Quantile estimation bias: Harrell-Davis and quantile(type=8)

Guillaume A. Rousselet

2019-01-16

Contents


```
# dependencies
library(rogme)
library(tibble)
library(tidyr)
library(cowplot)
library(retimes)
source("./functions/gh.txt")
source("./functions/akerd.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     retimes_0.1-2 cowplot_0.9.1 tidyr_0.7.2   tibble_1.4.2
## [6] rogme_0.2.0   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
```

Goal: to illustrate bias of quantiles using g & h distributions and Miller's 12 ex-Gaussian distributions. We limit our investigation to deciles, estimated using `hd()` and `quantile(type = 8)`.

The `quantile` function in base R offers 9 options. Option 8 was recommended by Hyndman and Fan (1996). It gives quantile estimates that are approximately median-unbiased regardless of the distribution of x. Preliminary tests suggest weaker bias for the extreme deciles compared to the Harrell-Davis quantile estimator. If this is the case, should be tested in `sim_gp_fp` in the hierarchical context.

g & h distributions

The ghdist() function is used to generate random numbers from g & h distributions. All such distributions have a median of zero. The parameter g controls the asymmetry of the distribution, while the parameter h controls the thickness of the tails. The g & h distributions are described in this 1985 book: http://eu.wiley.com/WileyCDA/WileyTitle/productCd-047004005X.html There is also a description in Rand Wilcox's book Introduction to Robust Estimation.

Estimate population parameters from large samples

```
set.seed(4)

gvec <- seq(0, 1, 0.1) # g values
pop.hd <- matrix(NA, nrow = length(gvec), ncol = 9)
pop.q <- matrix(NA, nrow = length(gvec), ncol = 9)
pop.skew <- matrix(NA, nrow = length(gvec), ncol = 1)
pop.kurt <- matrix(NA, nrow = length(gvec), ncol = 1)
n <- 1000000

for(G in 1:length(gvec)){
```

```
  pop <- sort(ghdist(n, g = gvec[G], h = 0))
  pop.hd[G,] <- hdseq(pop)
  for(D in 1:9){
  pop.q[G,D] <- pop[round(n*D/10)] # population deciles
  }
  pop.skew[G] <- skew(pop)
  pop.kurt[G] <- kurt(pop)
}
```

Population HD

```
round(pop.hd, digits = 2)
```

```
##        [,1]  [,2]  [,3]  [,4] [,5] [,6] [,7] [,8] [,9]
##  [1,] -1.28 -0.84 -0.52 -0.25    0 0.25 0.52 0.84 1.28
##  [2,] -1.20 -0.81 -0.51 -0.25    0 0.26 0.54 0.88 1.37
##  [3,] -1.13 -0.77 -0.50 -0.25    0 0.26 0.55 0.92 1.46
##  [4,] -1.06 -0.74 -0.48 -0.24    0 0.26 0.57 0.96 1.56
##  [5,] -1.00 -0.72 -0.47 -0.24    0 0.27 0.58 1.00 1.68
##  [6,] -0.94 -0.69 -0.46 -0.24    0 0.27 0.60 1.05 1.80
##  [7,] -0.89 -0.66 -0.45 -0.24    0 0.27 0.61 1.09 1.93
##  [8,] -0.84 -0.64 -0.44 -0.23    0 0.28 0.63 1.15 2.07
##  [9,] -0.80 -0.61 -0.43 -0.23    0 0.28 0.65 1.20 2.24
## [10,] -0.76 -0.59 -0.42 -0.23    0 0.28 0.67 1.26 2.41
## [11,] -0.72 -0.57 -0.41 -0.22    0 0.29 0.69 1.32 2.59
```

Population deciles

```
round(pop.q, digits = 2)
```

```
##        [,1]  [,2]  [,3]  [,4] [,5] [,6] [,7] [,8] [,9]
##  [1,] -1.28 -0.84 -0.52 -0.25    0 0.25 0.52 0.84 1.28
##  [2,] -1.20 -0.81 -0.51 -0.25    0 0.26 0.54 0.88 1.37
##  [3,] -1.13 -0.77 -0.50 -0.25    0 0.26 0.55 0.92 1.46
##  [4,] -1.06 -0.74 -0.48 -0.24    0 0.26 0.57 0.96 1.56
##  [5,] -1.00 -0.72 -0.47 -0.24    0 0.27 0.58 1.00 1.68
##  [6,] -0.95 -0.69 -0.46 -0.24    0 0.27 0.60 1.05 1.80
##  [7,] -0.90 -0.66 -0.45 -0.24    0 0.27 0.61 1.09 1.93
##  [8,] -0.84 -0.64 -0.44 -0.23    0 0.28 0.63 1.15 2.07
##  [9,] -0.80 -0.61 -0.43 -0.23    0 0.28 0.65 1.20 2.24
## [10,] -0.76 -0.59 -0.42 -0.23    0 0.28 0.67 1.26 2.41
## [11,] -0.72 -0.57 -0.41 -0.22    0 0.29 0.69 1.32 2.59
```

Population skewness

```
pop.skew
```

```
##               [,1]
##  [1,] -0.001221342
##  [2,]  0.301837185
```

```
##  [3,]  0.613084633
##  [4,]  0.951308319
##  [5,]  1.320886192
##  [6,]  1.762127029
##  [7,]  2.289295169
##  [8,]  2.889302163
##  [9,]  3.633011501
## [10,]  4.624178867
## [11,]  6.688973298
```

Population kurtosis

```
pop.kurt
```

```
##              [,1]
##  [1,]   3.007439
##  [2,]   3.171351
##  [3,]   3.682597
##  [4,]   4.653845
##  [5,]   6.185309
##  [6,]   8.920932
##  [7,]  13.455927
##  [8,]  20.754219
##  [9,]  31.009691
## [10,]  52.477324
## [11,] 144.376057
```

Simulation

```
gvec <- seq(0, 1, 0.1) # g values
nvec <- c(seq(30,100,10), 150, 200, 300) # vector of sample sizes to test
nsim <- 10000 # simulation samples
nboot <- 200 # samples for bias correction
prob.seq <- seq(0.1, 0.9, 0.1) # input to quantile function
q.type <- 8 # quantile type input - type 8 recommended by Hyndman and Fan (1996) - quantile estimates a

# declare matrices of results - save all iterations
sim.hd.bias <- array(NA, dim = c(nsim, length(gvec), length(nvec), 9))
sim.hd.bc <- array(NA, dim = c(nsim, length(gvec), length(nvec), 9))
boot.hd <- array(NA, dim = c(nboot, 9))
sim.q.bias <- array(NA, dim = c(nsim, length(gvec), length(nvec), 9))
sim.q.bc <- array(NA, dim = c(nsim, length(gvec), length(nvec), 9))
boot.q <- array(NA, dim = c(nboot, 9))

set.seed(21)
beep(3)

for(S in 1:nsim){ # simulation iterations

  if(S %% 500 == 0){
    print(paste0("HD bias: simulation ",S," out of ",nsim,"..."))
```

```
    beep(2)
  }

  for(G in 1:length(gvec)){

    large.sample <- ghdist(max(nvec), g = gvec[G], h = 0)

    for(N in 1:length(nvec)){ # sample sizes

      current.sample <- large.sample[1:nvec[N]]

      sim.hd.bias[S,G,N,] <- hdseq(current.sample)
      sim.q.bias[S,G,N,] <- quantile(current.sample, probs = prob.seq, type = 8)

      for(b in 1:nboot){
        boot.hd[b,] <- hdseq(sample(current.sample, nvec[N], replace = TRUE))
        boot.q[b,] <- quantile(sample(current.sample, nvec[N], replace = TRUE),
                               probs = prob.seq, type = 8)
      }
      sim.hd.bc[S,G,N,] <- 2*sim.hd.bias[S,G,N,] - apply(boot.hd, 2, mean) - pop.q[G,]
      sim.q.bc[S,G,N,] <- 2*sim.q.bias[S,G,N,] - apply(boot.q, 2, mean) - pop.q[G,]

      sim.hd.bias[S,G,N,] <- sim.hd.bias[S,G,N,] - pop.q[G,]
      sim.q.bias[S,G,N,] <- sim.q.bias[S,G,N,] - pop.q[G,]

    }
  }
}
save(
  sim.hd.bias,
  sim.q.bias,
  sim.hd.bc,
  sim.q.bc,
  gvec,
  nvec,
  nsim,
  file=paste0('./data/sim_hd_bias.RData'))
beep(8)
```

Compute bias

Bias = average of sample estimates minus population value (already subtracted during simulation).

```
load('./data/sim_hd_bias.RData')

bias.hd <- apply(sim.hd.bias, c(2,3,4), mean, na.rm = TRUE)
bias.hd.md <- apply(sim.hd.bias, c(2,3,4), median, na.rm = TRUE)
bias.hd.bc <- apply(sim.hd.bc, c(2,3,4), mean, na.rm = TRUE)

bias.q <- apply(sim.q.bias, c(2,3,4), mean, na.rm = TRUE)
bias.q.md <- apply(sim.q.bias, c(2,3,4), median, na.rm = TRUE)
bias.q.bc <- apply(sim.q.bc, c(2,3,4), mean, na.rm = TRUE)
```

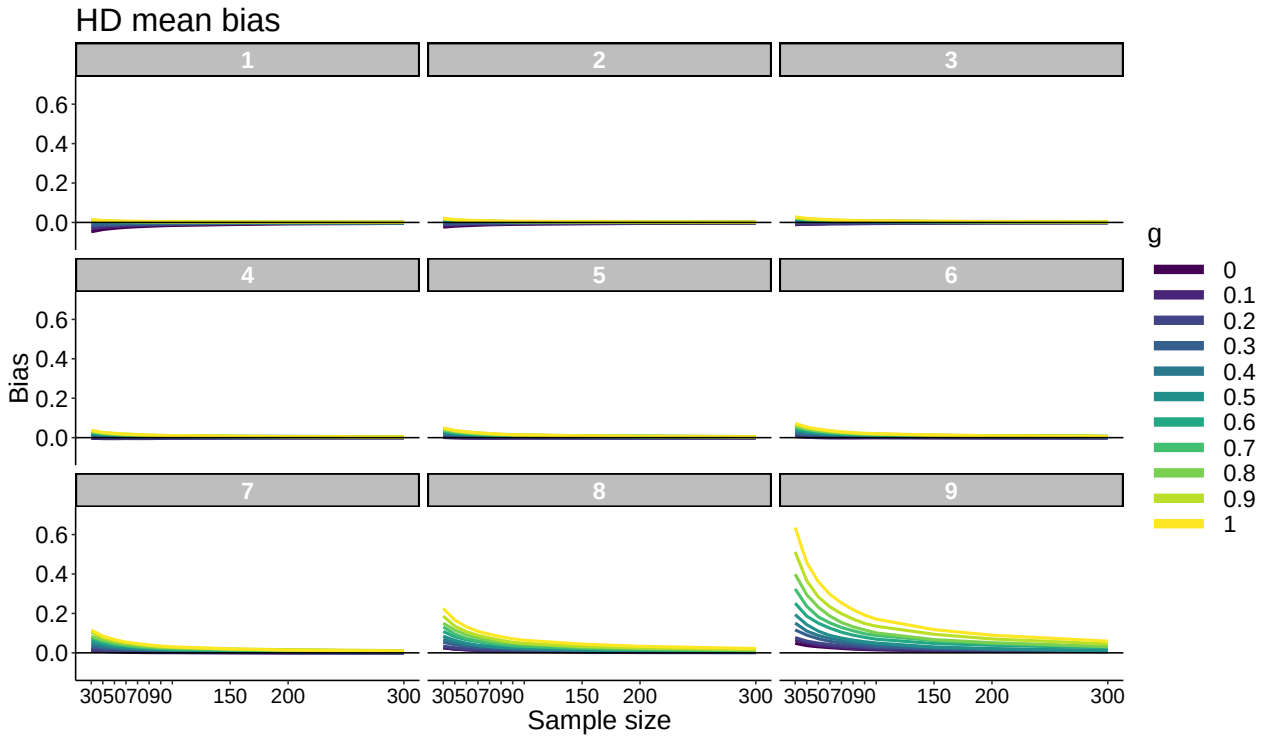
```
x.labels <- c("30","","50","","70","","90","", "150", "200", "300")
```

Illustrate bias results

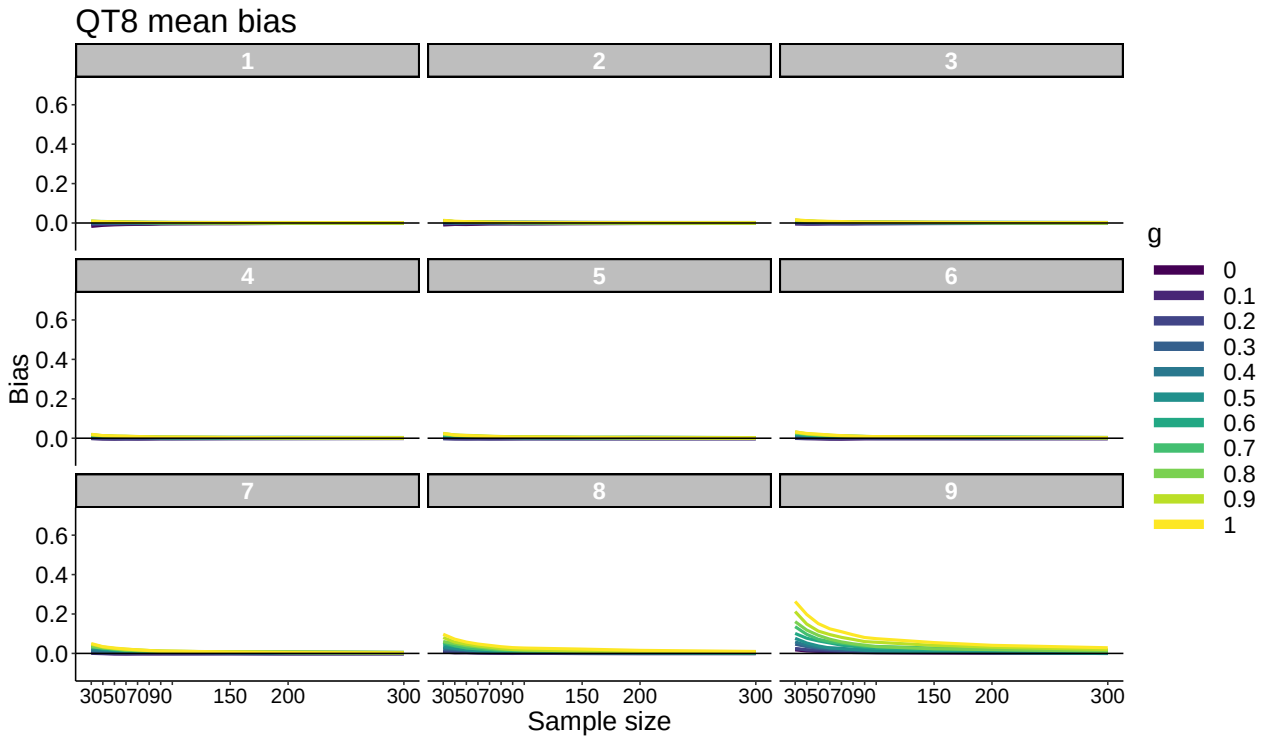
Make function

```
plot_bias_res <- function(data, gvec, nvec, x.labels){
df <- tibble(Bias = as.vector(data),
             g = factor(rep(gvec, length(nvec)*9)),
             Size = rep(rep(nvec,each=length(gvec)), 9),
             q = factor(rep(seq(1,9), each=length(gvec)*length(nvec))))

# make plot
p <- ggplot(df) + theme_classic() +
  geom_line(aes(x=Size, y=Bias, colour = g), size = 1) +
  geom_abline(intercept=0, slope=0, colour="black") +
  scale_colour_viridis_d() +
  scale_x_continuous(breaks=nvec, labels = x.labels) +
  scale_y_continuous(breaks=seq(0,0.6,0.2)) +
  coord_cartesian(ylim=c(-0.1,0.7)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 14, 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",
        # legend.position = c(0.55,0.85),
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.background=element_rect(fill="grey", colour="black"),
        strip.text=element_text(size=16, colour="white", face="bold")) +
  labs(x = "Sample size", y = "Bias") +
  # make thicker legend lines
  guides(colour = guide_legend(override.aes = list(size=3))) +
  facet_wrap( ~ q, ncol = 3)
p
}
```

```
p <- plot_bias_res(bias.hd, gvec, nvec, x.labels)
p <- p + ggtitle("HD mean bias")
p
```

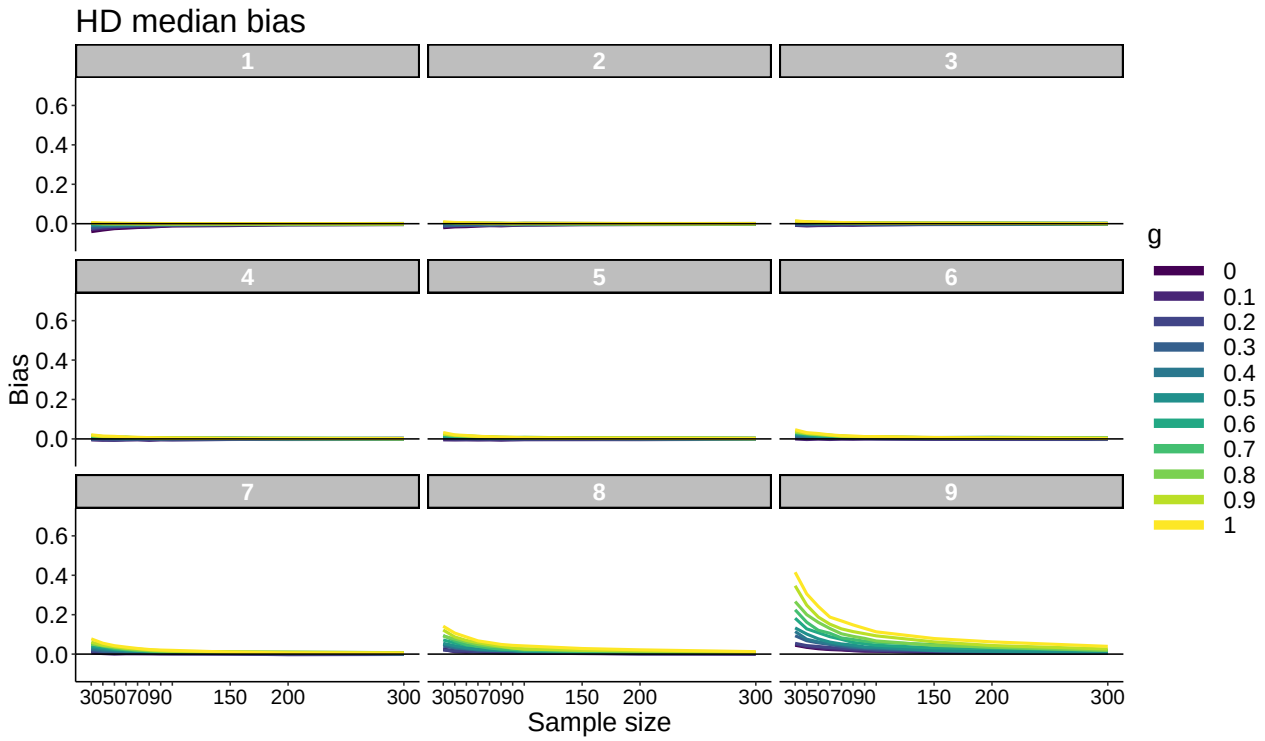



```
p <- plot_bias_res(bias.q, gvec, nvec, x.labels)
p <- p + ggtitle("QT8 mean bias")
p
```

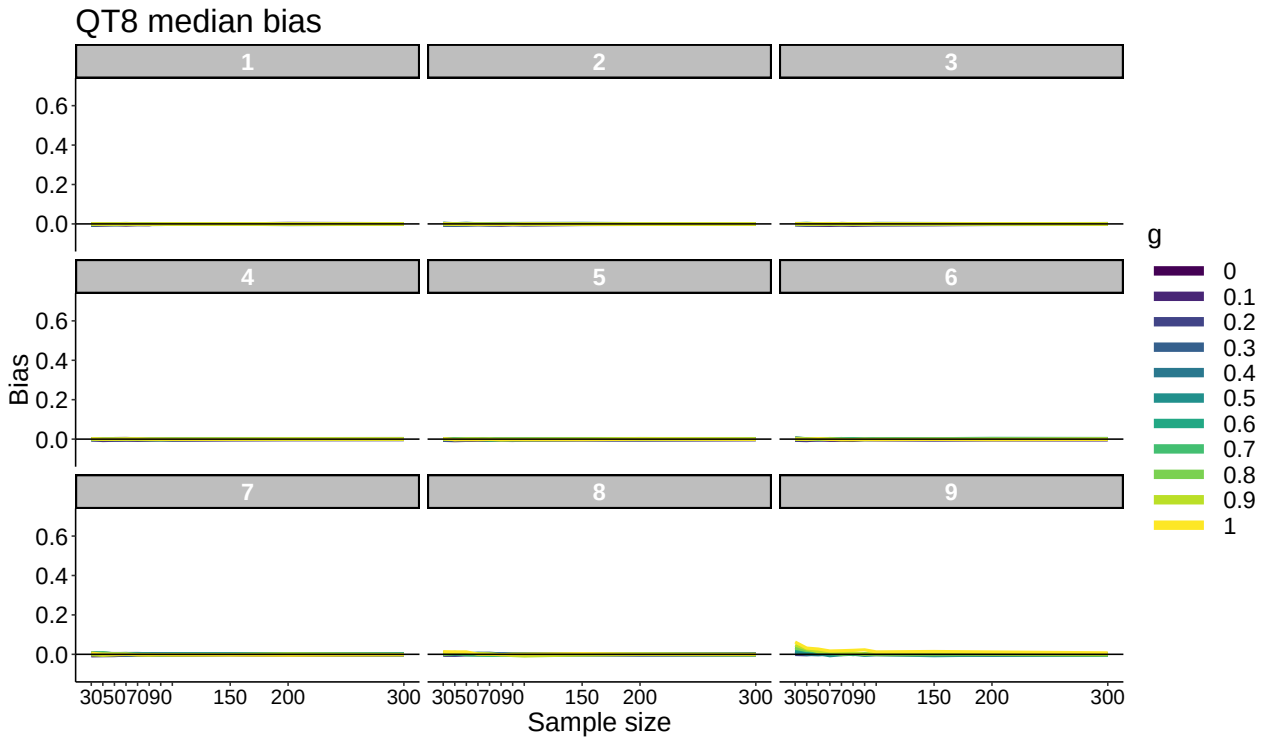


Illustrate median bias results

```
p <- plot_bias_res(bias.hd.md, gvec, nvec, x.labels)
p <- p + ggtitle("HD median bias")
p
```

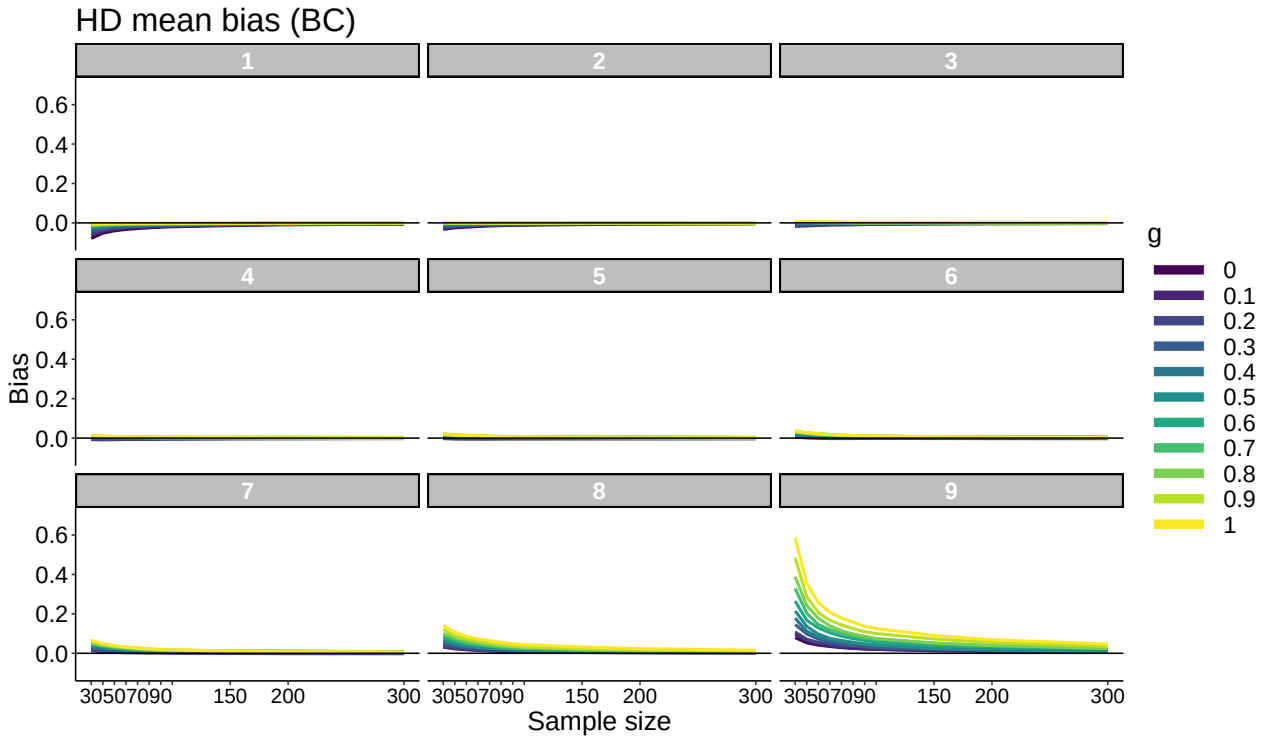



```
p <- plot_bias_res(bias.q.md, gvec, nvec, x.labels)
p <- p + ggtitle("QT8 median bias")
p
```

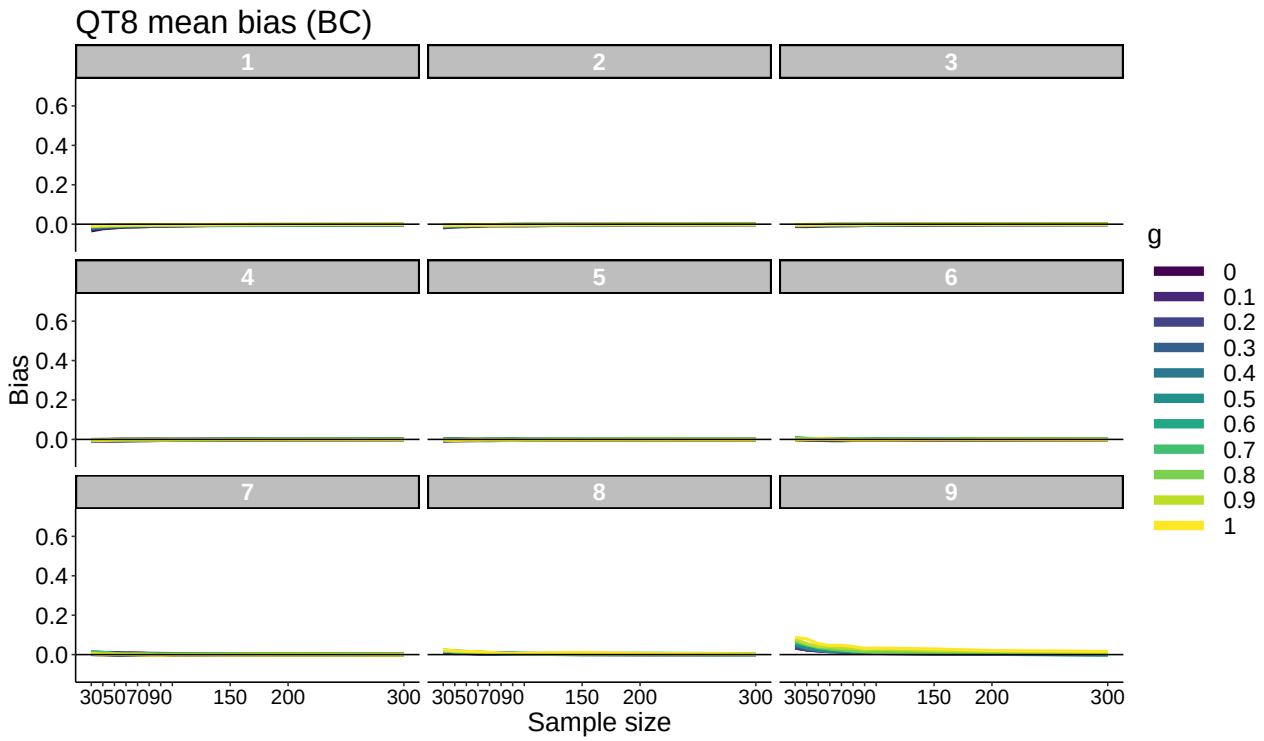


Illustrate bias corrected bias results

```
p <- plot_bias_res(bias.hd.bc, gvec, nvec, x.labels)
p <- p + ggtitle("HD mean bias (BC)")
p
```



```
p <- plot_bias_res(bias.q.bc, gvec, nvec, x.labels)
p <- p + ggtitle("QT8 mean bias (BC)")
p
```

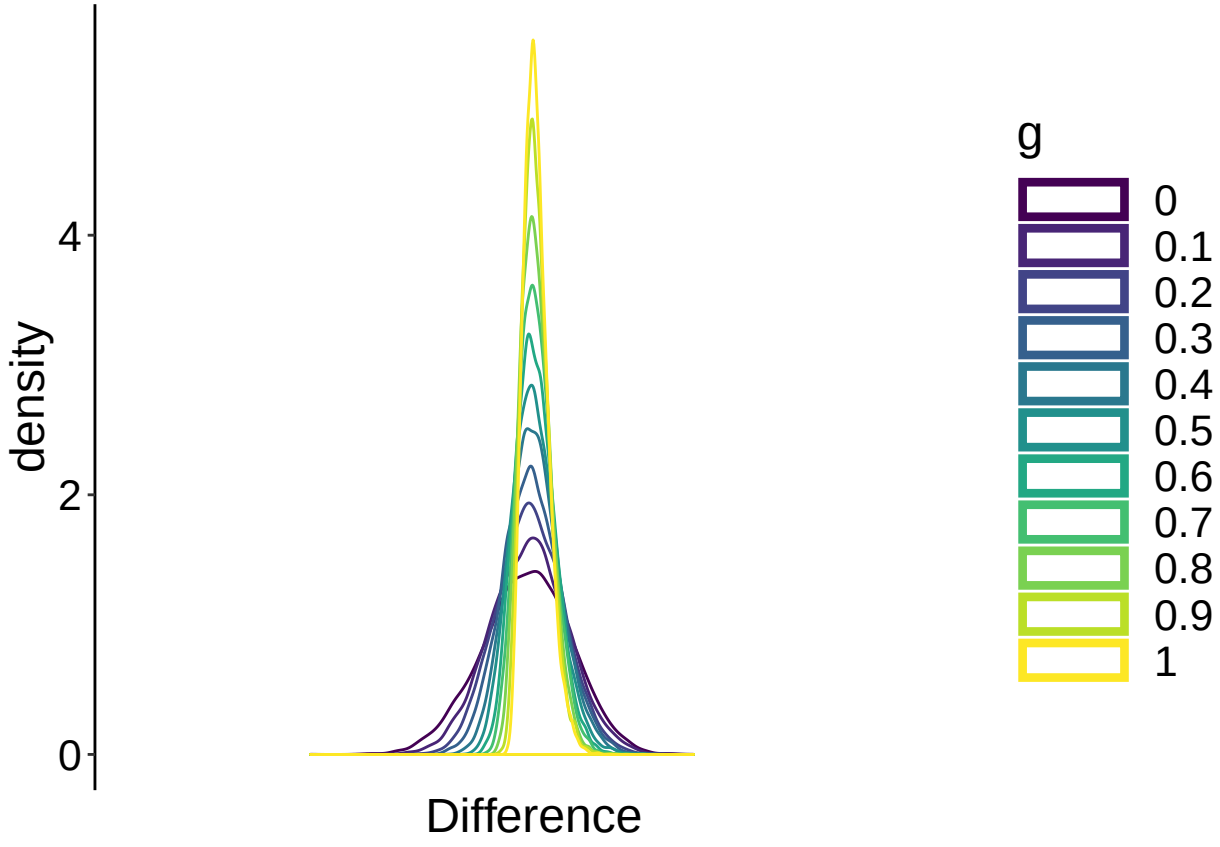


Illustrate sampling distributions: HD

Sample size = 30, q = 1

```
# fig.width = 10, fig.height = 6
df <- tibble(Difference = as.vector(sim.hd.bias[,,1,1]),
             g = factor(rep(gvec, each = nsim))
             )

# make plot
p <- ggplot(df, aes(x = Difference, colour = g)) + theme_classic() +
  geom_density() +
  scale_colour_viridis_d() +
  scale_x_continuous(breaks=nvec, labels = x.labels) +
  # scale_y_continuous(breaks=c(-5,seq(0,50,10))) +
  coord_cartesian(xlim=c(-2.5,2.5)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 14, 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(0.85,0.65),
        legend.text=element_text(size=16),
```

```
        legend.title=element_text(size=18)) +
  # labs(x = "Sample size", y = "Bias in ms") +
  # make thicker legend lines
  guides(colour = guide_legend(override.aes = list(size=3)))
 # + ggtitle("Median RT")
p
```

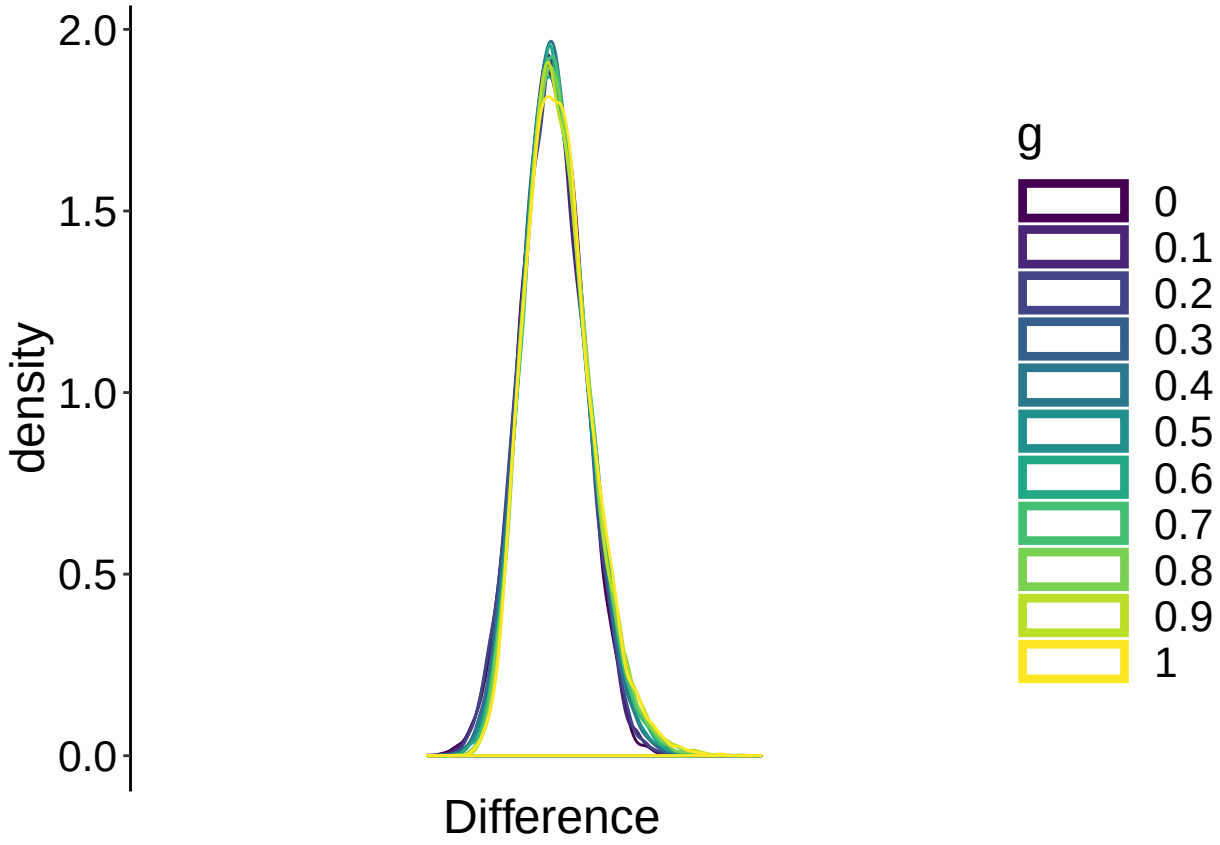



```
# save figure
# ggsave(filename=paste0('figure_miller_bias_md.pdf'),width=10,height=6)
```

Sample size = 30, q = 5

```
# fig.width = 10, fig.height = 6
df <- tibble(Difference = as.vector(sim.hd.bias[,,1,5]),
             g = factor(rep(gvec, each = nsim))
             )

# make plot
p <- ggplot(df, aes(x = Difference, colour = g)) + theme_classic() +
  geom_density() +
  scale_colour_viridis_d() +
  scale_x_continuous(breaks=nvec, labels = x.labels) +
  # scale_y_continuous(breaks=c(-5,seq(0,50,10))) +
  coord_cartesian(xlim=c(-2.5,2.5)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
```

```
        axis.text.x = element_text(size = 14, 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(0.85,0.65),
        legend.text=element_text(size=16),
        legend.title=element_text(size=18)) +
  # labs(x = "Sample size", y = "Bias in ms") +
  # make thicker legend lines
  guides(colour = guide_legend(override.aes = list(size=3)))
 # + ggtitle("Median RT")
p
```

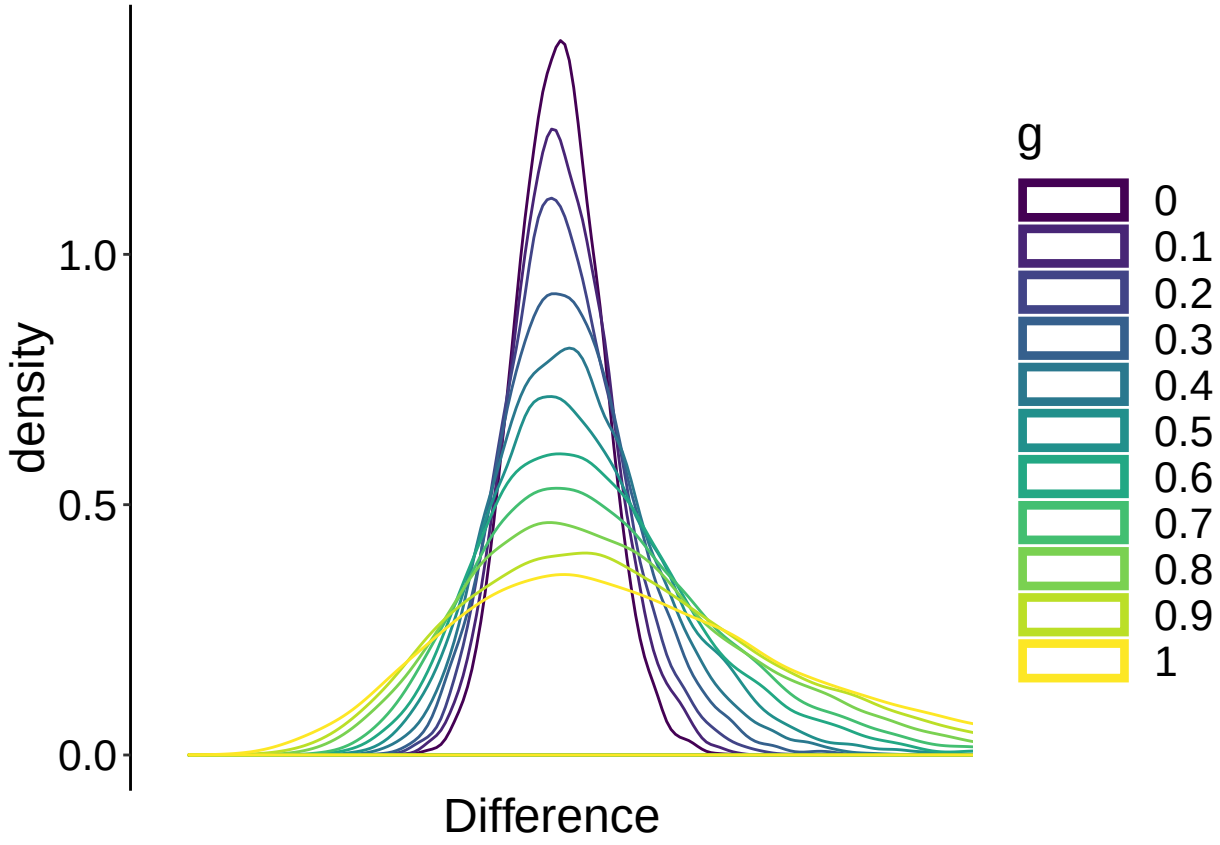



```
# save figure
# ggsave(filename=paste0('figure_miller_bias_md.pdf'),width=10,height=6)
```

Sample size = 30, q = 9

```
# fig.width = 10, fig.height = 6
df <- tibble(Difference = as.vector(sim.hd.bias[,,1,9]),
             g = factor(rep(gvec, each = nsim))
             )

# make plot
p <- ggplot(df, aes(x = Difference, colour = g)) + theme_classic() +
  geom_density() +
```

```
  scale_colour_viridis_d() +
  scale_x_continuous(breaks=nvec, labels = x.labels) +
  # scale_y_continuous(breaks=c(-5,seq(0,50,10))) +
  coord_cartesian(xlim=c(-2.5,2.5)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 14, 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(0.85,0.65),
        legend.text=element_text(size=16),
        legend.title=element_text(size=18)) +
  # labs(x = "Sample size", y = "Bias in ms") +
  # make thicker legend lines
  guides(colour = guide_legend(override.aes = list(size=3)))
 # + ggtitle("Median RT")
p
```

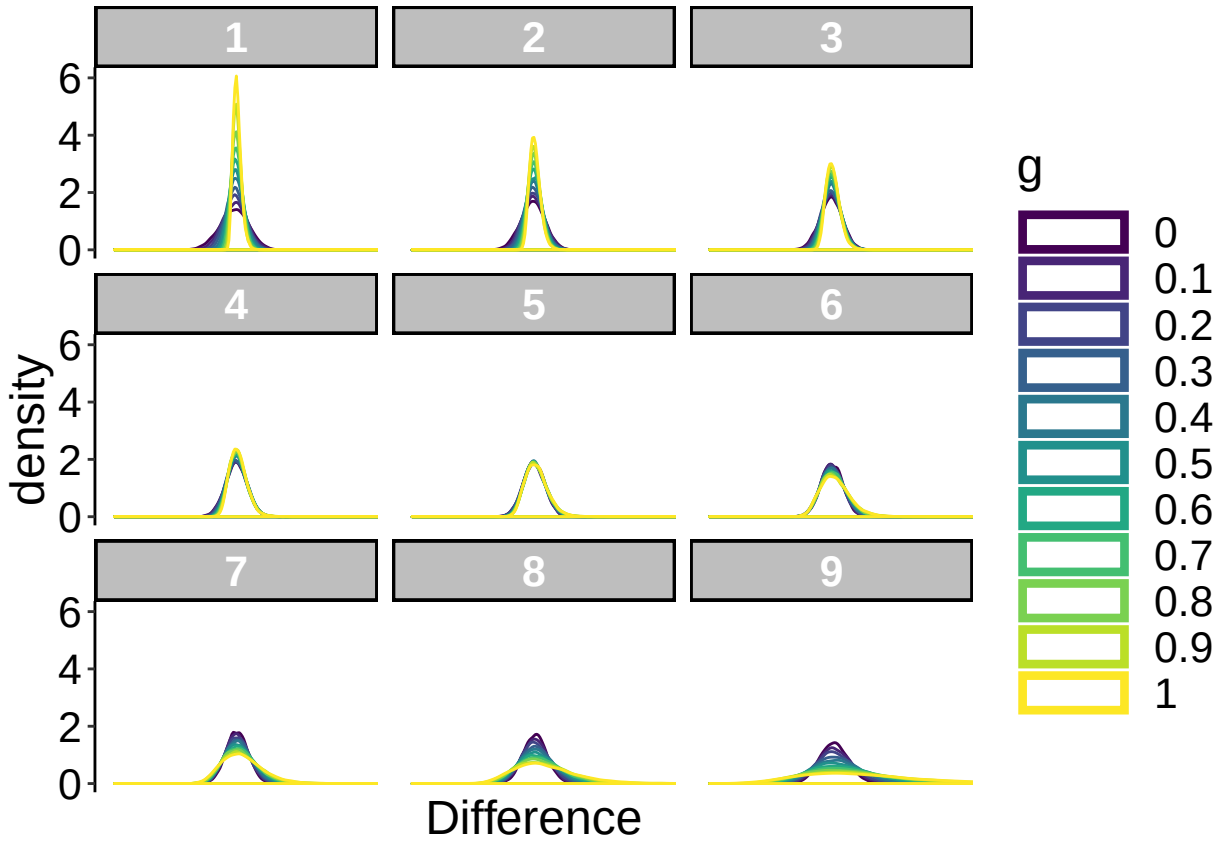



```
# save figure
# ggsave(filename=paste0('figure_miller_bias_md.pdf'),width=10,height=6)
```

Sample size = 30

```
# fig.width = 10, fig.height = 6
df <- tibble(Difference = as.vector(sim.hd.bias[,,1,]),
```

```
            g = factor(rep(rep(gvec, each = nsim),9)),
            q = factor(rep(seq(1,9), each = nsim*length(gvec)))
            )

# make plot
p <- ggplot(df, aes(x = Difference, colour = g)) + theme_classic() +
  geom_density() +
  scale_colour_viridis_d() +
  scale_x_continuous(breaks=nvec, labels = x.labels) +
  # scale_y_continuous(breaks=c(-5,seq(0,50,10))) +
  coord_cartesian(xlim=c(-2.5,2.5)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 14, 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(0.85,0.65),
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.background=element_rect(fill="grey", colour="black"),
        strip.text=element_text(size=16, colour="white", face="bold")) +
  # labs(x = "Sample size", y = "Bias in ms") +
  # make thicker legend lines
  guides(colour = guide_legend(override.aes = list(size=3))) +
 facet_wrap( ~ q, ncol = 3)
 # + ggtitle("Median RT")
p
```

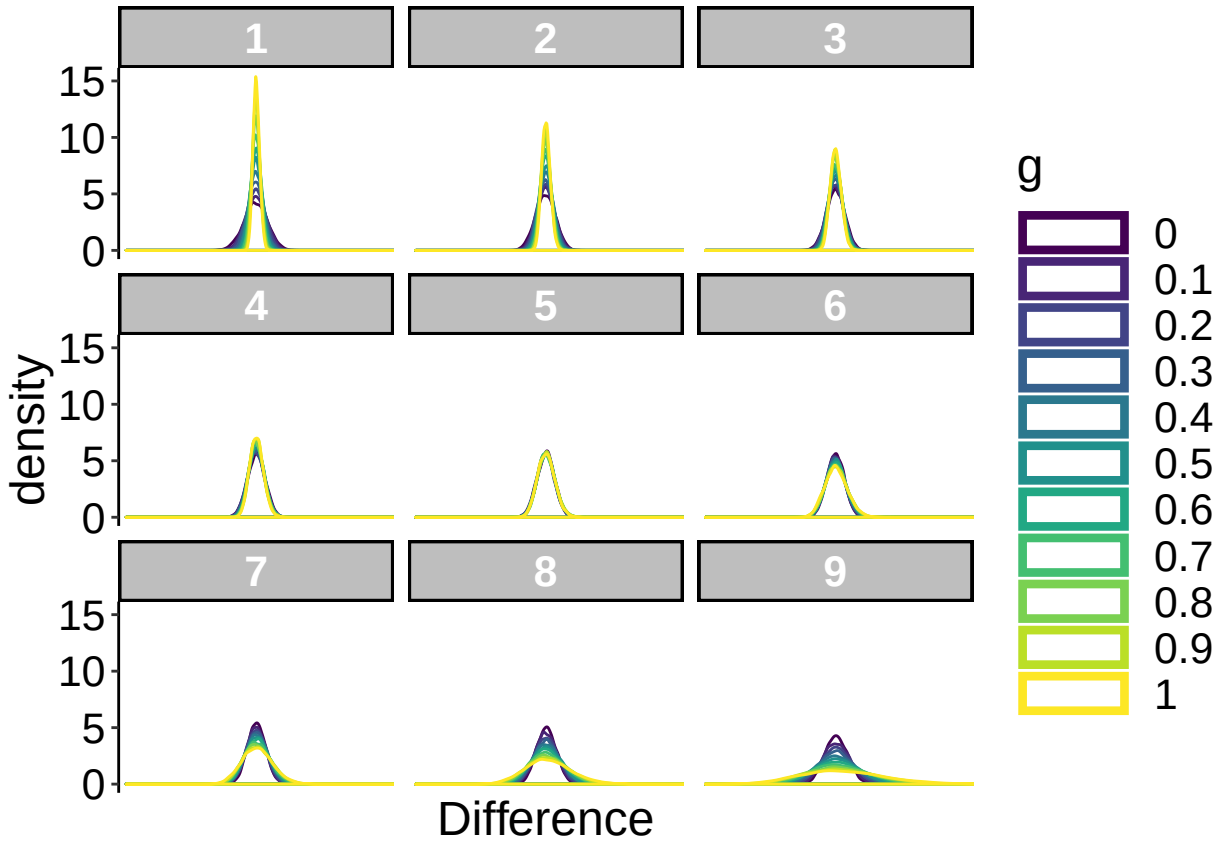



```
# save figure
# ggsave(filename=paste0('figure_miller_bias_md.pdf'),width=10,height=6)
```

Sample size = 300

```
# fig.width = 10, fig.height = 6
df <- tibble(Difference = as.vector(sim.hd.bias[,,11,]),
             g = factor(rep(rep(gvec, each = nsim),9)),
             q = factor(rep(seq(1,9), each = nsim*length(gvec)))
             )

# make plot
p <- ggplot(df, aes(x = Difference, colour = g)) + theme_classic() +
  geom_density() +
  scale_colour_viridis_d() +
  scale_x_continuous(breaks=nvec, labels = x.labels) +
  # scale_y_continuous(breaks=c(-5,seq(0,50,10))) +
  coord_cartesian(xlim=c(-1,1)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 14, 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(0.85,0.65),
        legend.text=element_text(size=16),
```

```
        legend.title=element_text(size=18),
        strip.background=element_rect(fill="grey", colour="black"),
        strip.text=element_text(size=16, colour="white", face="bold")) +
  # labs(x = "Sample size", y = "Bias in ms") +
  # make thicker legend lines
  guides(colour = guide_legend(override.aes = list(size=3))) +
 facet_wrap( ~ q, ncol = 3)
 # + ggtitle("Median RT")
p
```

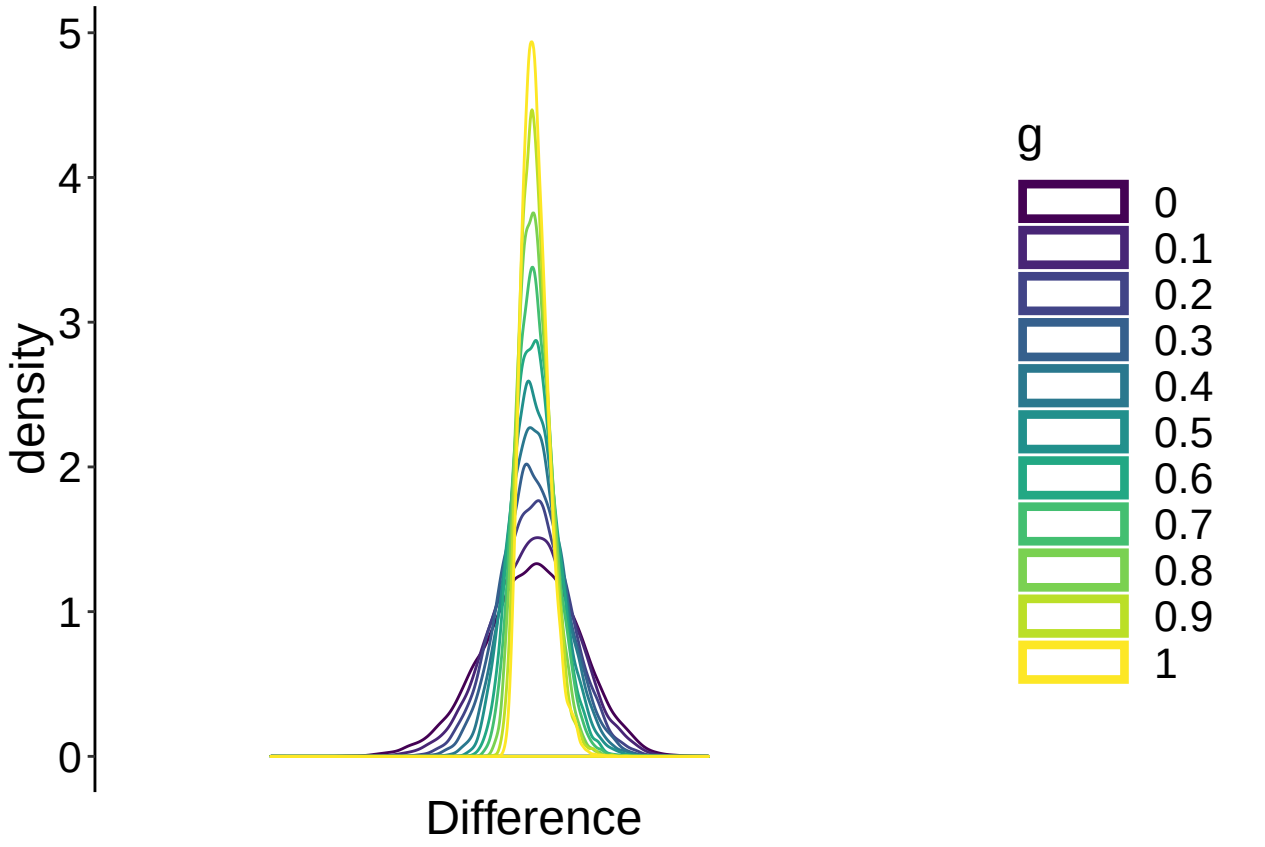


Illustrate sampling distributions: QT8

Sample size = 30, q = 1

```
# fig.width = 10, fig.height = 6
df <- tibble(Difference = as.vector(sim.q.bias[,,1,1]),
             g = factor(rep(gvec, each = nsim))
             )

# make plot
p <- ggplot(df, aes(x = Difference, colour = g)) + theme_classic() +
  geom_density() +
  scale_colour_viridis_d() +
  scale_x_continuous(breaks=nvec, labels = x.labels) +
  # scale_y_continuous(breaks=c(-5,seq(0,50,10))) +
```

```
  coord_cartesian(xlim=c(-2.5,2.5)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 14, 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(0.85,0.65),
        legend.text=element_text(size=16),
        legend.title=element_text(size=18)) +
  # labs(x = "Sample size", y = "Bias in ms") +
  # make thicker legend lines
  guides(colour = guide_legend(override.aes = list(size=3)))
 # + ggtitle("Median RT")
p
```

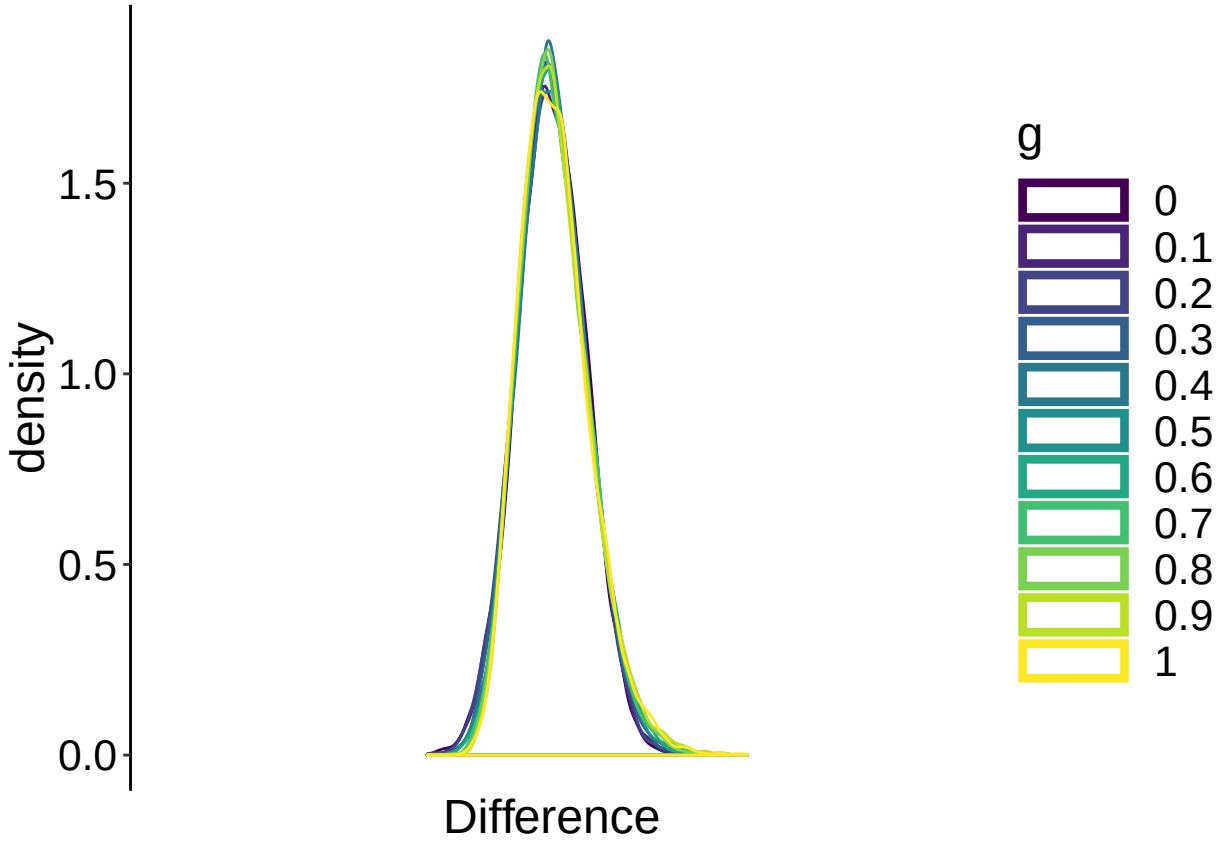



```
# save figure
# ggsave(filename=paste0('figure_miller_bias_md.pdf'),width=10,height=6)
```

Sample size = 30, q = 5

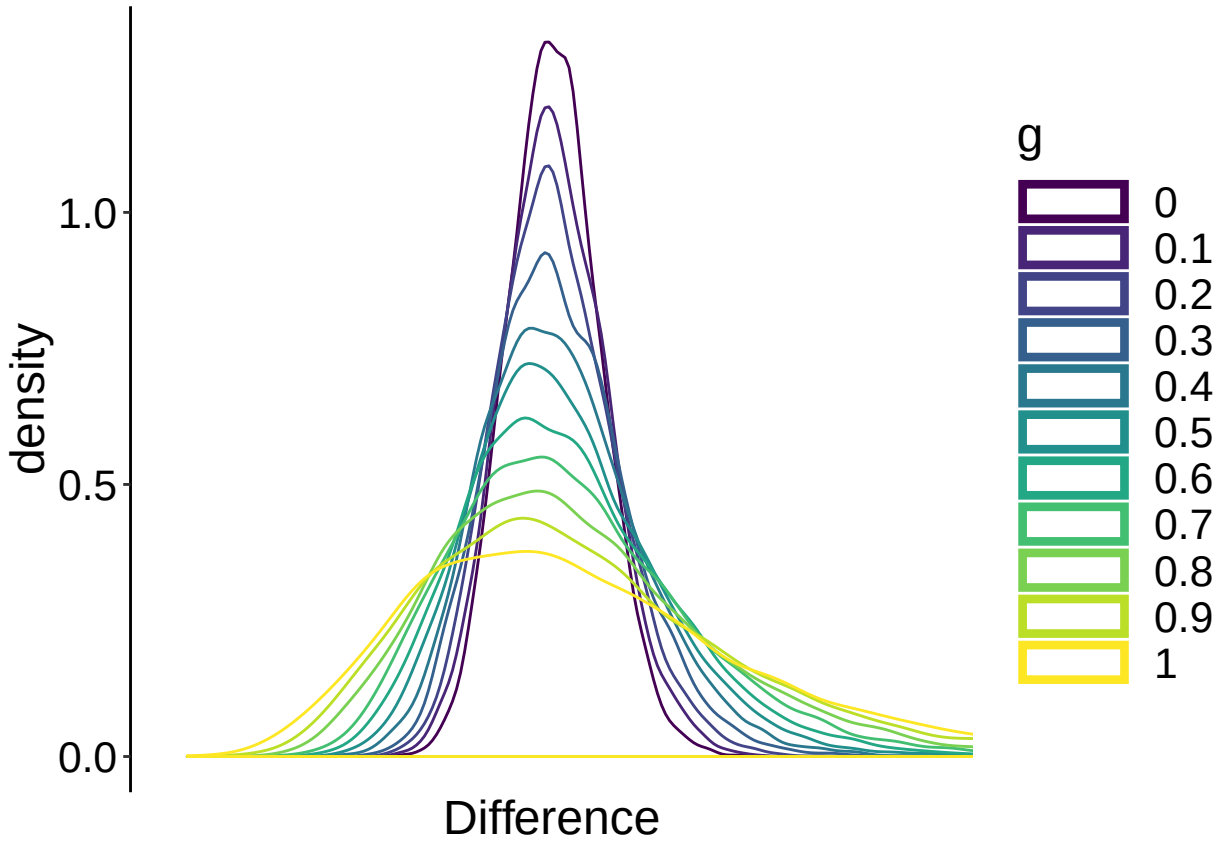
```
# fig.width = 10, fig.height = 6
df <- tibble(Difference = as.vector(sim.q.bias[,,1,5]),
             g = factor(rep(gvec, each = nsim))
             )
```

```
# make plot
p <- ggplot(df, aes(x = Difference, colour = g)) + theme_classic() +
  geom_density() +
  scale_colour_viridis_d() +
  scale_x_continuous(breaks=nvec, labels = x.labels) +
  # scale_y_continuous(breaks=c(-5,seq(0,50,10))) +
  coord_cartesian(xlim=c(-2.5,2.5)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 14, 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(0.85,0.65),
        legend.text=element_text(size=16),
        legend.title=element_text(size=18)) +
  # labs(x = "Sample size", y = "Bias in ms") +
  # make thicker legend lines
  guides(colour = guide_legend(override.aes = list(size=3)))
 # + ggtitle("Median RT")
p
```



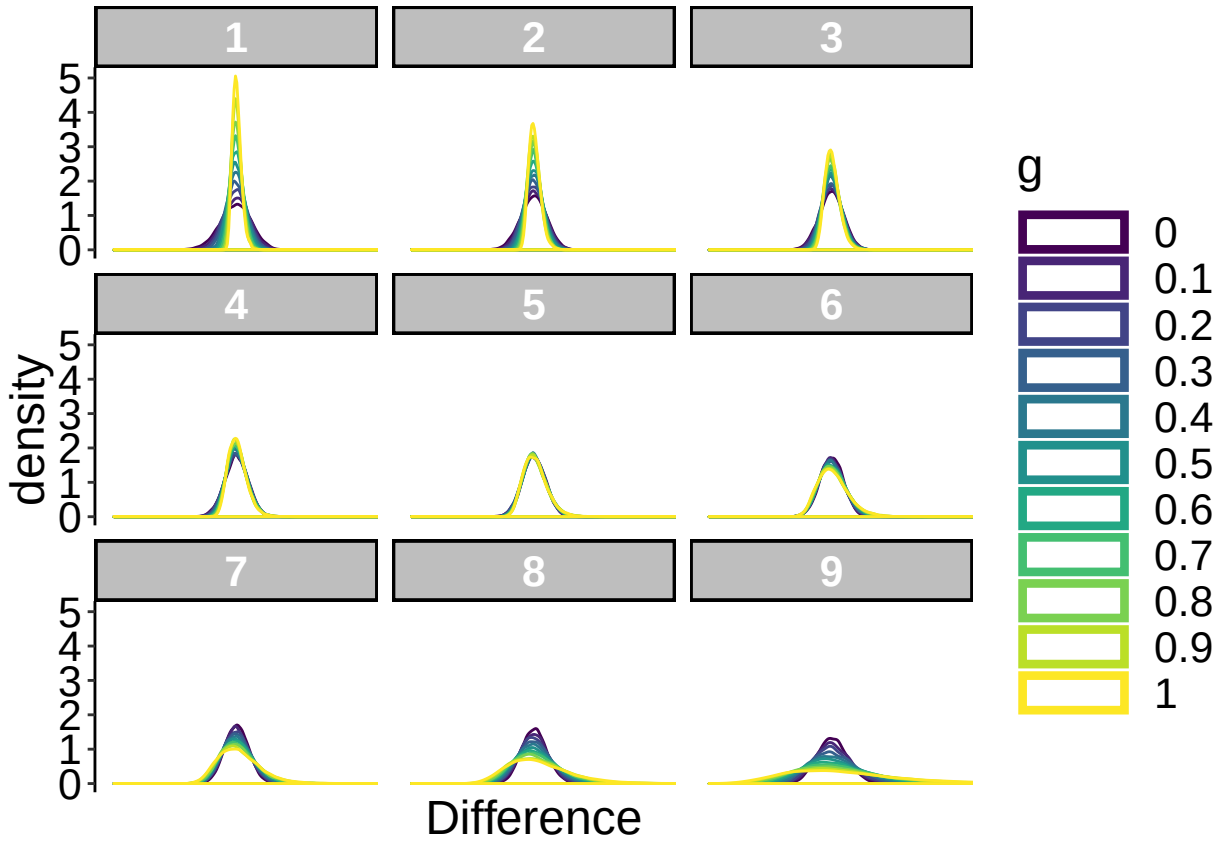
Sample size = 30, q = 9

```r
# fig.width = 10, fig.height = 6
df <- tibble(Difference = as.vector(sim.q.bias[,,1,9]),
             g = factor(rep(gvec, each = nsim))
             )

# make plot
p <- ggplot(df, aes(x = Difference, colour = g)) + theme_classic() +
  geom_density() +
  scale_colour_viridis_d() +
  scale_x_continuous(breaks=nvec, labels = x.labels) +
  # scale_y_continuous(breaks=c(-5,seq(0,50,10))) +
  coord_cartesian(xlim=c(-2.5,2.5)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 14, 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(0.85,0.65),
        legend.text=element_text(size=16),
        legend.title=element_text(size=18)) +
  # labs(x = "Sample size", y = "Bias in ms") +
  # make thicker legend lines
  guides(colour = guide_legend(override.aes = list(size=3)))
 # + ggtitle("Median RT")
p
```



Sample size = 30

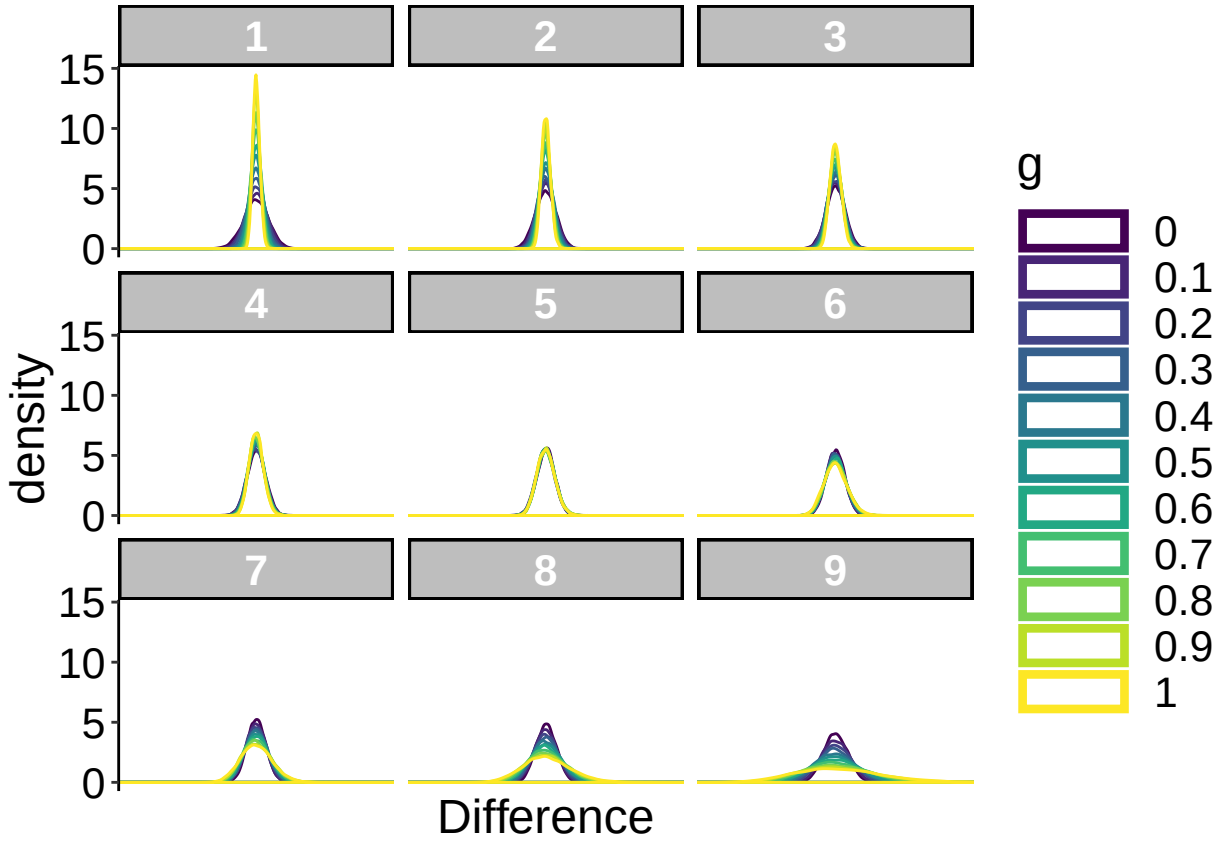
```
# fig.width = 10, fig.height = 6
df <- tibble(Difference = as.vector(sim.q.bias[,,1,]),
             g = factor(rep(rep(gvec, each = nsim),9)),
             q = factor(rep(seq(1,9), each = nsim*length(gvec)))
             )

# make plot
p <- ggplot(df, aes(x = Difference, colour = g)) + theme_classic() +
  geom_density() +
  scale_colour_viridis_d() +
  scale_x_continuous(breaks=nvec, labels = x.labels) +
  # scale_y_continuous(breaks=c(-5,seq(0,50,10))) +
  coord_cartesian(xlim=c(-2.5,2.5)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 14, 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(0.85,0.65),
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.background=element_rect(fill="grey", colour="black"),
        strip.text=element_text(size=16, colour="white", face="bold")) +
  # labs(x = "Sample size", y = "Bias in ms") +
  # make thicker legend lines
  guides(colour = guide_legend(override.aes = list(size=3))) +
 facet_wrap( ~ q, ncol = 3)
 # + ggtitle("Median RT")
p
```



Sample size = 300

```
# fig.width = 10, fig.height = 6
df <- tibble(Difference = as.vector(sim.q.bias[,,11,]),
             g = factor(rep(rep(gvec, each = nsim),9)),
             q = factor(rep(seq(1,9), each = nsim*length(gvec)))
             )

# make plot
p <- ggplot(df, aes(x = Difference, colour = g)) + theme_classic() +
  geom_density() +
  scale_colour_viridis_d() +
  scale_x_continuous(breaks=nvec, labels = x.labels) +
  # scale_y_continuous(breaks=c(-5,seq(0,50,10))) +
  coord_cartesian(xlim=c(-1,1)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 14, 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(0.85,0.65),
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.background=element_rect(fill="grey", colour="black"),
```

```
        strip.text=element_text(size=16, colour="white", face="bold")) +
  # labs(x = "Sample size", y = "Bias in ms") +
  # make thicker legend lines
  guides(colour = guide_legend(override.aes = list(size=3))) +
 facet_wrap( ~ q, ncol = 3)
 # + ggtitle("Median RT")
p
```



Ex-Gaussian distributions

Define Miller's ex-Gaussian parameters

```
load('./data/miller_exg_param.RData')
```

Population HD

```
round(pop.hd)
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
##  [1,]   -1   -1   -1    0    0    0    1    1    1
##  [2,]   -1   -1   -1    0    0    0    1    1    1
##  [3,]   -1   -1    0    0    0    0    1    1    1
##  [4,]   -1   -1    0    0    0    0    1    1    2
```

```
##  [5,]   -1   -1    0    0    0    0    1    1    2
##  [6,]   -1   -1    0    0    0    0    1    1    2
##  [7,]   -1   -1    0    0    0    0    1    1    2
##  [8,]   -1   -1    0    0    0    0    1    1    2
##  [9,]   -1   -1    0    0    0    0    1    1    2
## [10,]   -1   -1    0    0    0    0    1    1    2
## [11,]   -1   -1    0    0    0    0    1    1    3
```

Population deciles

```
round(pop.q)
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
##  [1,]   -1   -1   -1    0    0    0    1    1    1
##  [2,]   -1   -1   -1    0    0    0    1    1    1
##  [3,]   -1   -1    0    0    0    0    1    1    1
##  [4,]   -1   -1    0    0    0    0    1    1    2
##  [5,]   -1   -1    0    0    0    0    1    1    2
##  [6,]   -1   -1    0    0    0    0    1    1    2
##  [7,]   -1   -1    0    0    0    0    1    1    2
##  [8,]   -1   -1    0    0    0    0    1    1    2
##  [9,]   -1   -1    0    0    0    0    1    1    2
## [10,]   -1   -1    0    0    0    0    1    1    2
## [11,]   -1   -1    0    0    0    0    1    1    3
```

Population skewness

```
pop.skew
```

```
##               [,1]
##  [1,] -0.001221342
##  [2,]  0.301837185
##  [3,]  0.613084633
##  [4,]  0.951308319
##  [5,]  1.320886192
##  [6,]  1.762127029
##  [7,]  2.289295169
##  [8,]  2.889302163
##  [9,]  3.633011501
## [10,]  4.624178867
## [11,]  6.688973298
```

Population kurtosis

```
pop.kurt
```

```
##            [,1]
##  [1,]   3.007439
##  [2,]   3.171351
##  [3,]   3.682597
##  [4,]   4.653845
```

```
##  [5,]   6.185309
##  [6,]   8.920932
##  [7,]  13.455927
##  [8,]  20.754219
##  [9,]  31.009691
## [10,]  52.477324
## [11,] 144.376057
```

Simulation

```
nvec <- c(seq(30,100,10), 150, 200, 300) # vector of sample sizes to test
nsim <- 10000 # simulation samples
nboot <- 200 # samples for bias correction
prob.seq <- seq(0.1, 0.9, 0.1) # input to quantile function
q.type <- 8 # quantile type input - type 8 recommended by Hyndman and Fan (1996) - quantile estimates a

# declare matrices of results - save all iterations
sim.hd.bias <- array(NA, dim = c(nsim, nP, length(nvec), 9))
sim.hd.bc <- array(NA, dim = c(nsim, nP, length(nvec), 9))
boot.hd <- array(NA, dim = c(nboot, 9))
sim.q.bias <- array(NA, dim = c(nsim, nP, length(nvec), 9))
sim.q.bc <- array(NA, dim = c(nsim, nP, length(nvec), 9))
boot.q <- array(NA, dim = c(nboot, 9))

set.seed(21)
beep(3)

for(S in 1:nsim){ # simulation iterations

  if(S == 1){beep(2)}
  if(S %% 500 == 0){
    print(paste0("HD bias: simulation ",S," out of ",nsim,"..."))
    beep(2)
  }

  for(P in 1:nP){ # ex-Gaussian parameters

    mu <- miller.param[P,1]
    sigma <- miller.param[P,2]
    tau <- miller.param[P,3]
    large.sample <- rexgauss(max(nvec), mu = mu, sigma = sigma, tau = tau)

    for(N in 1:length(nvec)){ # sample sizes

      current.sample <- large.sample[1:nvec[N]]
      sim.hd.bias[S,P,N,] <- hdseq(current.sample)
      sim.q.bias[S,P,N,] <- quantile(current.sample, probs = prob.seq, type = 8)

      for(b in 1:nboot){
        boot.hd[b,] <- hdseq(sample(current.sample, nvec[N], replace = TRUE))
        boot.q[b,] <- quantile(sample(current.sample, nvec[N], replace = TRUE),
                               probs = prob.seq, type = 8)
```

```
      }
      sim.hd.bc[S,P,N,] <- 2*sim.hd.bias[S,P,N,] - apply(boot.hd, 2, mean) - pop.q[P,]
      sim.q.bc[S,P,N,] <- 2*sim.q.bias[S,P,N,] - apply(boot.q, 2, mean) - pop.q[P,]

      sim.hd.bias[S,P,N,] <- sim.hd.bias[S,P,N,] - pop.q[P,]
      sim.q.bias[S,P,N,] <- sim.q.bias[S,P,N,] - pop.q[P,]
    }
  }
}
save(
  sim.hd.bias,
  sim.hd.bc,
  sim.q.bias,
  sim.q.bc,
  gvec,
  nvec,
  nsim,
  file=paste0('./data/sim_hd_bias_exg.RData'))
beep(8)
```

Compute bias

Bias = average of sample estimates minus population value (already subtracted during simulation).

```
load('./data/sim_hd_bias_exg.RData')
bias.hd <- apply(sim.hd.bias, c(2,3,4), mean, na.rm = TRUE)
bias.hd.md <- apply(sim.hd.bias, c(2,3,4), median, na.rm = TRUE)
bias.hd.bc <- apply(sim.hd.bc, c(2,3,4), mean, na.rm = TRUE)

bias.q <- apply(sim.q.bias, c(2,3,4), mean, na.rm = TRUE)
bias.q.md <- apply(sim.q.bias, c(2,3,4), median, na.rm = TRUE)
bias.q.bc <- apply(sim.q.bc, c(2,3,4), mean, na.rm = TRUE)

x.labels <- c("30","","50","","70","","90","", "150", "200", "300")
```

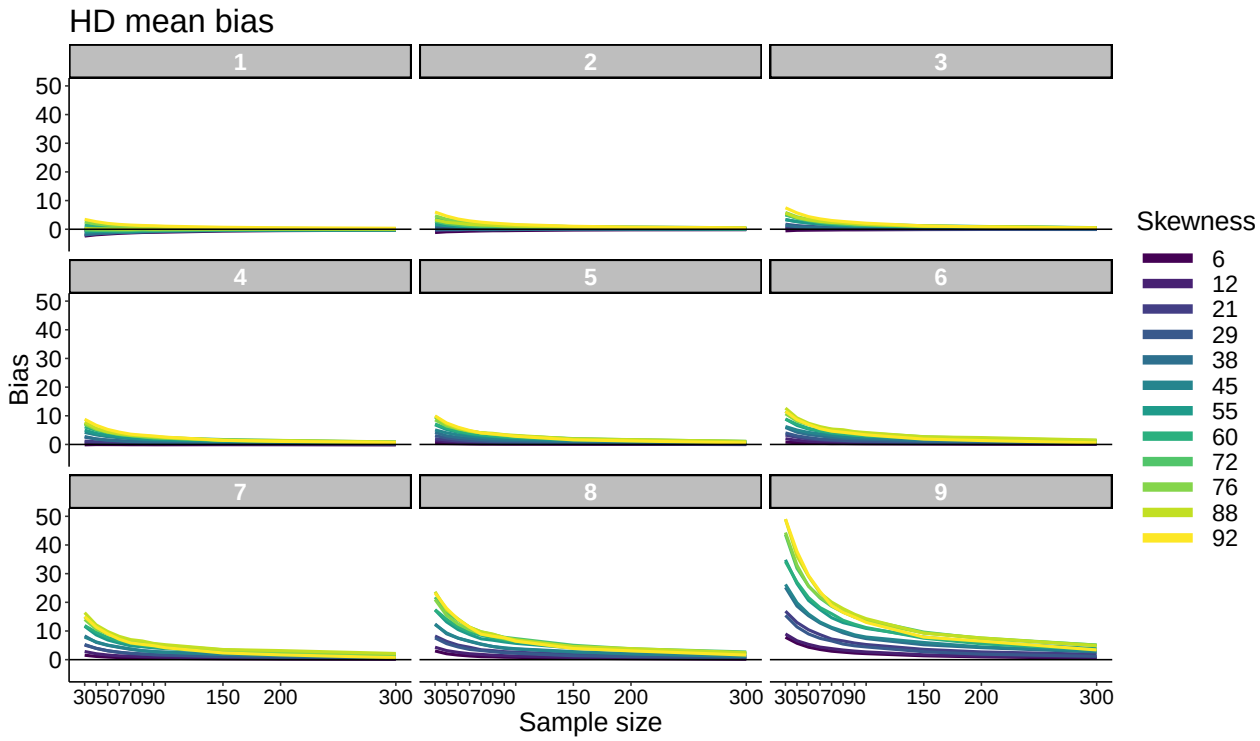
Illustrate bias results

Make function

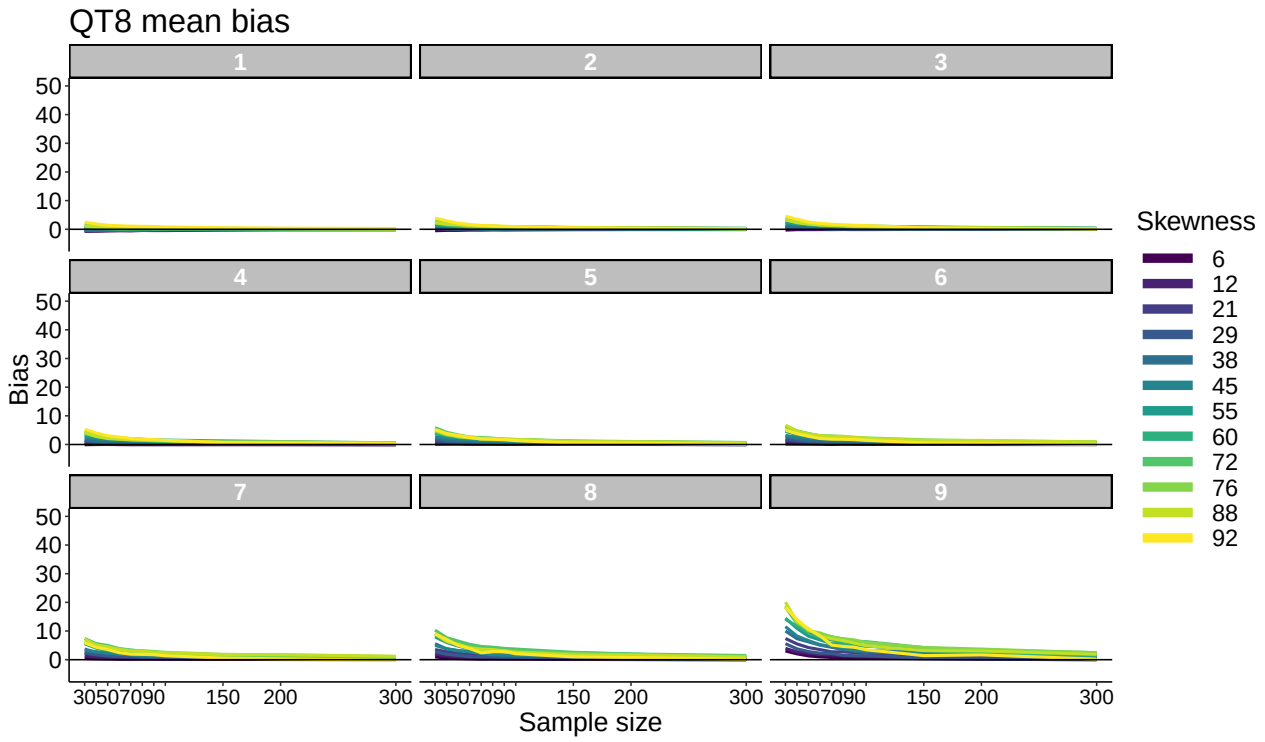
```
plot_bias_res <- function(data, pop.m, pop.md, nvec, nP, x.labels){
  df <- tibble(Bias = as.vector(data),
               Skewness = factor(rep(round(pop.m-pop.md), length(nvec)*9)),
               Size = rep(rep(nvec,each=nP), 9),
               q = factor(rep(seq(1,9), each=nP*length(nvec))))

# make plot
p <- ggplot(df) + theme_classic() +
  geom_line(aes(x=Size, y=Bias, colour = Skewness), size = 1) +
  geom_abline(intercept=0, slope=0, colour="black") +
  scale_colour_viridis_d() +
  scale_x_continuous(breaks=nvec, labels = x.labels) +
```

```
  scale_y_continuous(breaks=seq(0,50,10)) +
  coord_cartesian(ylim=c(-5,50)) +
  theme(plot.title = element_text(size=22),
        axis.title.x = element_text(size = 18),
        axis.text.x = element_text(size = 14, 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",
        # legend.position = c(0.55,0.85),
        legend.direction = "vertical",
        legend.text=element_text(size=16),
        legend.title=element_text(size=18),
        strip.background=element_rect(fill="grey", colour="black"),
        strip.text=element_text(size=16, colour="white", face="bold")) +
  labs(x = "Sample size", y = "Bias") +
  # make thicker legend lines
  guides(colour = guide_legend(override.aes = list(size=3))) +
  facet_wrap( ~ q, ncol = 3)
p
}
```

```
p <- plot_bias_res(bias.hd, pop.m, pop.md, nvec, nP, x.labels)
p <- p + ggtitle("HD mean bias")
p
```

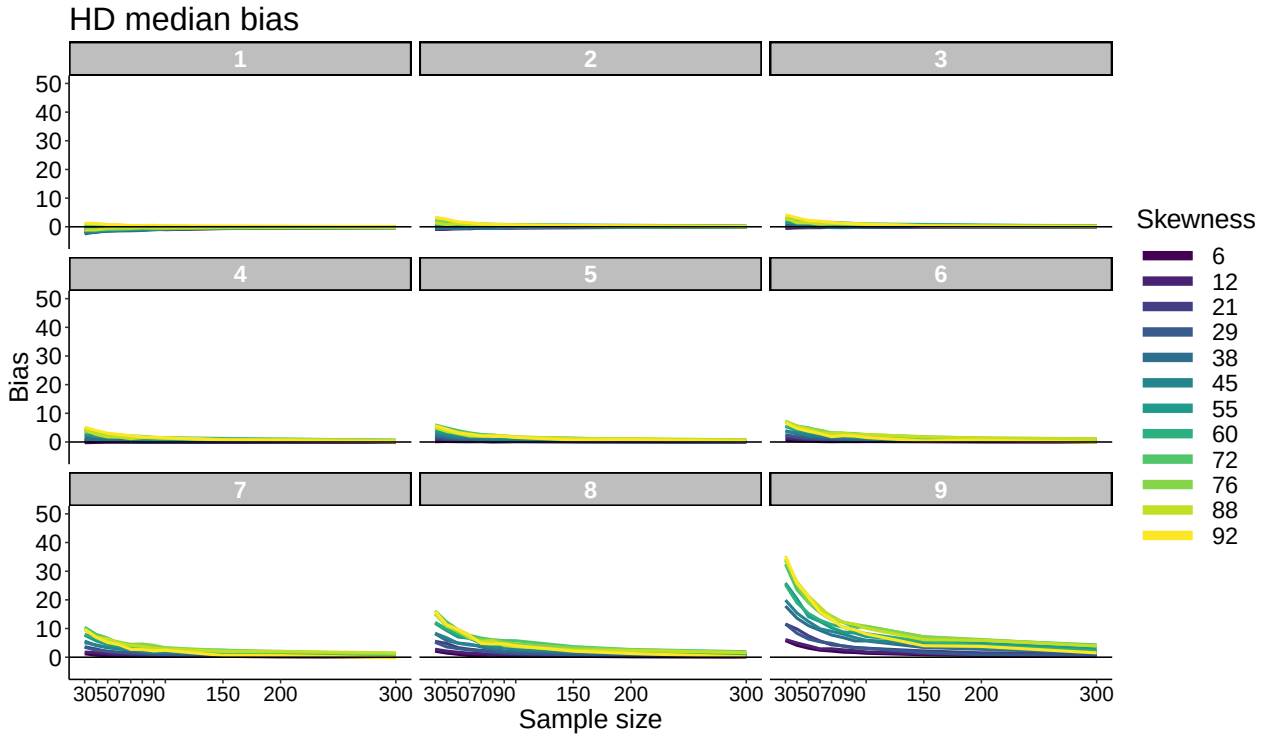



```
p <- plot_bias_res(bias.q, pop.m, pop.md, nvec, nP, x.labels)
p <- p + ggtitle("QT8 mean bias")
p
```

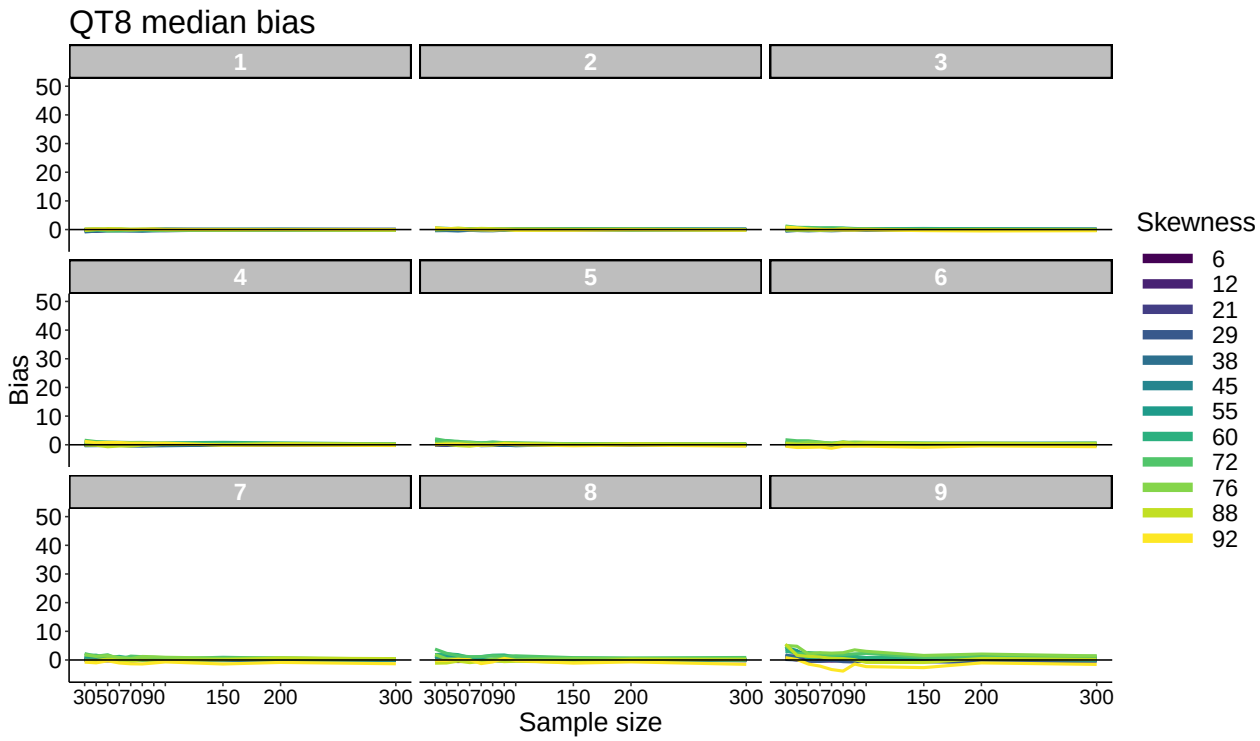


Illustrate median bias results

```
p <- plot_bias_res(bias.hd.md, pop.m, pop.md, nvec, nP, x.labels)
p <- p + ggtitle("HD median bias")
p
```

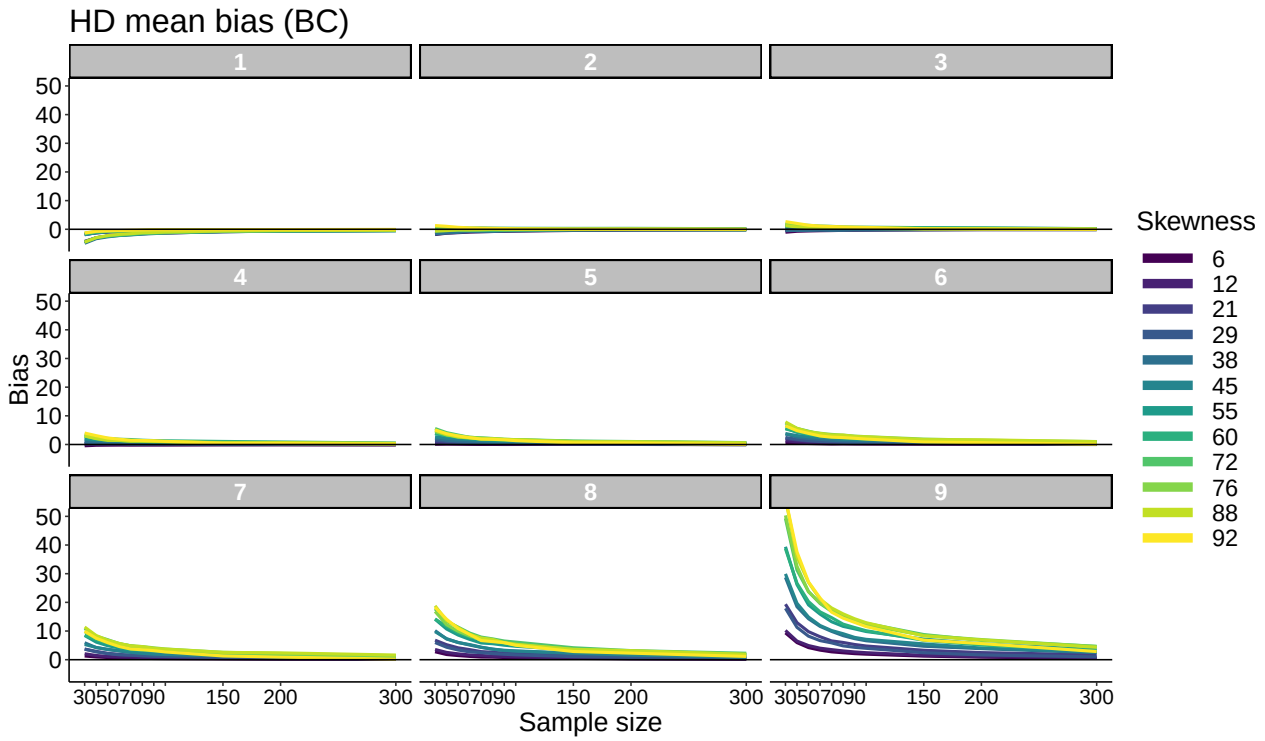


```
p <- plot_bias_res(bias.q.md, pop.m, pop.md, nvec, nP, x.labels)
p <- p + ggtitle("QT8 median bias")
p
```



Illustrate bias corrected bias results

```
p <- plot_bias_res(bias.hd.bc, pop.m, pop.md, nvec, nP, x.labels)
p <- p + ggtitle("HD mean bias (BC)")
p
```




```
p <- plot_bias_res(bias.q.bc, pop.m, pop.md, nvec, nP, x.labels)
p <- p + ggtitle("QT8 mean bias (BC)")
p
```