Binomial test of random placement of gene calls

Dave Schneider

March 13, 2024

Let q be the length fraction of the genome assigned the low GC content class and N be the total number of gene calls. If the gene calls are "random distributed" between regions of normal and low GC content then one would expect that the proportions of gene calls in the two classes would more or less follow the proportions of total lengths of the two classes. Let k be the number of gene calls in regions of low GC content.

Consider a single gene call (i.e., $N = 1$). If it is randomly placed on the genome then it will be assigned to a low GC content region with probability q and, correspondingly, probability $1 - q$ for occurrence in a region of normal GC content. In other words

$$\mathbb{P}\left[k = 0 \mid N = 1\right] = 1 - q \tag{1}$$

$$\mathbb{P}\left[k = 1 \mid N = 1\right] = 1 \tag{2}$$

Now consider the outcomes for $N = 2$ *independent* gene calls sampled from the same distribution. There is one way to get either zero or two "hits"

$$\mathbb{P}\left[k = 0 \mid N = 2\right] = \left(1 - q\right)^2 \tag{3}$$

$$\mathbb{P}\left[k = 2 \mid N = 2\right] = q^2 \tag{4}$$

but two ways to get exactly one hit

$$\mathbb{P}\left[k = 1 \mid N = 2\right] = 2\left(1 - q\right) q \tag{5}$$

the first hit could be to a low GC content region and the second to a normal region, or *vice versa*.

In general, the number of ways that the outcomes can be partitioned into two classes is given by the well known binomial coefficients. So, the general formula is for finding $k = n$ out of N hits in regions of low GC content is

$$\mathbb{P}\left[k = n \mid N\right] = \binom{n}{N} q^{n} \left(1 - q\right)^{N-n} \tag{6}$$

The binomial test (e.g., `scipy.stats.binomtest`) can be used to determine whether the observation of $k = n$ out of N hits occurring regions of low GC content is consistent with the null hypothesis that the the calls are random assigned to regions of low GC content with probability q. In other words, the null hypothesis implies that true probability mass function for k is given by Equation 6). This function is peaked in the vicinity of $n = Nq$. Note, you may want to use the `scipy.stats.binom` package to graph this function. The peaked nature of this function suggests that if k is very different than Nq then it is unlikely that the true probability mass function is given by Equation 6.

The question is how close do k and Nq have to be in order for Equation 6 to provide a plausible explanation of the observed data. The call

```
result = scipy.stats.binomtest(k, N, q, alternative='two-sided')
```

provides the answer. The `alternative=two-sided` argument implies that the test should consider the possibility that the true proportion is either greater or less than `q`. You can retrieve the so-called p-value

```
print('p-value: {}'.format(result.pvalue)
```

as well as the confidence interval (see documentation for details).