Controlling for false precision using Scale Simulation in ALDEx2

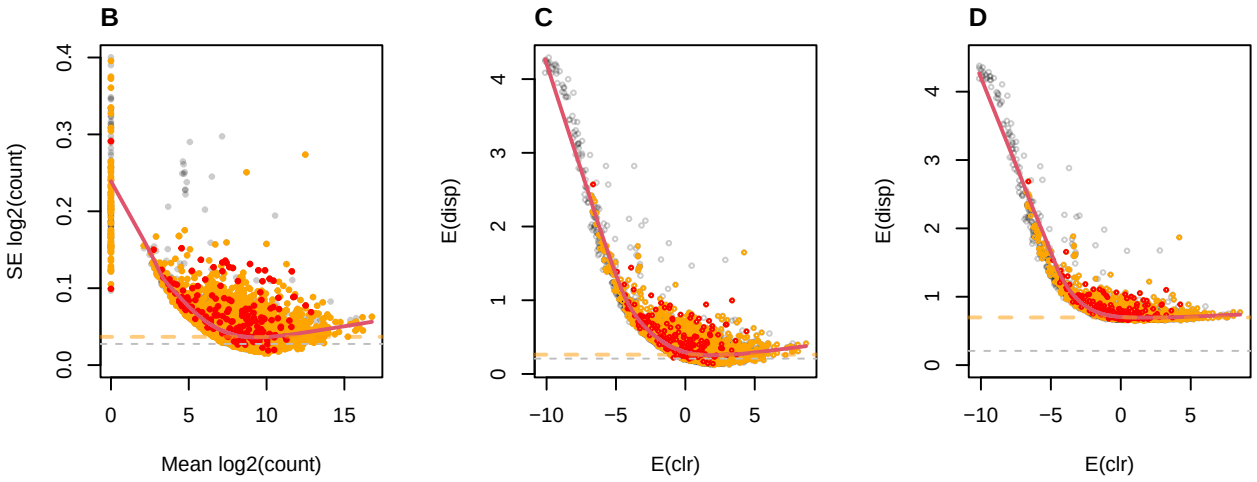
true true true

Abstract

In high-throughput sequencing (HTS) studies, sample-to-sample variation in sequencing depth is driven by technical factors, and not by variation in the scale (e.g., total size, microbial load, or total gene expression) of the underlying biological systems. Typically a statistical normalization is used to remove unwanted technical variation in the data or the parameters of the model to enable analyses that are sensitive to scale; e.g., differential abundance and differential expression analyses. Recently we showed that all normalizations make implicit assumptions about the unmeasured system scale and that errors in these assumptions can lead to dramatic increases in false positive and false negative rates. Here we describe updates to the ALDEx2 R package that mitigate these problems by directly modeling uncertainty in the unmeasured system scale through the use of a *scale model*. Scale models generalize the idea of normalizations and can be thought of as explicitly modeling the error in normalization by examining a distribution over all possible normalizations. Beyond enhancing the robustness of HTS analyses, the use of scale models within ALDEx2 enhances the transparency and reproducibility of analyses by making implicit normalizing assumptions an explicit part of the model building process.

Results

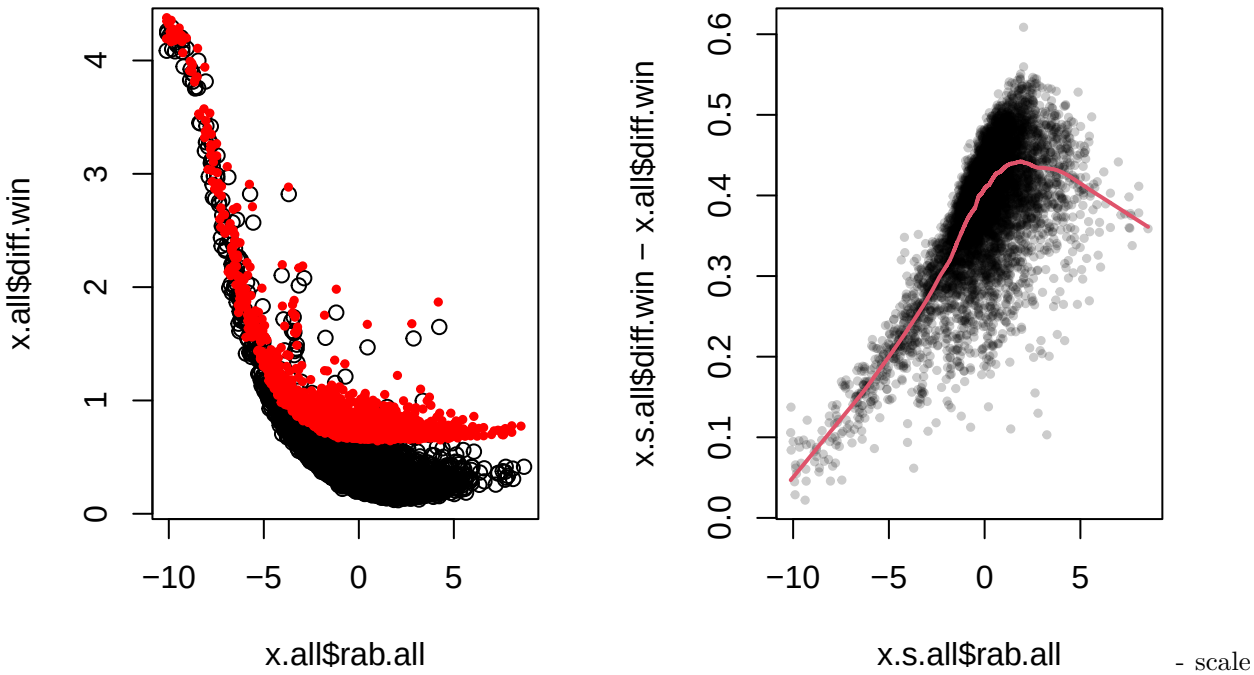
- HTS has a dispersion problem since the log(var(x)) ≠ var(log(x)) (Fig 1)
- this causes false positive inference and the double filtering approach is commonly used to weed out FP, or nuisance positives. However this causes specificity problems since BH corrections are useful only in the total dataset and not when subset
- HTS has a location problem, and common normalizations are assumed to correct for this. However, there is no universal normalization and no standard way to identify when the assumptions of the normalization are broken



There is an anomaly in the dispersion curve, whereby mid-range parts are less dispersed than high-range parts (Anders and Huber 2010, Fig 7).

```
# flattening the dispersion anomaly curve
par(mfrow=c(1,2))
```

```
 plot(x.all$rab.all, x.all$diff.win)
 points(x.s.all$rab.all,x.s.all$diff.win, col='red', pch=19, cex=0.5)

ls.diff <- data.frame(x.s.all$rab.all, x.s.all$diff.win -  x.all$diff.win)

 plot(x.s.all$rab.all, x.s.all$diff.win -  x.all$diff.win, pch=19, col=rgb(0,0,0,0.2), cex=0.5)
lines(lowess(ls.diff, f=.1), col=2, lwd=2)
```



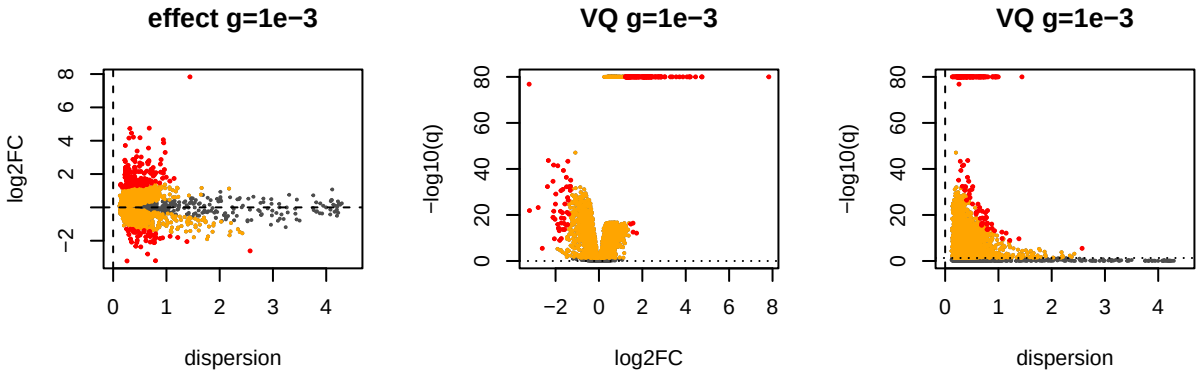
- scale is a universal way of dealing with both of these common problems.

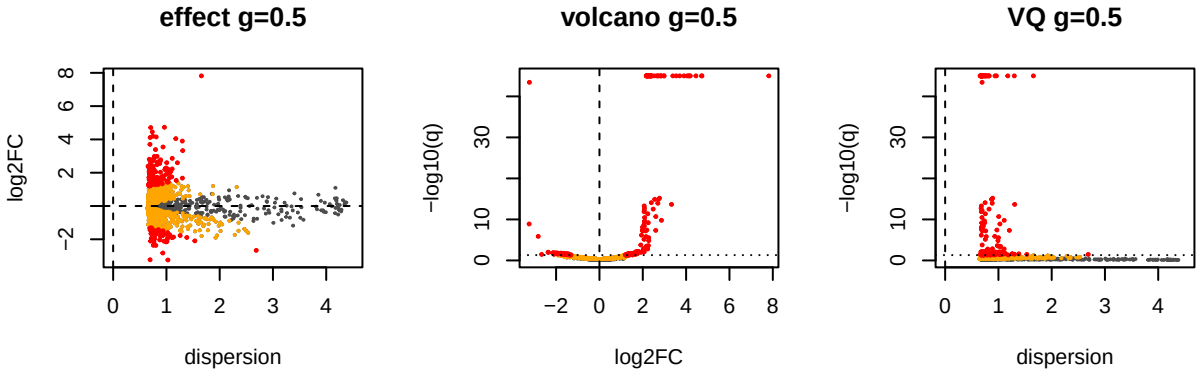
- all normalizations can be written as the denominator of a ratio, and all normalizations make an assumption about the scale of the system.
  - Proportions, and any other constant denominator normalisation implicitly assume that the scale is equal between every sample is constant.
  - The CLR or other log-ratios where the denominator is calculated from the data implicitly assume that the value of the denominator is related to the scale of the data
  - by analogy, the TMM assumes that a subset of parts in one sample can be chosen as the scale reference
  - by analogy, the RLE assumes that a different part in each sample can be chosen as the scale reference

```
# code/yeast-aldex.R
load("analysis/x.all.Rda") # gamma=1e-3
load("analysis/x.s.all.Rda") # gamma=0.5

sig.ald <- x.all$we.eBH < 0.05
sig.s.ald <- x.s.all$we.eBH < 0.05

par(mfrow=c(2,3))
# effect plot
plot(x.all$diff.win, x.all$diff.btw, pch=19, cex=0.2, col="grey30",
  xlab="dispersion", ylab="log2FC", xlim=c(0, 4.5))
points(x.all$diff.win[sig.ald], x.all$diff.btw[sig.ald],
  pch=19, cex=0.2, col="orange")
```

```
points(x.all$diff.win[sig.s.ald], x.all$diff.btw[sig.s.ald],
  pch=19, cex=0.3, col="red")
title(main="effect g=1e-3")
abline(v=0, lty=2)
abline(h=0, lty=2)

# volcano plot
plot(x.all$diff.btw, -log10(x.all$we.eBH+1e-80), pch=19, cex=0.2, col="grey30",
  xlab="log2FC", ylab="-log10(q)")
points(x.all$diff.btw[sig.ald], -log10(x.all$we.eBH+1e-80)[sig.ald],
  pch=19, cex=0.2, col="orange")
points(x.all$diff.btw[sig.s.ald], -log10(x.all$we.eBH+1e-80)[sig.s.ald],
  pch=19, cex=0.3, col="red")
title(main="VQ g=1e-3")
abline(h=0, lty=3)


# Dispersion (variance) Q plot
plot(x.all$diff.win, -log10(x.all$we.eBH+1e-80), pch=19, cex=0.2, col="grey30",
  xlab="dispersion", ylab="-log10(q)", xlim=c(0, 4.5))
points(x.all$diff.win[sig.ald], -log10(x.all$we.eBH+1e-80)[sig.ald],
  pch=19, cex=0.2, col="orange")
points(x.all$diff.win[sig.s.ald], -log10(x.all$we.eBH+1e-80)[sig.s.ald],
  pch=19, cex=0.3, col="red")
title(main="VQ g=1e-3")
abline(v=0, lty=2)
abline(h=-log10(0.05), lty=3)

## MA plot
#plot(x.all$rab.all, x.all$diff.btw, pch=19, cex=0.2, col="grey30",
#  xlab="log2rAB", ylab="log2FC")
#points(x.all$rab.all[sig.ald], x.all$diff.btw[sig.ald],
#  pch=19, cex=0.2, col="orange")
#points(x.all$rab.all[sig.s.ald], x.all$diff.btw[sig.s.ald],
#  pch=19, cex=0.3, col="red")
#title(main="MA g=1e-3")
#abline(h=0, lty=2)

# effect plot
plot(x.s.all$diff.win, x.s.all$diff.btw, pch=19, cex=0.2, col="grey30",
  xlab="dispersion", ylab="log2FC", xlim=c(0, 4.5))
points(x.s.all$diff.win[sig.ald], x.s.all$diff.btw[sig.ald],
  pch=19, cex=0.2, col="orange")
points(x.s.all$diff.win[sig.s.ald], x.s.all$diff.btw[sig.s.ald],
  pch=19, cex=0.3, col="red")
title(main="effect g=0.5")
abline(v=0, lty=2)
abline(h=0, lty=2)

# volcano plot
plot(x.s.all$diff.btw, -log10(x.s.all$we.eBH+1e-45), pch=19, cex=0.2, col="grey30",
  xlab="log2FC", ylab="-log10(q)")
points(x.s.all$diff.btw[sig.ald], -log10(x.s.all$we.eBH+1e-45)[sig.ald],
```

```
  pch=19, cex=0.2, col="orange")
points(x.s.all$diff.btw[sig.s.ald], -log10(x.s.all$we.eBH+1e-45)[sig.s.ald],
  pch=19, cex=0.3, col="red")
title(main="volcano g=0.5")
abline(v=0, lty=2)
abline(h=-log10(0.05), lty=3)

# Dispersion (variance) Q plot
plot(x.s.all$diff.win, -log10(x.s.all$we.eBH+1e-45), pch=19, cex=0.2, col="grey30",
  xlab="dispersion", ylab="-log10(q)", xlim=c(0, 4.5))
points(x.s.all$diff.win[sig.ald], -log10(x.s.all$we.eBH+1e-45)[sig.ald],
  pch=19, cex=0.2, col="orange")
points(x.s.all$diff.win[sig.s.ald], -log10(x.s.all$we.eBH+1e-45)[sig.s.ald],
  pch=19, cex=0.3, col="red")
title(main="VQ g=0.5")
abline(v=0, lty=2)
abline(h=-log10(0.05), lty=3)
```






```
## MA plot
#plot(x.s.all$rab.all, x.s.all$diff.btw, pch=19, cex=0.2, col="grey30",
#  xlab="log2rAB", ylab="log2FC")
#points(x.s.all$rab.all[sig.ald], x.s.all$diff.btw[sig.ald],
#  pch=19, cex=0.2, col="orange")
#points(x.s.all$rab.all[sig.s.ald], x.s.all$diff.btw[sig.s.ald],
#  pch=19, cex=0.3, col="red")
#title(main="MA g=0.5")
#abline(h=0, lty=2)
```

Anders, Simon, and Wolfgang Huber. 2010. "Differential Expression Analysis for Sequence Count Data." *Genome Biol* 11 (10): R106. https://doi.org/10.1186/gb-2010-11-10-r106.