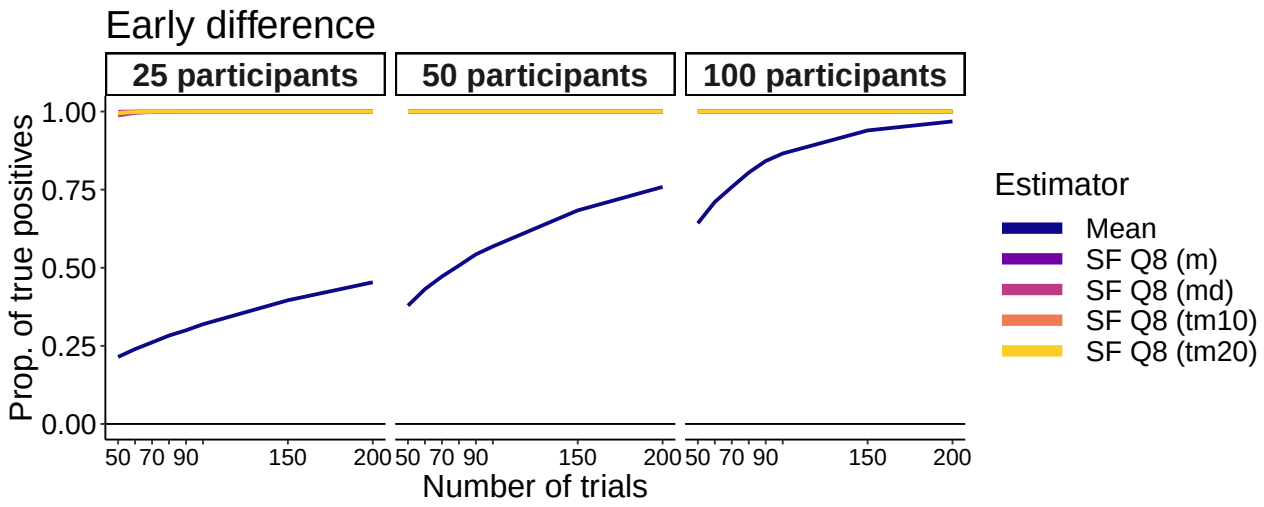
M, SF HD

```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20))
Est.in <- c("Mean","SF HD (m)", "SF HD (md)", "SF HD (tm10)","SF HD (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Early difference")
p
```



M, SF QT8

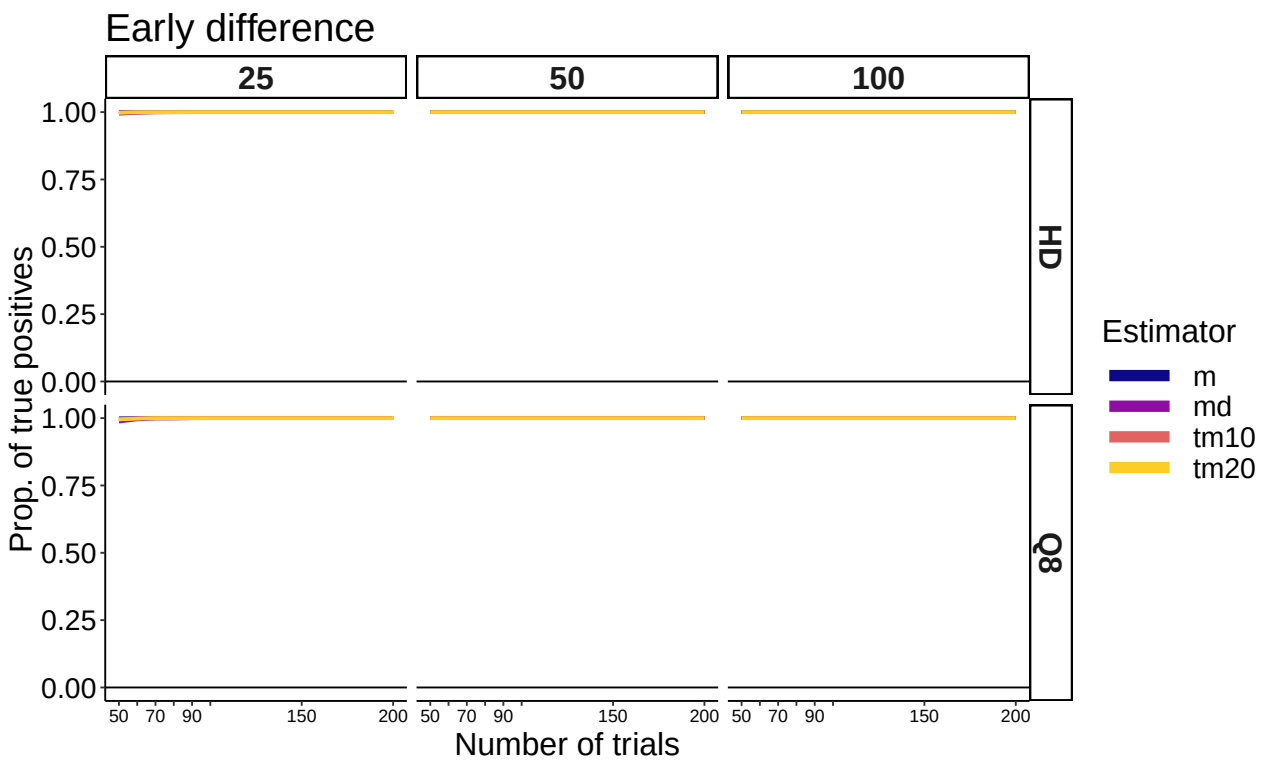
```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20))
Est.in <- c("Mean","SF Q8 (m)", "SF Q8 (md)", "SF Q8 (tm10)","SF Q8 (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Early difference")
p
```



Compare SF results

```
df <- tibble(FP = c(as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20)),
             Estimator = factor(rep(rep(c("m", "md", "tm10","tm20"),2),
               each=length(ntseq)*length(npseq))),
             Type = factor(rep(c("Q8","HD"), each = length(ntseq)*length(npseq)*4)),
             Trials = rep(ntseq,length(npseq)*8),
```

```
            Participants = factor(rep(rep(npseq,each=length(ntseq)),8))
            )
# make plot
p <- ggplot(df) + theme_classic() +
  geom_line(aes(x=Trials, y=FP, colour = Estimator), size = 1) +
  geom_abline(intercept=0, slope=0, colour="black") +
  scale_colour_viridis_d(option = "plasma", end=0.9) +
  scale_x_continuous(breaks = ntseq,
                     labels = c("50","","70","","90","","150","200")) +
  coord_cartesian(ylim=c(0,1)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 10, colour="black"),
        axis.text.y = element_text(size = 16, colour="black"),
        axis.title.y = element_text(size = 18),
        legend.key.width = unit(1.5,"cm"),
        legend.position = "right",#c(0.55,0.85),
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.text = element_text(size=18, face="bold"),
        strip.background = element_rect(colour="black", fill="white")) +
  labs(x = "Number of trials", y = "Prop. of true positives") +
  guides(colour = guide_legend(override.aes = list(size=3))) + # make thicker legend lines
  ggtitle("Early difference") +
  facet_grid(rows = vars(Type),
             cols = vars(Participants))
p
```



Group simulation 6 (2.3): tau difference, one group = always 200 trials, other group n trials

Simulation

```
npseq <- c(25, 50, 100)
ntseq1 <- c(seq(50, 100, 10), 150, 200)
ntseq2 <- rep(200, length(ntseq1))
maxnp <- max(npseq) # max number of participants
maxnt1 <- max(ntseq1) # max number of trials
maxnt2 <- max(ntseq2) # max number of trials
nsim <- 10000 # simulation samples
qseq <- seq(0.1,0.9,0.1) # define deciles for quantile function
ntseq <- ntseq1

# ex Gaussian parameters
mu <- 500
sigma <- 50
tau <- 200
ES <- 15

alpha <- 0.05

# declare matrices of results - save all iterations
res.m.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.md.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.25.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.75.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfhd.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfhd.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

res.sfqt8.md.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm10.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.tm20.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.sfqt8.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

pvals.0.hd <- vector(mode = "numeric", length = 9)
pvals.10.hd <- vector(mode = "numeric", length = 9)
pvals.20.hd <- vector(mode = "numeric", length = 9)

pvals.0.qt8 <- vector(mode = "numeric", length = 9)
pvals.10.qt8 <- vector(mode = "numeric", length = 9)
pvals.20.qt8 <- vector(mode = "numeric", length = 9)

# create 2 populations
set.seed(21)
npop <- 1000000
pop1 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau + ES)
pop2 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau)
```

```r
beep(3)

# simulation samples
mc.data1 <- array(sample(pop1, maxnt1*maxnp*nsim, replace = TRUE), dim = c(maxnt1, maxnp, nsim))
mc.data2 <- array(sample(pop2, maxnt2*maxnp*nsim, replace = TRUE), dim = c(maxnt2, maxnp, nsim))

for(TR in 1:length(ntseq)){ # number of trials
  print(paste0("number of trials: ",ntseq[TR],"..."))
  beep(2)

  for(PT in 1:length(npseq)){ # number of participants
    print(paste0("number of participants: ",npseq[PT],"..."))

    # compute estimates
    todo1 <- mc.data1[1:ntseq1[TR], 1:npseq[PT],]
    todo2 <- mc.data2[1:ntseq2[TR], 1:npseq[PT],]
    mc.m <- apply(todo1, c(2,3), mean) - apply(todo2, c(2,3), mean) # diff in mean RTs
    mc.md <- apply(todo1, c(2,3), median) - apply(todo2, c(2,3), median) # diff in median RTs
    mc.25 <- apply(todo1, c(2,3), quantile, probs = .25) - apply(todo2, c(2,3), quantile, probs = .25)
    mc.75 <- apply(todo1, c(2,3), quantile, probs = .75) - apply(todo2, c(2,3), quantile, probs = .75)

    # tests for each simulation ======================
    # one-sample t-test
    res.m.m.sig[, TR, PT] <- ttest.sig(mc.m)
    # parametric median test
    res.md.md.sig[, TR, PT] <- apply(mc.md, 2, sint.sig)
    res.25.md.sig[, TR, PT] <- apply(mc.25, 2, sint.sig)
    res.75.md.sig[, TR, PT] <- apply(mc.75, 2, sint.sig)

    # shift function
    # array of differences: 9 deciles x participants x simulations
    mc.hd <- apply(todo1, c(2,3), hdseq) - apply(todo2, c(2,3), hdseq)
    mc.qt8 <- apply(todo1, c(2,3), quantile, probs = qseq, type = 8) -
              apply(todo2, c(2,3), quantile, probs = qseq, type = 8)
    # median test for each decile distribution
    pval.hd <- apply(mc.hd, c(1,3), sintv2.pval)
    pval.qt8 <- apply(mc.qt8, c(1,3), sintv2.pval)
    # trimmed mean tests for each decile distribution
    tvals.0.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0)
    tvals.10.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.1)
    tvals.20.hd <- apply(mc.hd, 1, ctval.yuen, tr = 0.2)
    tvals.0.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0)
    tvals.10.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.1)
    tvals.20.qt8 <- apply(mc.qt8, 1, ctval.yuen, tr = 0.2)
    df0 <- df.yuen(mc.hd[1,,1], tr = 0)
    df10 <- df.yuen(mc.hd[1,,1], tr = 0.1)
    df20 <- df.yuen(mc.hd[1,,1], tr = 0.2)
    for(S in 1:nsim){
      res.sfhd.md.sig[S, TR, PT] <- sum(p.adjust(pval.hd[,S], method = "hochberg") <= alpha)
      res.sfqt8.md.sig[S, TR, PT] <- sum(p.adjust(pval.qt8[,S], method = "hochberg") <= alpha)

      for(Q in 1:9){
        pvals.0.hd[Q] <- cpval(tvals.0.hd[S,Q], df0)
```

```
        pvals.10.hd[Q] <- cpval(tvals.10.hd[S,Q], df10)
        pvals.20.hd[Q] <- cpval(tvals.20.hd[S,Q], df20)
        pvals.0.qt8[Q] <- cpval(tvals.0.qt8[S,Q], df0)
        pvals.10.qt8[Q] <- cpval(tvals.10.qt8[S,Q], df10)
        pvals.20.qt8[Q] <- cpval(tvals.20.qt8[S,Q], df20)
      }
      res.sfhd.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.hd, method = "hochberg") <= alpha)
      res.sfhd.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.hd, method = "hochberg") <= alpha)
      res.sfhd.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.hd, method = "hochberg") <= alpha)
      res.sfqt8.m.sig[S, TR, PT] <- sum(p.adjust(pvals.0.qt8, method = "hochberg") <= alpha)
      res.sfqt8.tm10.sig[S, TR, PT] <- sum(p.adjust(pvals.10.qt8, method = "hochberg") <= alpha)
      res.sfqt8.tm20.sig[S, TR, PT] <- sum(p.adjust(pvals.20.qt8, method = "hochberg") <= alpha)
    }
  }
}

save(
  res.m.m.sig,
  res.md.md.sig,
  res.25.md.sig,
  res.75.md.sig,
  res.sfhd.md.sig,
  res.sfhd.tm10.sig,
  res.sfhd.tm20.sig,
  res.sfhd.m.sig,
  res.sfqt8.md.sig,
  res.sfqt8.tm10.sig,
  res.sfqt8.tm20.sig,
  res.sfqt8.m.sig,
  ntseq1,
  ntseq2,
  ntseq,
  npseq,
  mu,
  ES,
  sigma,
  tau,
  nsim,
  file=paste0('./data/sim_gp_tp6.RData'))
beep(8)
```

Simulation: estimate tau

```
npseq <- c(25, 50, 100)
ntseq1 <- c(seq(50, 100, 10), 150, 200)
ntseq2 <- rep(200, length(ntseq1))
maxnp <- max(npseq) # max number of participants
maxnt1 <- max(ntseq1) # max number of trials
maxnt2 <- max(ntseq2) # max number of trials
nsim <- 10000 # simulation samples

# ex Gaussian parameters
```

```r
mu <- 500
sigma <- 50
tau <- 200
ES <- 15

alpha <- 0.05

# declare matrices of results - save all iterations
# res.mu.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
# res.sigma.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))
res.tau.m.sig <- array(NA, dim = c(nsim, length(ntseq), length(npseq)))

# create 2 populations
set.seed(21)
npop <- 1000000
pop1 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau + ES)
pop2 <- rexgauss(npop, mu = mu, sigma = sigma, tau = tau)

beep(3)

# simulation samples
mc.data1 <- array(sample(pop1, maxnt1*maxnp*nsim, replace = TRUE), dim = c(maxnt1, maxnp, nsim))
mc.data2 <- array(sample(pop2, maxnt2*maxnp*nsim, replace = TRUE), dim = c(maxnt2, maxnp, nsim))

for(TR in 1:length(ntseq1)){ # number of trials
  print(paste0("number of trials: ",ntseq1[TR],"..."))
  beep(2)
  tau_diff <- matrix(NA, nrow = max(npseq), ncol = nsim)
  # Fit Ex-Gaussians
    tofit1 <- mc.data1[1:ntseq1[TR],,]
    tofit2 <- mc.data2[1:ntseq2[TR],,]

    for(iter.s in 1:nsim){
      for(iter.p in 1:max(npseq)){
       out1 <- timefit(tofit1[,iter.p,iter.s])
       out2 <- timefit(tofit2[,iter.p,iter.s])
       tau_diff[iter.p,iter.s] <- out1@par[3] - out2@par[3]
      }
    }

  for(PT in 1:length(npseq)){ # number of participants
    # one-sample t-test
    res.tau.m.sig[, TR, PT] <- ttest.sig(tau_diff[1:npseq[PT],])
  }
}

save(
  res.tau.m.sig,
  ntseq1,
  ntseq2,
  npseq,
  mu,
  ES,
```

```
  sigma,
  tau,
  nsim,
  file='./data/sim_gp_tp6_exgfit.RData')
beep(8)
```

Compute true positive probability

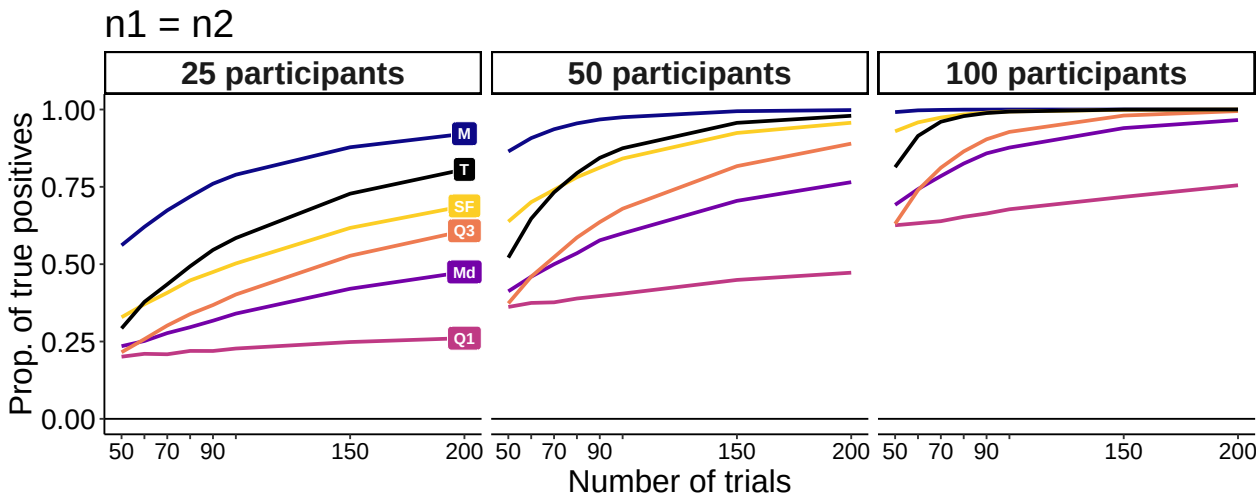
```
load('./data/sim_gp_tp6.RData')
fp.m.m <- apply(res.m.m.sig, c(2,3), mean)
fp.md.md <- apply(res.md.md.sig, c(2,3), mean)
fp.25.md <- apply(res.25.md.sig, c(2,3), mean)
fp.75.md <- apply(res.75.md.sig, c(2,3), mean)
fp.sfhd.md <- apply(res.sfhd.md.sig > 0, c(2,3), mean)
fp.sfhd.m <- apply(res.sfhd.m.sig > 0, c(2,3), mean)
fp.sfhd.tm10 <- apply(res.sfhd.tm10.sig > 0, c(2,3), mean)
fp.sfhd.tm20 <- apply(res.sfhd.tm20.sig > 0, c(2,3), mean)
fp.sfqt8.md <- apply(res.sfqt8.md.sig > 0, c(2,3), mean)
fp.sfqt8.m <- apply(res.sfqt8.m.sig > 0, c(2,3), mean)
fp.sfqt8.tm10 <- apply(res.sfqt8.tm10.sig > 0, c(2,3), mean)
fp.sfqt8.tm20 <- apply(res.sfqt8.tm20.sig > 0, c(2,3), mean)
load('./data/sim_gp_tp6_exgfit.RData')
fp.tau.m <- apply(res.tau.m.sig, c(2,3), mean)
```

Illustrate results

M, Md, Q1, Q3, SF

```
# add labels to facet 1
lab_text <- tibble(FP = c(fp.m.m[length(ntseq), 1],
                          fp.md.md[length(ntseq), 1],
                          fp.25.md[length(ntseq), 1],
                          fp.75.md[length(ntseq), 1],
                          fp.sfqt8.tm20[length(ntseq), 1],
                          fp.tau.m[length(ntseq), 1]),
                   Estimator = factor(c("Mean", "Median", "Q1", "Q3", "SF", "Tau")),
                   lab = c("M", "Md", "Q1", "Q3", "SF", "T"),
                   Trials = rep(ntseq[length(ntseq)],6),
                   Participants = factor(25,levels = c("25","50","100")))

FP.in <- c(as.vector(fp.m.m), as.vector(fp.md.md), as.vector(fp.25.md),
           as.vector(fp.75.md), as.vector(fp.sfqt8.tm20), as.vector(fp.tau.m))
Est.in <- c("Mean", "Median", "Q1", "Q3", "SF", "Tau")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("n1 = n2") +
           scale_colour_manual(values = c(cm$colour, "black"))  +
      theme(legend.position = "none")

p <- p +  geom_label(data = lab_text,
                     aes(x = Trials, y = FP, label = lab, fill = Estimator),
                         colour = 'white', fontface = "bold", size = 3) +
```

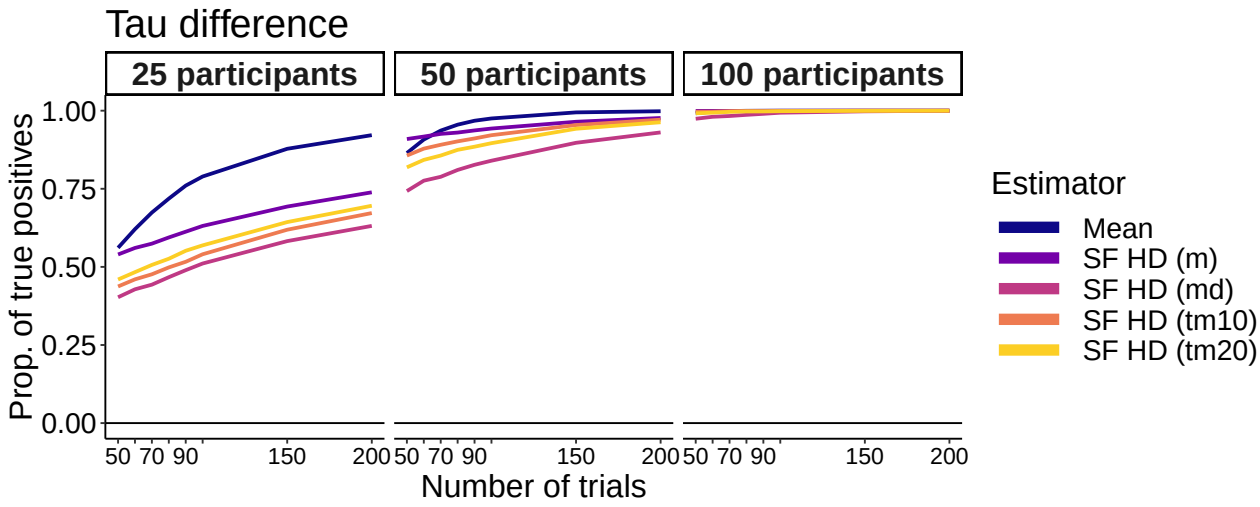
```
        scale_fill_manual(values = c(cm$colour, "black"), guide = FALSE)
p
```




```
p.tau.res2 <- p
```

M, SF HD

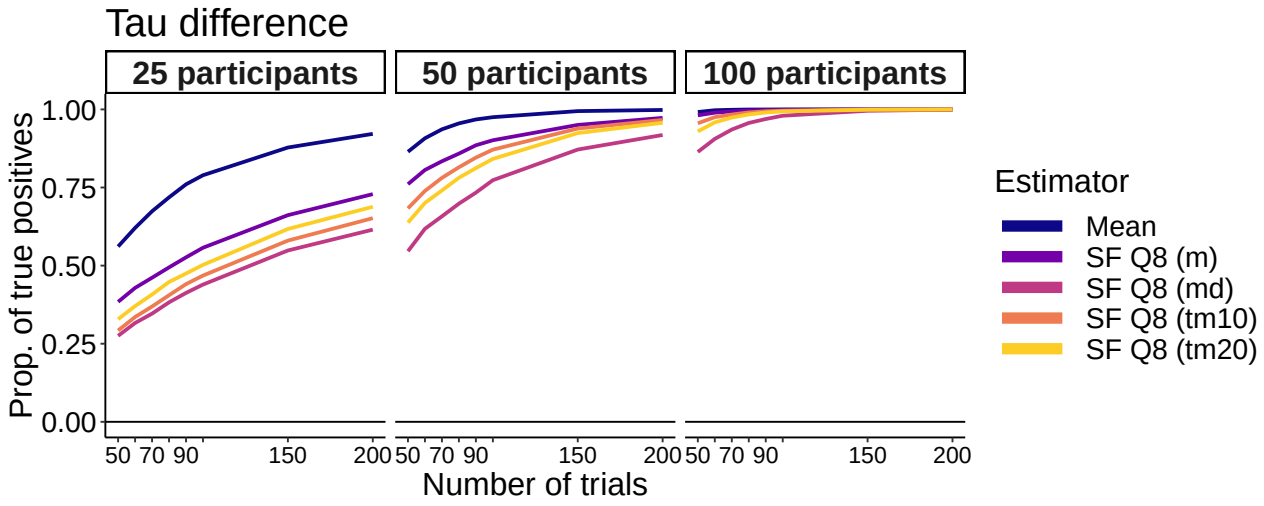
```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20))
Est.in <- c("Mean","SF HD (m)", "SF HD (md)", "SF HD (tm10)","SF HD (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Tau difference")
p
```



M, SF QT8

```
FP.in <- c(as.vector(fp.m.m),
                    as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20))
```

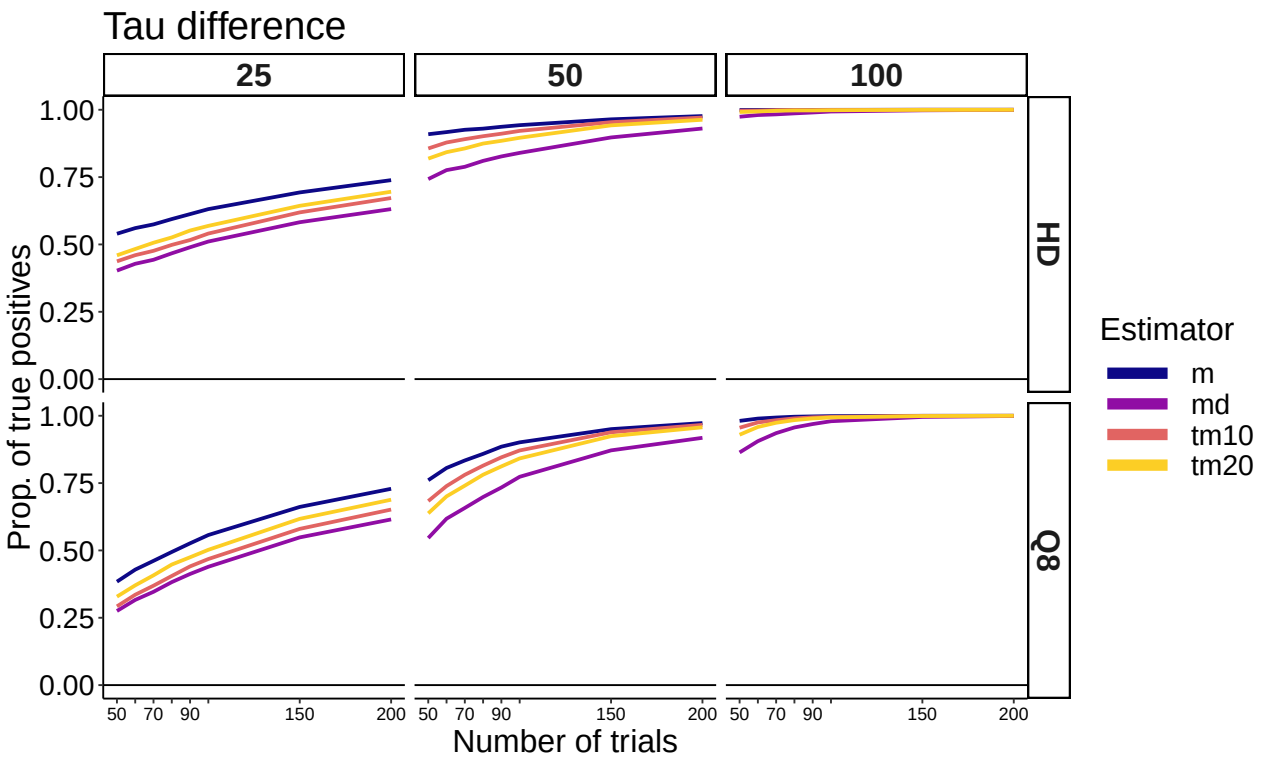
```
Est.in <- c("Mean","SF Q8 (m)", "SF Q8 (md)", "SF Q8 (tm10)","SF Q8 (tm20)")
p <- plot_tp(FP.in, Est.in, ntseq, npseq) + ggtitle("Tau difference")
p
```



Compare SF results

```
df <- tibble(FP = c(as.vector(fp.sfqt8.m),as.vector(fp.sfqt8.md),
                    as.vector(fp.sfqt8.tm10),as.vector(fp.sfqt8.tm20),
                    as.vector(fp.sfhd.m),as.vector(fp.sfhd.md),
                    as.vector(fp.sfhd.tm10),as.vector(fp.sfhd.tm20)),
             Estimator = factor(rep(rep(c("m", "md", "tm10","tm20"),2),
               each=length(ntseq)*length(npseq))),
             Type = factor(rep(c("Q8","HD"), each = length(ntseq)*length(npseq)*4)),
             Trials = rep(ntseq,length(npseq)*8),
             Participants = factor(rep(rep(npseq,each=length(ntseq)),8))
             )
# make plot
p <- ggplot(df) + theme_classic() +
  geom_line(aes(x=Trials, y=FP, colour = Estimator), size = 1) +
  geom_abline(intercept=0, slope=0, colour="black") +
  scale_colour_viridis_d(option = "plasma", end=0.9) +
  scale_x_continuous(breaks = ntseq,
                     labels = c("50","","70","","90","","150","200")) +
  coord_cartesian(ylim=c(0,1)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 10, colour="black"),
        axis.text.y = element_text(size = 16, colour="black"),
        axis.title.y = element_text(size = 18),
        legend.key.width = unit(1.5,"cm"),
        legend.position = "right",#c(0.55,0.85),
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.text = element_text(size=18, face="bold"),
        strip.background = element_rect(colour="black", fill="white")) +
```

```
  labs(x = "Number of trials", y = "Prop. of true positives") +
  guides(colour = guide_legend(override.aes = list(size=3))) + # make thicker legend lines
  ggtitle("Tau difference") +
  facet_grid(rows = vars(Type),
             cols = vars(Participants))
p
```



Summary figures

Uniform shift

```
row1 <- cowplot::plot_grid(p.uni.dist, p.uni.sf,
                           labels = c("A", "B"),
                           ncol = 2,
                           nrow = 1,
                           label_size = 20,
                           hjust = -0.5,
                           scale=.95)

cowplot::plot_grid(row1, p.uni.res, p.uni.res2,
                           labels = c("", "C", "D"),
                           ncol = 1,
                           nrow = 3,
                           label_size = 20)
# save figure
ggsave(filename=('./figures/figure_sim_gp_tp_uni_summary.pdf'),width=12,height=15)
```

Spread difference

```
row1 <- cowplot::plot_grid(p.spr.dist, p.spr.sf,
                           labels = c("A", "B"),
                           ncol = 2,
                           nrow = 1,
                           label_size = 20,
                           hjust = -0.5,
                           scale=.95)

cowplot::plot_grid(row1, p.spr.res, p.spr.res2,
                           labels = c("", "C", "D"),
                           ncol = 1,
                           nrow = 3,
                           label_size = 20)
# save figure
ggsave(filename=('./figures/figure_sim_gp_tp_spr_summary.pdf'),width=12,height=15)
```

Early difference

```
row1 <- cowplot::plot_grid(p.early.dist, p.early.sf,
                           labels = c("A", "B"),
                           ncol = 2,
                           nrow = 1,
                           label_size = 20,
                           hjust = -0.5,
                           scale=.95)

cowplot::plot_grid(row1, p.early.res, p.early.res2,
                           labels = c("", "C", "D"),
                           ncol = 1,
                           nrow = 3,
                           label_size = 20)
# save figure
ggsave(filename=('./figures/figure_sim_gp_tp_early_summary.pdf'),width=12,height=15)
```

Tau difference

```
row1 <- cowplot::plot_grid(p.tau.dist, p.tau.sf,
                           labels = c("A", "B"),
                           ncol = 2,
                           nrow = 1,
                           label_size = 20,
                           hjust = -0.5,
                           scale=.95)

cowplot::plot_grid(row1, p.tau.res, p.tau.res2,
                           labels = c("", "C", "D"),
                           ncol = 1,
                           nrow = 3,
```

```
                          label_size = 20)
# save figure
ggsave(filename=('./figures/figure_sim_gp_tp_tau_summary.pdf'),width=12,height=15)
```