What is a 0 anyway

gg

2018-04-12

To run this file: Rscript -e "rmarkdown::render('zero.Rmd')"

0 is an observation, not a ground truth

We will explore what a 0, or any other value for that matter means in a probabilistic sense.

We take the dataset, which has 7 technical replicates of the yeast dataset and compare one technical replicate with the other.

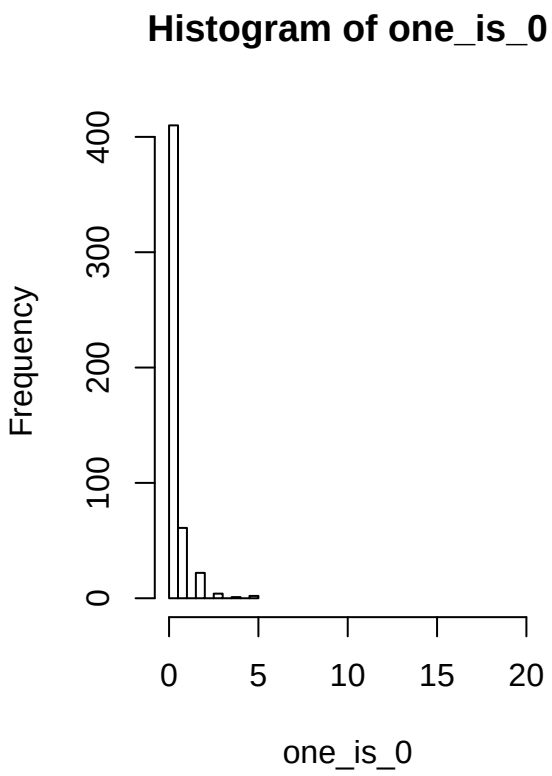
```
d <- as.matrix(read.table("data/countfinal2.tsv", header=T, row.names=1))

# remove 0 sum features
d.n0 <- d[rowSums(d) > 0,]
```

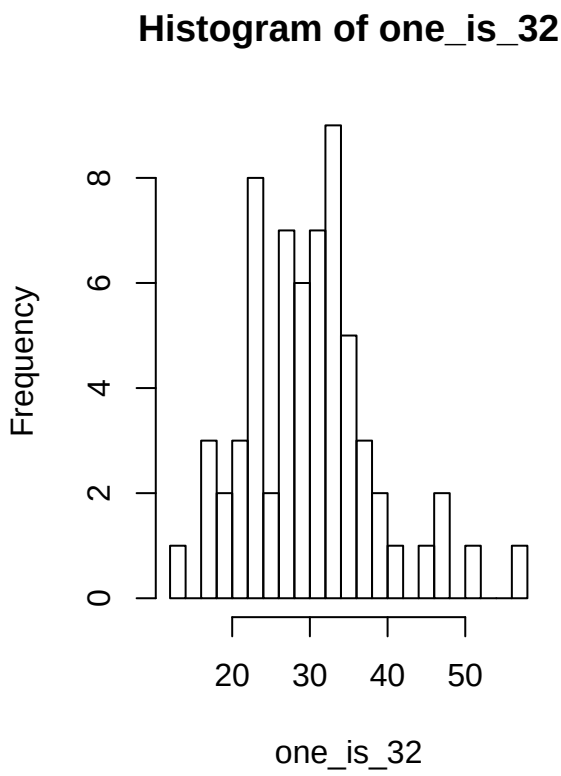
At this point we have removed all features that are 0 in all samples. These features are almost certainly not 0 if we sequence 10X more deeply, but they are of no consequence for the present analysis.

In the context of RNA-seq, 0 is an anomaly since most genes have a low level of stochastic expression. In the context of single cell seq, a 0 can be a 0 because expression is somewhat binary in single cells, but averaged over a population RNA expression is continuous.

We can now examine technical replicates and see what a 0 means in the context of a near perfect replicate. What happens if one replicate is 0?

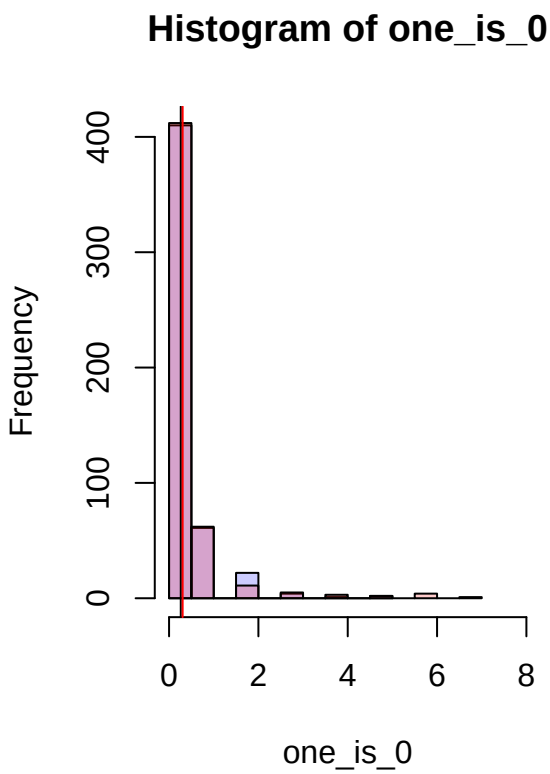
```
# choose the features of replicate 2 where the value of replicate
# 1 is 0, adn so on
one_is_0 <- d.n0[,2][d.n0[,1] == 0]
one_is_1 <- d.n0[,2][d.n0[,1] == 1]
one_is_2 <- d.n0[,2][d.n0[,1] == 2]
one_is_32 <- d.n0[,2][d.n0[,1] == 32]

# plot the distribution of values from replicate 2 where replicate 1
# has a value of 0 or a value of 32
par(mfrow=c(1,2))
hist(one_is_0, breaks=9, xlim=c(0,20))
hist(one_is_32, breaks=20)
```

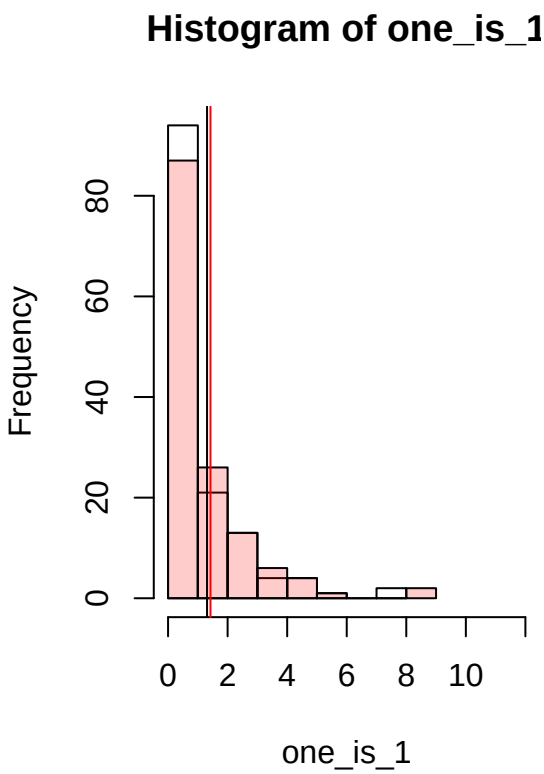




Just for completeness, we can choose a second replicate to compare. Here the the first technical replicate is in blue, the second technical replicate is in pink, and the expected value of 0 in the technical replicate is calculated as the mean of the replicate distribution. In this dataset, E(0) is approximately 0.3. Note that the most likely value for features with a count of 1 in the first replicate is 0, but the expected value of these features is about 1.4.

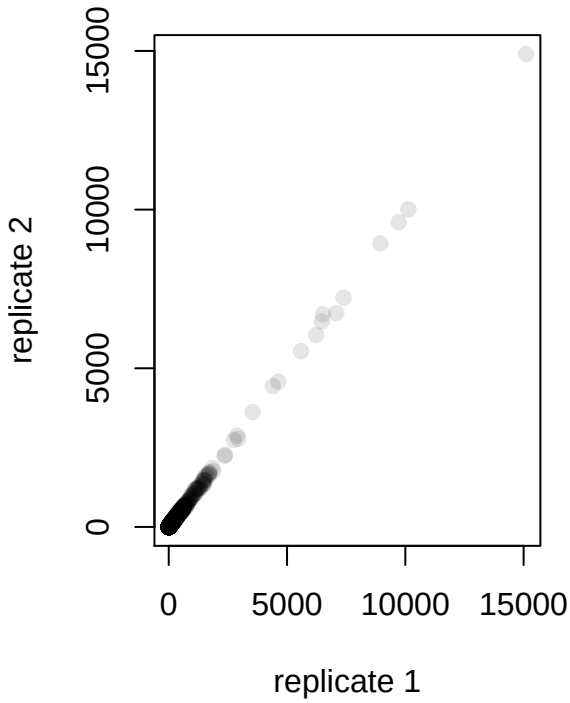
```
# compare replicates
one_is_0b <- d.n0[,3][d.n0[,1] == 0]
one_is_1b <- d.n0[,3][d.n0[,1] == 1]
one_is_2b <- d.n0[,3][d.n0[,1] == 2]
one_is_32b <- d.n0[,3][d.n0[,1] == 32]

par(mfrow=c(1,2))
hist(one_is_0, breaks=9, xlim=c(0,8), col=rgb(0,0,1,0.2))
hist(one_is_0b, breaks=15, add=T, col=rgb(1,0,0,0.2))
abline(v=mean(one_is_0)) # the expected value of 0 for pair
abline(v=mean(one_is_0b), col="red")

hist(one_is_1, breaks=9, xlim=c(0,12))
hist(one_is_1b, breaks=9, add=T, col=rgb(1,0,0,0.2))
abline(v=mean(one_is_1))
abline(v=mean(one_is_1b), col="red")
```

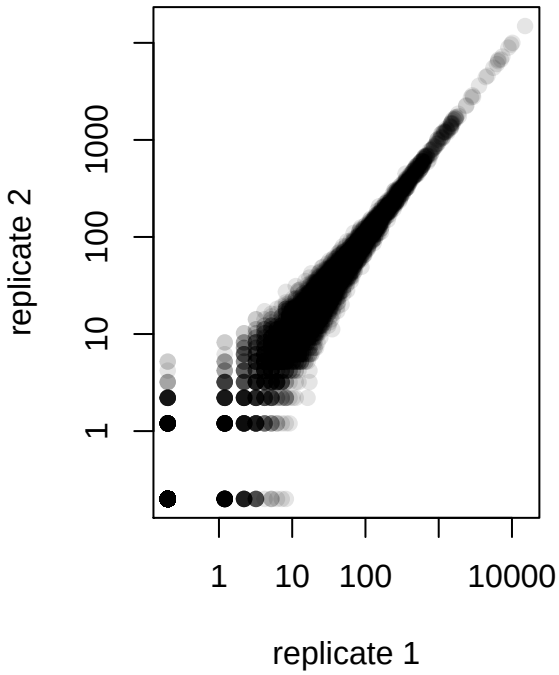




We can generate a scatter plot all vs all for technical replicate 1 vs technical replicate 2 in either Euclidian or log-log space. Note where the imprecision is - at the low count margin.

```
par(mfrow=c(1,2))
plot(d.n0[,1] + 0.2, d.n0[,2] + 0.2, pch=19, col=rgb(0,0,0,0.1), xlab="replicate 1", ylab="replicate 2"
plot(d[,1] +0.2, d[,2] +0.2, log="xy", pch=19, col=rgb(0,0,0,0.1), xlab="replicate 1", ylab="replicate
```





In RNA-seq we typically do not have enough samples to have independent power for each feature. We really

need about 10 samples for every feature we want to examine independently.

The data are discrete probabilities of observing the feature in the DNA sequencing library scaled by the read count; i.e., converted to integers by 'counting'. If we sequenced 10X deeper, then 0 values would be converted to a integers between 0 and ~10 (sampling). If we sequenced 1000X deeper, then we would get an even better estimate of the actual underlying value for our 0s. Most of the 0s at some point will be converted to a count—so what is an appropriate value for 0? Somewhere between 0 (can never occur in the observable universe) and 1 (we always should see it) is a least likely value that, over many experiments will be least likely to perturb the system? That value is 0.5. Not always the best, usually not the optimum, but in general the least wrong.

I typically choose either of two methods for zero replacement/estimation: The first is to use the zCompositions package to replace 0 values with a count zero multiplicative estimate. This approach replaces 0 values with a suitable pseudocount, but keeps the relationships between all the other features constant. The pseudocount only replaces the 0 value, not the rest. This approach is useful for exploratory data analysis, such as compositional biplots, clustering, or using the propr package with point estimates. The second approach is instantiated in the ALDEx2 package, where we assume that the prior probability of 0 is 0.5. In the prior approach we are adding 0.5 to *every* value in the matrix. On the surface this seems to be undesirable. However, note that the expected value of low count features in this dataset is actually larger than the observed value: recall that it is about 0.3 for a count of 0, about 1.3 for a count of 1, about 2.7 for a count of 2, and about 31 for a count of 32. Thus, adjusting each value by a constant begins to make more sense.

If we recall that the basis of log-ratio analysis is the ratio between components, then adding a prior probability to each feature starts to make sense. As shown in Table 1, adding the prior is a much better estimate of the ratios of the Expected values than is the ratio of the point observations. The addition of the prior is in fact an even better estimate than is the 0 replacement method in this dataset.

Table 1: Count vs E(count) and the ratio relationships. The table shows the ratio between counts, and the expected value of the count in another technical replicate (E ratio), compared to the ratio of the count + a prior of 0.5 (P ratio).

Count 1	Count 2	E(count) 1	E(count) 2	Count ratio	E ratio	P ratio
0	1	0.3	1.3	0/1	0.3/1.3	0.5/1.5
0	2	0.3	2.7	0/2	0.3/2.7	0.5/2.5
0	32	0.3	31	0/32	0.3/31	0.5/32.5

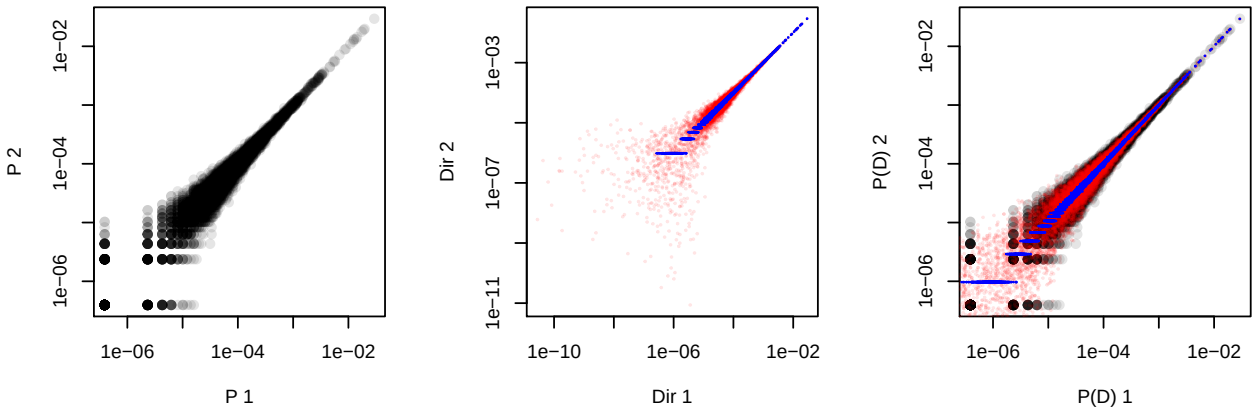
Lets generate random instances of the data. The maximum likelihood probability is the count divided by the sum of the sample. This is a discrete probability since we are dealing with finite values.

Since we know that a pure replicate is not identical and differs by sampling variation the ML estimate is strongly affected by sampling. We can convert the discrete estimate of the probability to a continuous probability by modelling as a multivariate Poisson process with a fixed sum: this is a Dirichlet distribution.

Even though the E(Dir) is very close to the ML estimate, the individual Dir instances are variable. This variation is a reasonable match for the actual underlying technical variation (tails are a bit broader in general), but is reasonable.

```
par(mfrow=c(1,3))
plot((d.n0[,1] + 0.2) /sum(d.n0[,1] + 0.2),
    (d.n0[,2] + 0.2)/sum(d.n0[,2] + 0.2),
    log="xy", xlab="P 1", ylab="P 2",
    pch=19, col=rgb(0,0,0,0.1))


d.dir <- rdirichlet(16, d.n0[,1]+0.5)
```

```
ml <- (d.n0[,1]+0.5) / sum(d.n0[,1]+0.5)
E_dir <- apply(d.dir, 2, mean)
plot(d.dir[1,], d.dir[2,], pch=19, cex=0.2,xlab="Dir 1",
    ylab="Dir 2", col=rgb(1,0,0,0.1), log="xy")
points(E_dir, ml, pch=19,cex=0.1, col="blue")

plot((d.n0[,1] + 0.2) /sum(d.n0[,1] + 0.2),
    (d.n0[,2] + 0.2)/sum(d.n0[,2] + 0.2),
    log="xy", pch=19, col=rgb(0,0,0,0.1),
    xlab="P(D) 1", ylab="P(D) 2")
points(d.dir[1,], d.dir[2,], pch=19, cex=0.2, col=rgb(1,0,0,0.1))
points(d.dir[1,], d.dir[3,], pch=19, cex=0.2, col=rgb(1,0,0,0.1))
points(d.dir[1,], d.dir[4,], pch=19, cex=0.2, col=rgb(1,0,0,0.1))
points(E_dir, ml, pch=19,cex=0.1, col="blue")
```



In summary an observed count of 0 is does not mean that 0 is what we would observe upon replication. In other words, and observation of 0 does not mean that there is 0 probability of observing a 0. In fact, in this dataset a value of 0 has an Expected value of about 0.3. Thus there is a good justification to adjust 0 values with a pseudocount (from zCompositions) or a prior probability to more accurately capture the real underlying structure of the data at the low count margin.