Analysis of Single Moculecule Photobleaching

Keegan Hines

January 4, 2014

Introductory

This tutorial is a walkthrough of the analysis functions that accompany the paper *Inferring Subunit Stoichiometry from Single Molecule Photobleaching* which are found in the R package **smp**.

smp

If you haven't installed the package yet, the easiest way is install it from github using the **devtools** package.

```
library(devtools)
install_github("smp", username = "keeganhines")
```

Now when we import the **smp** package, we should have access to the important functions and an example dataset called `bleaching_data`.

```
library(smp)
bleaching_data
```

```
##   [1] 3 1 3 1 2 1 0 2 3 2 1 3 1 2 1 1 1 3 1 0 2 2 3 1 3 1 1 3 0 3 2 2 3 1 2
##  [36] 1 0 3 2 2 3 3 1 1 2 1 1 2 1 3 3 2 1 2 3 1 2 2 3 2 1 2 2 3 2 3 2 1 3 2
##  [71] 2 2 1 2 2 2 2 2 2 4 2 2 3 4 2 2 1 2 3 1 0 4 3 3 2 2 1 2 3 2
```

Analysis Functions

One of functions available to us is called `estimate.theta()`. Recall from the main text of the accompanying paper that the parameter θ is the probability of a particular binary event happening. This parameter plays an important role in a binomial distribution, Bn(n,θ), which is the probability distribution of observing any number of events, given that n are possible and each occurs with probability θ. Ultimately, we need to estimate n and θ from some observations. Let's try it out with the example dataset.

```
estimate.theta(bleaching_data)
```

```
## [1] "For 4 subunits, best estimate of theta is 0.48"
```

By defeault, this function finds the largest observed number of events, here it is 4, and fixes n to that value. Given that n is 4, the best point estimate of θ is 0.48. In the paper, it was noted that the estimate of θ depends on an assumption about n such that as n increases, the estimate of θ will decrease. The function `estimate.theta()` provides a point estimate of θ with respect to a particular n. Let's abandon the example dataset for now and use simulated data (using `rbinom()`) so that we can compare the results to the "true" values.

```
> simulated.data <- rbinom(100, 4, 0.5)  #100 observations from Bn(4,.5) model
> table(simulated.data)

## simulated.data
##  0  1  2  3  4
##  7 25 43 21  4

> estimate.theta(simulated.data)

## [1] "For 4 subunits, best estimate of theta is 0.47"
```

By default, this function finds the largest observation and assumes that to be n. However, we can specify that we wish to estimate θ with respect to any potential n.

```
> estimate.theta(simulated.data, n = 5)

## [1] "For 5 subunits, best estimate of theta is 0.38"
```

It is pointed out in the main text that estimating the true n from such observations can be problematic. As we just saw, the optimal estimate of θ will shift downward as we consider larger n. In some instances, the result will be that many potential n can fit some data identically well. Due to this, methods were developed to quantify confidence when anaylzing such observtions. The two methods of quantification are described in equations [4] and [9] in the main text and are implemented in the function `confidence()`.

This function requires only one argument which is the name of the variable containing the data. Here, we will simulate random data an to more easily examine the effects of the parameters on estimation confidence.

```
> confidence(simulated.data)

## [1] "Number of subunits is estimated to be 4 with confidence of 0.5484"

> confidence(rbinom(100, 4, 0.5))

## [1] "Number of subunits is estimated to be 4 with confidence of 0.6662"
```

For example, increasing θ results in increased confidence, as we might have expected. Additionally, larger n results in less confidence and larger N results in higher confidence.

```
> # Notice effect of increasing theta
> confidence(rbinom(100, 4, 0.6))

## [1] "Number of subunits is estimated to be 4 with confidence of 0.9474"

> confidence(rbinom(100, 4, 0.7))

## [1] "Number of subunits is estimated to be 4 with confidence of 0.9944"

> # Notice effect of increasing n
> confidence(rbinom(100, 4, 0.7))

## [1] "Number of subunits is estimated to be 4 with confidence of 0.9944"

> confidence(rbinom(100, 8, 0.7))
```

```
## [1] "Number of subunits is estimated to be 8 with confidence of 0.6999"

> # Notice effect of larger sample size
> confidence(rbinom(100, 4, 0.5))

## [1] "Number of subunits is estimated to be 4 with confidence of 0.7727"

> confidence(rbinom(500, 4, 0.5))

## [1] "Number of subunits is estimated to be 4 with confidence of 0.9738"
```

I mentioned that the function **confidence()** can implement one of two calculations. By default, this function computes the full Bayesian estimate of confidence which is described by equation [9] of the main text. In the paper, I argue that this method is the most conservative and appropriate way to estimate confidence. Nonetheless, the simpler calculation of equation [4] can also be implemented by including the argument `Bayes=FALSE`.

```
> confidence(simulated.data)

## [1] "Number of subunits is estimated to be 4 with confidence of 0.5484"

> confidence(simulated.data, Bayes = FALSE)

## [1] "Number of subunits is estimated to be 4 with confidence of 0.5464 ."
```

In general, the non-Bayes estimate will be an overestimate of confidence. However, for small sample sizes (or low θ), it is not impossible for the Bayesian estimate to be larger. The reason for this is that the Bayesian estimator takes into the account the full variance in the observations, as where the simpler method just assumes that θ is known with perfect precision and does not consider the actual data. As a result, the Bayesian estimator is more sensitive to sampling variance in small sample sizes. Additionally, as N increases, the two methods will become equivalent.

```
> confidence(rbinom(550, 4, 0.5))

## [1] "Number of subunits is estimated to be 4 with confidence of 0.9943"

> confidence(rbinom(550, 4, 0.5), Bayes = FALSE)

## [1] "Number of subunits is estimated to be 4 with confidence of 0.998 ."
```

Also, it is worth pointing out a potential bug that might skew your interpretation. Recall that the function `confidence(...,Bayes=FALSE)` computes equation [4] from the main paper. That is, it computes $\alpha = 1 - (1 - \hat{\theta}^{\hat{k}+1})^N$ for the input dataset. Note that the value of α will always be between 0 and 1 and can *never* be exactly 1. However,

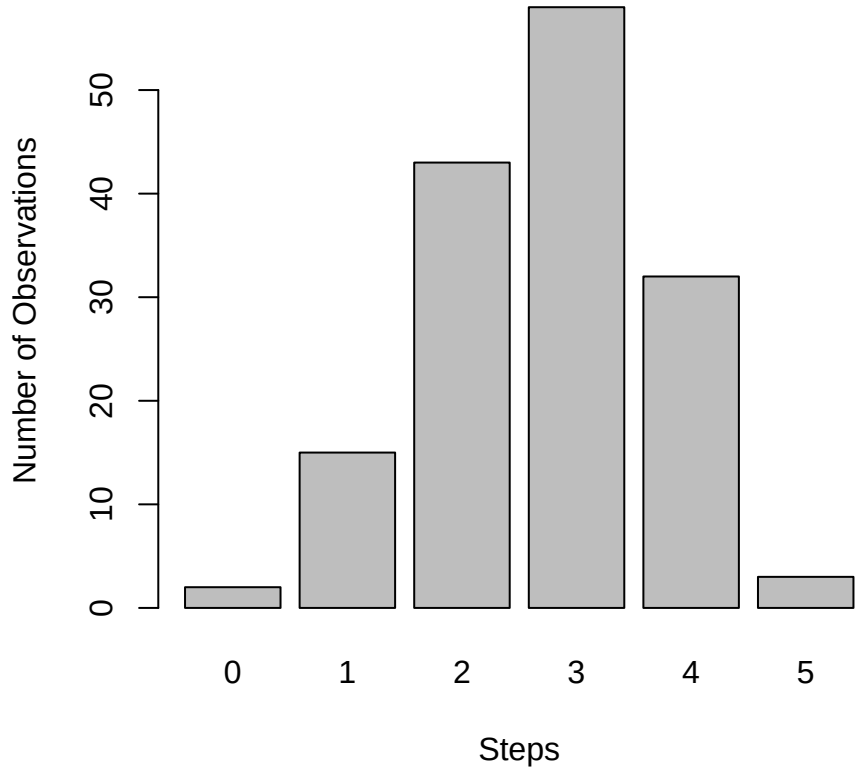
```
confidence(rbinom(500, 4, 0.7), Bayes = FALSE)

## [1] "Number of subunits is estimated to be 4 with confidence of 1 ."
```

We are told that α equals 1 because we have exceeded the machine precision that R is able to represent. That is, the value of α is so slightly below 1, that R doesn't bother representing it and instead returns the integer 1. Since we know this can never be the case, we just need to keep in mind that α is just higher than some arbitrarily high threshold, such as $\alpha > .9999$ or so.

The final function can be used to address whether certain observations might be artifacts. As described in equation [10] of the main text, the parameter γ is an estimate of whether the largest number of observed n occur with anomalously low prevalence.For example, let's suppose that the true n is four and that a data collection algorithm resulted in a small number of artifactual observations larger than four. We can visually inspect the data and notice that the distribution looks inconsistent. The function `gamma()` provides an estimate of observing a similar distribution under the null hypothesis that n is equal the largest number of observations. Similar to a p-value, a low value for γ is evidence against the null hypothesis and thus we might exclude all observations larger than four as artifacts.

```
> messy.data <- c(rbinom(150, 4, 0.7), 5, 5, 5)  # We have 150 observations from a
> # binomial distribution with n=4, and also several anomalous observations
> # of size 5.
> barplot(table(messy.data), xlab = c("Steps"), ylab = c("Number of Observations"))
```




```
> # Data looks strange. It seems we should conclude n=5, but maybe the
> # events of size 5 are artifacts.
> gamma(messy.data, Bayes = FALSE)

## [1] 0.0559
```

In this instance, γ is very small and therefore we might reject the null hypothesis, depending on our threshold. If we remove all observations of size 5 from the data set, we can ask again whether the resulting data are anomalous with respect to how many events of size n are observed.

```
> gamma(messy.data[messy.data < 5], Bayes = FALSE)

## [1] 0.6536

> confidence(messy.data[messy.data < 5], Bayes = FALSE)

## [1] "Number of subunits is estimated to be 4 with confidence of 0.999 ."
```

This estimate of γ is large, which indicates that the distribution is not atypical. Therefore, we might conclude that n=4 and it turns out that we can make that claim with high confidence.