Poisson PCA

Tianshu Huang and Toby Kenney

April 30, 2019

`Poisson_Corrected_PCA(X,k=dim(X)[2],transformation=NA,seqdepth=FALSE)`

Performs PCA estimation with Poisson noise correction. X is the data matrix of Poisson random variables with means Lambda given by some underlying multivariate distribution. We want to estimate the principal components of the distribution of f(Lambda) where f is some transformation. We may also correct for sequencing depth — that is, we assume that the Poisson means in each sample are subject to large multiplicative noise associated more with the experimental procedure than with the important signal. This assumption is common in microbiome analysis.

Example:

```
> library(PoissonPCA)
> X <- rbind(c(0,1,3,5,2),c(2,1,4,4,0),c(3,2,2,0,5))
> Poisson_Corrected_PCA(X)

Call:
Poisson_Corrected_PCA(X = X)

Standard deviations:
[1] 3.4537660 0.6257951       NaN       NaN       NaN

 3  variables and  3 observations.
```

`LinearCorrectedVariance(X)`

`LinearCorrectedVarianceSeqDepth(X)`

These functions estimate the latent variance of the Poisson means. The first function assumes that total count is a potentially important variable. The second assumes it has high noise, and treats the latent Poisson means as compositional.

`TransformedVariance(X,g,CVar)`

`TransformedVarianceECV(X,g,ECVar)`

These functions estimate the variance of some function of the Poisson means. g is an estimator of this function of the Poisson means. CVar and ECVar are estimators of the conditional variance of g(X) for fixed latent Lambda. CVar is a single estimator for a given X, while ECVar estimates the expected value of g(X) over a sample of values with different values of Lambda.

`make_compositional_variance(Sigma)`
`make_compositional_min_var(Sigma)`

These functions take a variance estimate Sigma and assume that each entry has had a constant added to it. The first function estimates the compositional variance matrix that best approximates Sigma, while the second function subtracts the largest possible constant from all entries of Sigma, so that the resulting matrix remains non-negative definite.

`get_scores(X,V,d,k,transformation,mu)`
`get_scores_log(X,V,d,k,mu)`

Given the principal components of the transformation of Lambda, these functions estimate the scores of each observation X. That is, the projection of the corresponding latent Lambda onto the principal component space. This essentially gives the projection of the data onto the latent principal component space. The first function applies to a general transformation of Lambda, while the second function works specifically for the logarithm transformation.

`polynomial_transformation(coeffs)`
`makelogtransformation(a,N)`

For many of the methods in this package, a transformation consists of a list of 4 functions:

- A function `f` which evaluates the transformation on a particular value.
- A function `g` which estimates `f(Lambda)` from an observation `X`.
- A function `solve` which computes `Lambda` from `f(Lambda)`.
- A function `CVar` which estimates the conditional variance of `g(X)` for the latent value of `Lambda` corresponding to an observed value of `X`. For some transformations, it is more convenient to have a function `ECVar` which computes the average of this conditional expectation over a vector of observed values of `X`.

Example:

```
> library(PoissonPCA)
> X <- rbind(c(0,1,3,5,2),c(2,1,4,4,0),c(3,2,2,0,5))
> Poisson_Corrected_PCA(X)

Call:
Poisson_Corrected_PCA(X = X)

Standard deviations:
[1] 3.4537660 0.6257951       NaN       NaN       NaN

 3  variables and  3 observations.
```