Z-Score vs ASH

```
set.seed(1)
```

Alternative data set, generated by Poisson distribution

We simulate data such as $c_{ij} \sim Poisson(\mu_{ij})$, with μ_{ij} (expression level) like the following table.

1	2
2	1
10	20
20	10
$\vdots$	$\vdots$
10^7	2×10^7
2×10^7	10^7

Each small block is 100×50. In other words, for each expression level, we have 1000 genes and 50 samples for case and control each.

With the simulated count matrix and the condition vector, we could get a $\hat{\beta}$ and $\hat{s}$, as well as a z-score $= \hat{\beta}/\hat{s}$ for each gene by both OLS and voom + limma pipeline.

```
source("../code/fit_method.R")

    ## effect size and standard error estimated by OLS
    log_counts = log2(counts + 1)
    ols_fit <- get_ols(log_counts = log_counts, condition = condition)
    betahat_ols = ols_fit$betahat
    sebetahat_ols = ols_fit$sebetahat
    df_ols = ols_fit$df
    z_ols = betahat_ols / sebetahat_ols

    ## effect size and measurement error estimated by voom + limma
    voom_fit = voom_transform(counts, condition)
    betahat_voom = voom_fit$betahat
    sebetahat_voom = voom_fit$sebetahat
    df_voom = voom_fit$df
    z_voom = betahat_voom / sebetahat_voom
```

We group the genes to 16 bins of 1000 genes. Then we plot $\hat{s}$ and z-score with respect to the expression level.

```
        ## grouping genes into K subgroups from low to high expression
  group_id = rep(1:(2 * length(mu)), each = Ngene)

  boxplot(abs(betahat_ols) ~ group_id)
  boxplot(sebetahat_ols ~ group_id, xlab = "expression level", ylab = "sebetahat", main = "OLS")
  boxplot(log(abs(z_ols)) ~ group_id, xlab = "expression level", ylab = "log |z-score|", main = "OLS")
  abline(log(qnorm(0.975)), 0, lty = 3, col = "red")
```

```
  boxplot(abs(betahat_voom) ~ group_id)
  boxplot(sebetahat_voom ~ group_id, xlab = "expression level", ylab = "sebetahat", main = "TMM + voom
  boxplot(log(abs(z_voom)) ~ group_id, xlab = "expression level", ylab = "log |z-score|", main = "TMM +
  abline(0, 0, lty = 3, col = "red")
```

Now we use Poisson thinning to put artificial N(0, 1) effect to a count matrix.

```
pois_thinning <- function(counts, log2foldsd) {
        log2foldchanges <- rnorm(dim(counts)[1], mean = 0, sd = log2foldsd)
        foldchanges <- 2 ^ log2foldchanges
        Nsamp = dim(counts)[2]/2

        ## thin group A
        counts[log2foldchanges > 0, 1:Nsamp] <-
            matrix(rbinom(sum(log2foldchanges >
            0) * Nsamp, size = c(as.matrix(counts[log2foldchanges >
            0, 1:Nsamp])), prob = rep(1 / foldchanges[log2foldchanges > 0], Nsamp)),
            ncol = Nsamp)
        ## thin group B
        counts[log2foldchanges < 0, (Nsamp + 1):(2 * Nsamp)] <-
            matrix(rbinom(sum(log2foldchanges <
            0) * Nsamp, size = c(as.matrix(counts[log2foldchanges <
            0, (Nsamp + 1):(2 * Nsamp)])), prob = rep(foldchanges[log2foldchanges <
            0], Nsamp)), ncol = Nsamp)

    return(list(counts = counts, log2foldchanges = log2foldchanges))
}
```

```
Nsample = 20
counts = c(rpois(1000 * Nsample * 2, 1), rpois(1000 * Nsample * 2, 10), rpois(1000 * Nsample * 2, 100),
counts = matrix(counts, ncol = Nsample * 2, byrow = TRUE)
counts = pois_thinning(counts = counts, log2foldsd = 1)$counts
condition = rep(1:2, each = Nsample)
```

```
source("../code/fit_method.R")

    ## effect size and standard error estimated by OLS
    log_counts = log2(counts + 1)
    ols_fit <- get_ols(log_counts = log_counts, condition = condition)
    betahat_ols = ols_fit$betahat
    sebetahat_ols = ols_fit$sebetahat
    df_ols = ols_fit$df
    z_ols = betahat_ols / sebetahat_ols

    ## effect size and measurement error estimated by voom + limma
    voom_fit = voom_transform(counts, condition)
    betahat_voom = voom_fit$betahat
    sebetahat_voom = voom_fit$sebetahat
    df_voom = voom_fit$df
    z_voom = betahat_voom / sebetahat_voom

    ## effect size estimated by ash
    ash_fit_voom = ashr::ash(betahat_voom, sebetahat_voom, method = "shrink", mixcompdist = "normal")
```

```
  beta_est_voom = ash_fit_voom$PosteriorMean
  ash_fit_ols = ashr::ash(betahat_ols, sebetahat_ols, method = "shrink", mixcompdist = "normal")
  beta_est_ols = ash_fit_ols$PosteriorMean
```

```
      ## grouping genes into K subgroups from low to high expression
group_id = rep(1:4, each = 1000)

boxplot(sebetahat_ols ~ group_id, xlab = "expression level", ylab = "sebetahat", main = "OLS")
boxplot(z_ols ~ group_id, xlab = "expression level", ylab = "z-score", main = "OLS")
abline(0, 0, lty = 3, col = "red")

boxplot(sebetahat_voom ~ group_id, xlab = "expression level", ylab = "sebetahat", main = "voom + limma
boxplot(z_voom ~ group_id, xlab = "expression level", ylab = "z-score", main = "voom + limma")
abline(0, 0, lty = 3, col = "red")

boxplot(beta_est_ols ~ group_id, xlab = "expression level", ylab = "estimated effect size", main = "O
boxplot(beta_est_voom ~ group_id, xlab = "expression level", ylab = "estimated effect size", main = "
```