- 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.

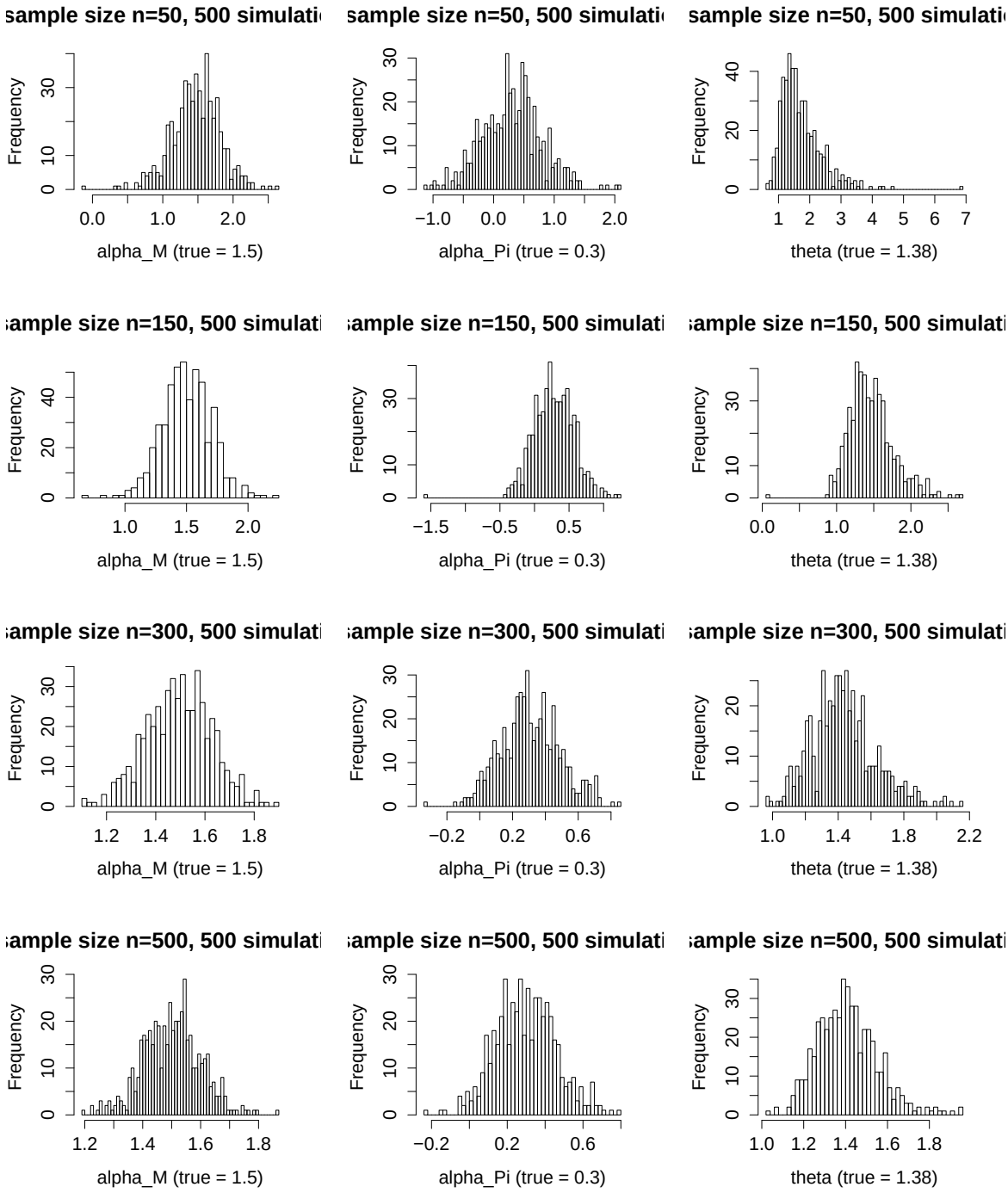
When tested, two steps together did not work

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).


sample size n=50, 500 simulati
sample size n=50, 500 simulati
sample size n=50, 500 simulati
Frequency
alpha_M (true = 1.5)
alpha_Pi (true = 0.3)
theta (true = 1.38)
sample size n=150, 500 simulati
sample size n=150, 500 simulati
sample size n=150, 500 simulati
Frequency
alpha_M (true = 1.5)
alpha_Pi (true = 0.3)
theta (true = 1.38)
sample size n=300, 500 simulati
sample size n=300, 500 simulati
sample size n=300, 500 simulati
Frequency
alpha_M (true = 1.5)
alpha_Pi (true = 0.3)
theta (true = 1.38)
sample size n=500, 500 simulati
sample size n=500, 500 simulati
sample size n=500, 500 simulati
Frequency
alpha_M (true = 1.5)
alpha_Pi (true = 0.3)
theta (true = 1.38)