SKETCH OF THE "ALTERNATING SQP" METHOD FOR FITTING POISSON TOPIC MODELS

PETER CARBONETTO*

1. Some derivations. Given an $n \times p$ matrix of counts X with entries $x_{ij} \geq 0$, the aim is to fit a Poisson model of the counts,

$$\begin{aligned} p(x) &= \prod_{i=1}^{n} \prod_{j=1}^{p} p(x_{ij}) \\ &= \prod_{i=1}^{n} \prod_{j=1}^{p} \mathrm{Poisson}(x_{ij}; \lambda_{ij}), \end{aligned} \tag{1.1}$$

in which the Poisson rates are given by $\lambda_{ij} = \sum_{k=1}^{K} l_{ik} f_{jk}$. The model is determined by a $p \times K$ matrix F with entries $f_{ik} \geq 0$ (the "factors") and an $n \times K$ matrix L with entries $l_{ik} \geq 0$ (the "loadings"). Fitting F and L is equivalent to non-negative matrix factorization with the "beta-divergence" cost function [3]. It can also be used to recover a maximum-likelihood estimate for the latent Dirichlet allocation (LDA) model [1]. So fitting this model is useful for a wide range of applications.

The log-likelihood for the Poisson model is

$$\log p(x \mid F, L) \propto \sum_{i=1}^{n} \sum_{j=1}^{p} x_{ij} \log(\textstyle\sum_{k=1}^{K} l_{ik} f_{jk}) - \displaystyle\sum_{i=1}^{n} \sum_{j=1}^{p} \sum_{k=1}^{K} l_{ik} f_{jk}, \tag{1.2}$$

where the constant of proportionality is obtained from factorial terms in the Poisson densities. Our specific aim is to find a F and L that maximizes the log-likelihood (1.2); that is, we would like to solve

$$\begin{array}{ll} \text{minimize} & -\log p(x \mid F, L) \\ \text{subject to} & F \geq 0, L \geq 0. \end{array} \tag{1.3}$$

In the remainder, we derive an efficient approach to doing this.

Our strategy for solving (1.3) is to alternate between solving for F with L fixed, and solving for L with F fixed. When solving for F with L fixed (and vice versa), the problem naturally decomposes into a collection of much smaller subproblems that are much more tractable to solve. All the subproblems are of the following form:

$$\begin{array}{ll} \text{minimize} & \phi(y; B, w) \\ \text{subject to} & y_k \geq 0 \text{ for all } k = 1, \ldots, K, \end{array} \tag{1.4}$$

in which the objective function is defined as

$$\phi(y; B, w) = -\sum_{i=1}^{n} w_i \log\left(\textstyle\sum_{k=1}^{K} b_{ik} y_k\right) + \displaystyle\sum_{i=1}^{n} \sum_{k=1}^{K} b_{ik} y_k. \tag{1.5}$$

*Dept. of Human Genetics and the Research Computing Center, University of Chicago, Chicago, IL

To see the connection between subproblem (1.4) and the original optimization problem (1.3), observe that the negative log-likelihood can be recovered as

$$-\log p(x \mid F, L) = \sum_{i=1}^{n} \phi(l_i; F, x_i), \tag{1.6}$$

where x_i is the ith row of X and l_i is the ith row of L. Alternatively, it can be recovered as

$$-\log p(x \mid F, L) = \sum_{j=1}^{p} \phi(f_j; L, x_i), \tag{1.7}$$

in which x_j is the jth column of X, and f_j is the jth row of F. Therefore, when F is fixed, each row of L can be separately optimized by solving a problem of the form (1.4), and when L is fixed, each row of F can be separately optimized by solving a problem of the form (1.4).

Initially this may seem like a sensible strategy, but directly optimizing 1.4 turns out to be difficult to do for numerical reasons: in practice, the entries of B can be very large or very small, resulting in solutions y in which all the entries are either very large or very small. This makes it difficult to devise an algorithm that will work well for all possible input matrices B.

I propose to solve for y indirectly by instead solving

$$\begin{array}{ll} \text{minimize} & f(t; P, u) \\ \text{subject to} & t_k \geq 0 \text{ for all } k = 1, \ldots, K, \end{array} \tag{1.8}$$

in which the new objective function is

$$f(t; P, u) = -\sum_{i=1}^{n} u_i \log\left(\textstyle\sum_{k=1}^{K} p_{ik} t_k\right) + \sum_{k=1}^{K} t_k, \tag{1.9}$$

where I've defined

$$\begin{aligned} u_i &= \frac{w_i}{\sum_{i'=1}^{n} w_{i'}} \\ p_{ik} &= b_{ik} \times \frac{\sum_{i'=1}^{n} w_{i'}}{\sum_{i'=1}^{n} b_{i'k}}. \end{aligned}$$

After finding the solution $t^\star$ to (1.8), the solution $y^\star$ to (1.4) is recovered as

$$y_k^\star = t_k^\star \times \frac{\sum_{i=1}^{n} w_i}{\sum_{i=1}^{n} b_{ik}}. \tag{1.10}$$

The main advantage of solving (1.8) is that the solution is numerically well behaved; in particular, the entries of the solution $t^\star$ sum to 1 [2].

REFERENCES

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan, *Latent Dirichlet allocation*, Journal of Machine Learning Research, 3 (2003), pp. 993–1022.

[2] Y. Kim, P. Carbonetto, M. Stephens, and M. Anitescu, *A fast algorithm for maximum likelihood estimation of mixture proportions using sequential quadratic programming*, arXiv, 1806.01412 (2019), https://arxiv.org/abs/1806.01412.

[3] D. D. Lee and H. S. Seung, *Algorithms for non-negative matrix factorization*, in Advances in Neural Information Processing Systems 13, 2001, pp. 556–562.