- n – number of cells
- J – number of genes

Current model:

$$\log(M) = X_M * \alpha_M + U * V$$
$$logit(\Pi) = X_\Pi * \alpha_\Pi + U * W$$

- M is $n \times J$ matrix. M_{ij} is the mean parameter for the NB distribution describing the expression of gene j in cell i
- Π is $n \times J$ matrix of dropout probabilities
- X_M is the known $n \times kx_M$ design matrix for the negative binomial part regression.
- X_Π is the known $n \times kx_\Pi$ design matrix for the logistic regression.
- U is the unknown $n \times p$ matrix of latent factors affecting both M and Π but with different coefficients (resp. V and W)

Parameters to estimate are α_M, α_Π, U, W and θ_j (gene-specific (at least for the moment) dispersion parameters for $j = 1, \ldots, J$).

The supposed method to estimate parameters:

1. Initialize all unknown parameters (with PCA or RUV?)
2. Alternate between two steps:
    - All left hand sides are fixed, estimate right hand sides by maximum likelihood
    - All right hand sides are fixed, estimate U by maximum likelihood

Where we are:

There are codes for each of the steps separately and the one which puts two steps together.

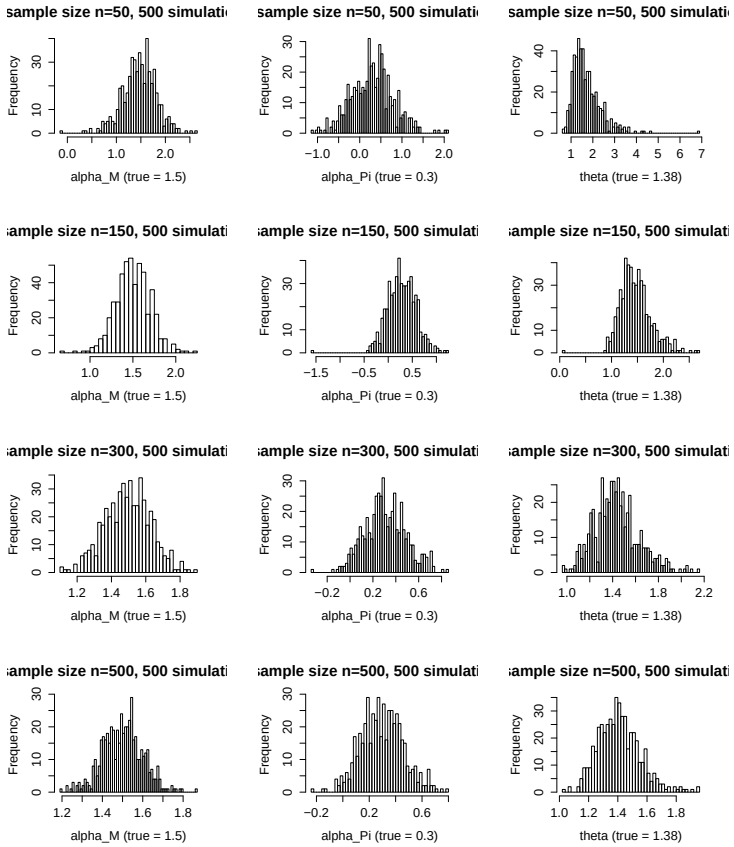
When tested, two steps together did not work

Code was put in a form of R package (by JP) where I cleaned up the file functions_svetlana.R which currently contains likelihood functions and gradient functions for each of the two steps. I also added the descriptions of parameters.

To debug code, I compared the output of my code for the first step (optimization wrt "right parts") with the output of pscl, U being fixed equal to its true value which was used to simulate data.

The outputs are the same (as expected because the first step with known U is basically the same thing that the pscl implementation)

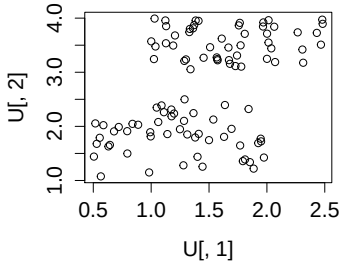
I did some numerical experiences with this first step optimization in order to see how stable is it and how it depends on the sample size n (optimization is done gene by gene, so the sample size is the number of cells n).



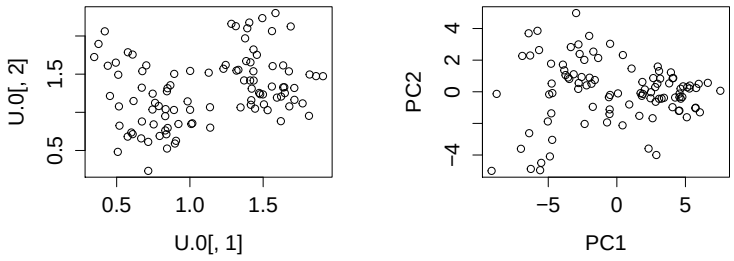
Some points to discuss:

- I tested one alternation on a real data set with 96 cells and 8000 genes. Time was around 20 minutes (for one round of optimization). We should probably find a way to accelerate.
- Question of normalization: if we want to make an advantage of using NB applied to counts instead of normalized data (which is not integer any more), we should fix the problem of size factors.
- PCA versus our method: some first comparison.

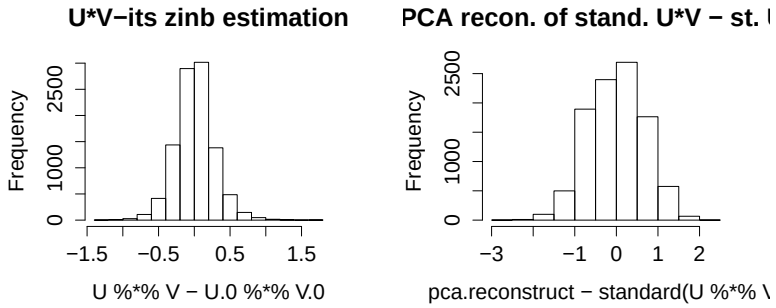
  Data generated with two groups, matrix U has two columns, 37% of zero values in count matrix.



  Representation of cells with their estimated values of U and in PC1-PC2 projection.



  Histograms of errors of matrix reconstruction:



Likelihoods at 5 successful iterations :

$$-49742.67 \; -44183.39 \; -44591.79 \; -44606.31 \; -44554.55$$

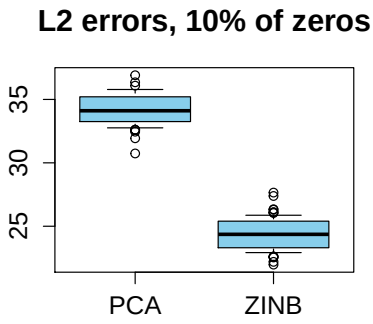
March 1, 2016:

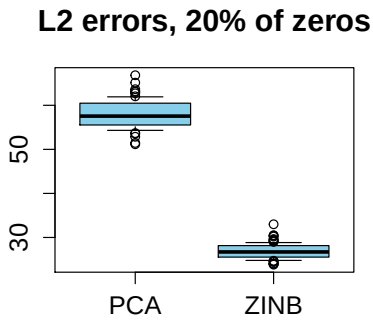
- Genes without zeros: initially were estimated by ordinary negative binomial regression (glm.nb) which has a problem with stability of optimization (errors). At present all gene parameters (V and W matrices) are estimated via zero inflated likelihood maximization with penalization taking care of genes without zeros.
- Initialization is at present done by PCA
- Added a stop criterion of optimization: when the change in likelihood function is less than 0.5%, optimization stops. In practice, when 2 factors are to be estimated, about 4 iterations were required on simulated data.

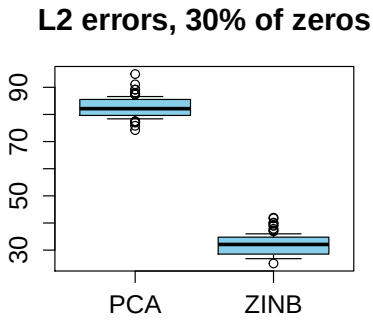
Comparison with PCA:

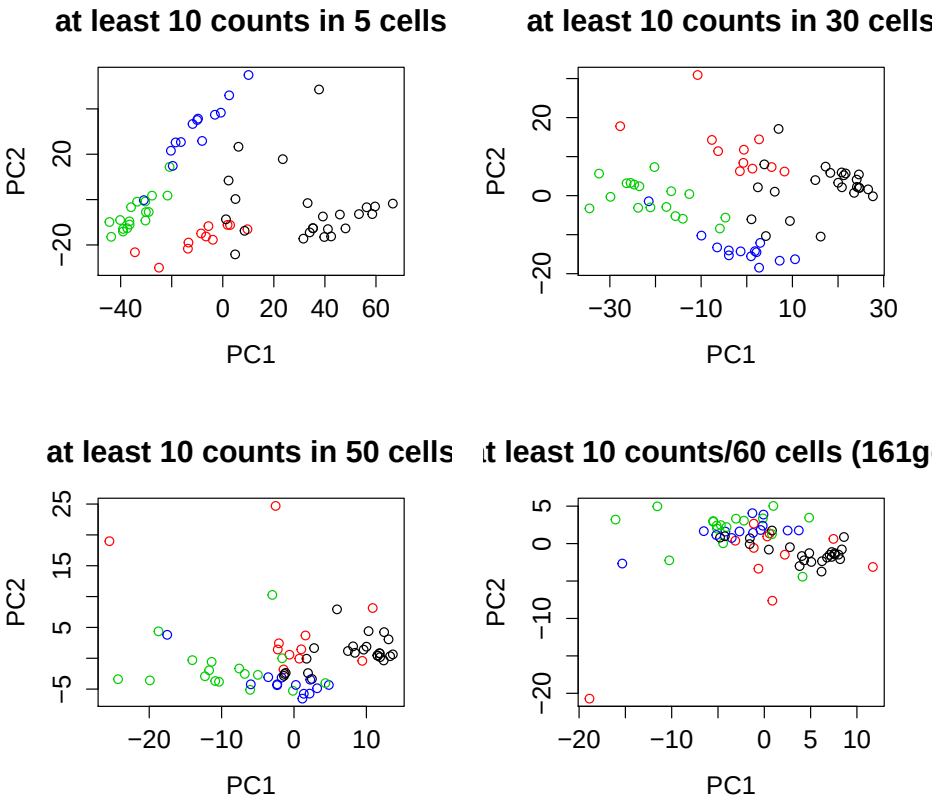
- Three values of zero proportions are taken: 10%, 20%, 30%
- For each fraction of zeros, 50 data sets with "two groups" of cells were simulated.
- PCA and ZINB were performed on each data set
- ZINB reconstructs directly the matrix of log expressions ($\log M = UV$). Its error is quantified as L_2 distance between the ZINB reconstructed matrix and the "true" underlying matrix.

- PCA is done on logs of counts plus 1 with centering and scaling. I take the reconstruction based on first two principal components, then I multiply by sd and add mean to put both reconstructions on the same scale and I compute again the L_2 distance between PCA reconstructed matrix and the true matrix.
- As expected, ZINB is smarter than PCA even at 10% of zeros and the ratio of the PCA error over the ZINB error increase with the fraction of zeros.








at least 10 counts in 5 cells
at least 10 counts in 30 cells
at least 10 counts in 50 cells
t least 10 counts/60 cells (161g
PC1
PC2