Multivariate Poisson Lognormal model optimisation, inference and application in high dimensions

Master Thesis

supervised by

Julien Chiquet, Joon Kwon, Laure Sansonnet

Master 2 Mathematics, Vision, Learning

September 2021

Bastien Batardière

Contents

Acknowledgements

First and foremost I want to thank my advisors Julien, Joon and Laure. I learned a lot from them. The meetings were very interesting and enriching. I want to thank Armand for all the snack breaks, the good atmosphere in the office, his advices and his meticulous re-reading. I also want to thank Pierre, Stéphane and Julien for the discussions on the PLN model, especially for the importance sampling part.
I trully enjoyed this internship at INRAE, the atmosphere and the people. It made me want to continue, so that I will pursue a PhD under Julien's supervison. I am looking forward to engaging in this experience.

Integrity statement on plagiarism

To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made.

Bastien Batardiere

1 Introduction

quantity

1.1 Notations

We introduce here some notations. Let p and $\tilde{p}$ be two probability distributions on $\mathbb{R}^q$, $q \in \mathbb{N}^\star$, let

$$A = (a_{ij})_{1 \leqslant i,j \leqslant m}$$
$$B = (b_{ij})_{1 \leqslant i \leqslant m, 1 \leqslant j \leqslant k}$$
$$C = (c_{ij})_{1 \leqslant i \leqslant m, 1 \leqslant j \leqslant k}$$

- $\mathrm{tr}(A) = \sum_{i=1}^{m} a_{ii}$.
- $\|B\| = \left(\sum_{i=1}^{m} \sum_{j=1}^{k} b_{ij}^2 \right)^{\frac{1}{2}}$
- $\mathcal{S}_q^{++} = \{M \in \mathcal{M}_{q,q} \mid M = M^\top, (\forall x \in \mathbb{R}^q, Mx = \lambda x) \implies \lambda > 0\}$
- $KL(p||\tilde{p}) = \mathbb{E}_p \left[\log \left(\frac{p(X)}{\tilde{p}(X)} \right) \right]$
- $B \odot C = (d_{ij})_{1 \leqslant i \leqslant m, 1 \leqslant j \leqslant k}, \ d_{ij} = b_{ij} \times c_{ij}$

1.2 Problem

In ecology or genomics, it is usual to deal with non Gaussian observations (number of species, number of genes, etc.). A typical example is the raw single-cell transcriptomic count distribution, aiming to understand the co-variations between the activity of each gene (number of times the gene is expressed) in the cell. The Gaussian setting provides a canonical way to model such dependencies, but cannot be applied to count data. The Poisson lognormal model allows us to do so. We focus on two different inference approaches: a variational algorithm that infer an approximate maximum likelihood in Section 2 and a Monte-Carlo approach that estimates the gradients of the log likelihood in Section 3. The variational inference has already been studied in Chiquet et al. (2018), we study it again with the aim of finding the better maximum likelihood approximation in the shortest possible time.

1.3 Previous work

The following two paragraphs can be found in Chiquet et al. (2018).

The Gaussian setting is obviously convenient as the dependency structure is entirely encoded in the covariance matrix. In many applications (Royle and Wikle (2005), Srivastava and Chen (2010)) Gaussian models need to be adapted to handle specific measurement types, such as binary or count data. For count data, the multivariate Poisson distribution seems a natural counterpart of the multivariate normal. However, no canonical form exist for this distribution (N. L. Johnson, 1997), and several alternatives have been proposed in the literature including Gamma-Poisson (Nelson, 1985) and lognormal Poisson (Aitchison and Ho, 1989). The latter takes advantage of the properties of the Gaussian distribution to display a larger panel of dependency structure than the former, but maximum likelihood-based inference raises some issues

as the MLE of the covariance matrix is not always positive definite. One main issue of non-Gaussian setting arises from the fact that their conditional distribution is often intractable which hampers the use of an Expectation-Minimization (Dempster et al., 1977) (EM) strategy.

Variational approximations (Jordan et al. (1999) Wainwright and Jordan (2008)) have become a standard tool to approximate such conditional distributions. Karlis (2005) uses such an approximation for the inference of the one-dimensional Poisson-lognormal model and derives a variational EM (VEM) algorithm. Hall et al. (2011) provide a theoretical analysis of this approximation for the same model and prove the consistency of the estimators. Indeed, even the conditional distribution of one single hidden coordinate (given allothers) is unknown, which makes regular Gibbs sampling inaccessible. As a consequence, Lee and Seung (2001) use moment estimates, whereas, in a Bayesian context, Li and Tao (2010) resort to a variational approximation of the conditional distribution.

Looking for the Maximum likelihood Estimator (MLE) is an alternative to variational approximation. It is challenging since the denominator of the conditional distribution (distribution of the latent variables given the observations) is unknown. Numerical integration is possible for small dimensions (Williams and Ebel (2012), Izsák (2008)) but unreachable for dimensions bigger than 4. Aitchison and Ho (1989) used Newton-Raphson algorithm to obtain the MLE in dimension 4. Silva et al. (2019) used the PLN model for clustering Transcriptome Sequencing Data using MCMC methods in dimension 18 but the running time has not been made public. In a bayesian setting, Wang et al. (2020) used it for crash prediction, applying the Integrated Nested Laplace Approximation (INLA) approach proposed by Rue et al. (2009) to carry out the Bayesian inference.

1.4 The Poisson lognormal (PLN) model

We introduce here the Poisson lognormal (PLN) model (Aitchison and Ho, 1989). Let $n, p, d, q \in \mathbb{N}_\star^4$. We consider:

- n cells $(i = 1, \dots, n)$
- p genes $(j = 1, \dots, p)$
- n measures $X_i = (x_{ih})_{1 \leqslant h \leqslant d}$: X_{ih} = given descriptor (covariate) for cell i.
- n measures $Y_i = (Y_{ij})_{1 \leqslant j \leqslant p}$: Y_{ij} corresponds to the number of times the gene j is expressed in cell i.

We assume that for all $1 \leqslant i \leqslant n$, the observed abundances $(Y_{ij})_{1 \leqslant j \leqslant p}$ are independent conditionally on a latent variable $Z_i \in \mathbb{R}^p$ such that :

$$\begin{aligned} W_i &\sim \mathcal{N}(0, I_q) \\ Z_i &= \beta^\top X_i + C W_i \\ (Y_{ij} \mid Z_{ij}) &\sim \mathcal{P}(\exp(o_{ij} + Z_{ij})) \end{aligned} \tag{1}$$

where $O = (o_{ij})_{1 \leqslant i \leqslant n, 1 \leqslant j \leqslant p}$ are known offsets, $\beta = (\beta_{kj})_{1 \leqslant k \leqslant d, 1 \leqslant j \leqslant p}$ is an unknown regression parameter and $C \in \mathbb{R}^{p \times q}$ (unknown) sends the latent variable W_i from a space of dimension q to a space of dimension p. For $i_0 \neq i_1$, we assume $W_{i_0} \perp\!\!\!\perp W_{i_1}$ so that $Y_{i_0} \perp\!\!\!\perp Y_{i_1}$. We denote $Y \in \mathbb{R}^{n \times p}$ (resp. $X \in \mathbb{R}^{n \times d}, Z \in \mathbb{R}^{n \times p}$) the matrix obtained by stacking the Y_i's (resp. X_i, Z_i) in line.

Note that mulitplying C by an orthogonal matrix does not modify the model, so that C is not identifiable. The unknown (and identifiable) parameter is $\theta = (\Sigma, \beta)$, where $\Sigma = CC^\top$ is the covariance matrix of each Z_i. The dimension $q \leqslant p$ is a hyperparameter that also needs to be tuned. We will consider two very different cases :

- p=q
- q<p

When $p < q$, the model is called PLN-PCA (PCA for Principal-Composent-Analysis) since Σ is a rank q matrix in a space of dimension p ($\Sigma = CC^\top$ has rank at most q since $C \in \mathbb{R}^{p \times q}$). We can either see the latent variables as Z_i or W_i, both being equivalent since the variables Z_i are (unseen and) directly derived from W_i. An equivalent model that would see Z as latent variables would be the following:

$$\begin{aligned} Z_i &\sim \mathcal{N}(\beta^\top X_i, \Sigma) \\ Y_{ij} \mid Z_{ij} &\sim \mathcal{P}\left(\exp\left(o_{ij} + Z_{ij}\right)\right) \end{aligned} \tag{2}$$

When $p = q$, we will prefer to see Z_i as latent variables, since Z_i will be Gaussian with a positive definite covariance matrix, whereas when $p < q$, we will prefer to see W as latent variables.

1.5 Inference approaches

Our goal is to infer the model parameter by maximizing the log likelihood. We want to compute

$$\hat{\theta} = \text{argmax}_\theta \log p_\theta(Y) = \text{argmax}_\theta \log \int_{\mathbb{R}^q} p_\theta(Y, W) \mathrm{d}W \tag{3}$$

where $p_\theta(Y) = p_\theta(Y_1, Y_2, \ldots, Y_n) = \prod_{i=1}^n p_\theta(Y_i)$. Since the model depends on unobserved latent variables, we want to apply the EM algorithm. However, the integral in (3) being intractable, we can neither rely on the EM algorithm nor simple gradient ascent methods. We consider two different approaches to address the issue:

- Variational EM (Section 2)
- Gradient ascent methods based on Monte-Carlo (MC) Estimation (Section 3)

2 Variational Expectation Maximisation

A variational inference for the PLN model has already been studied in Chiquet et al. (2018). We study it once again aiming at finding a good approximation for large n and large p, such as $n = 10000$, $p = 2000$ in the shortest possible time.

For this section, we will suppose $p = q$.

2.1 Expectation-Maximisation

Given an initalisation θ^0, the Expectation-Maximisation (Dempster et al., 1977) (EM) algorithm alternates between two steps : the E step and the M step. The E step aims at computing some moments of the complete log likelihood given the data:

$$\mathbb{E}_{\theta^t}\left[\log p_\theta(Y, Z) \mid Y\right]$$

The M step consists in maximizing this quantity with respect to θ :

$$\theta^{t+1} = \arg\max_\theta \mathbb{E}_{\theta^t}\left[\log p_\theta(Y, Z) \mid Y\right]$$

The main property of this algorithm is the following :

$$\log p_{\theta^{t+1}}(Y) \geqslant \log p_{\theta^t}(Y)$$

EM requires to compute the following :

$$\mathbb{E}_{\theta^t}\left[\log p_\theta(Y, Z) \mid Y\right] = \mathbb{E}_{\theta^t}\left[\log p_\theta(Y \mid Z) \mid Y\right] + \mathbb{E}_{\theta^t}\left[\log p_\theta(Z) \mid Y\right]$$

The resulting integral is intractable because of the unknown density $p_\theta(Z_i \mid Y_i)$. Karlis (2005) suggests numerical integration but this approach is computationally too demanding when dealing with even a moderate number of variables. Monte-Carlo integration is possible but we don't deal with it in this thesis. We thus choose a variational strategy to compute it.

2.2 Variational EM

Variationnal inference (Jordan et al., 1999) aims at approximating $p_\theta(Z \mid Y)$ with some law $q^\star(Z) \in \mathcal{Q}$ from which we can compute the expectation, where $\mathcal{Q}$ is a set of probability distributions we define. Variational inference makes it possible to do inference, but we are giving up some accuracy substituting $p_\theta(Z \mid Y)$ with $q^\star(Z)$.

We find $q^\star$ by maximizing the Evidence Lower BOund (ELBO), that is :

$$\hat{q} = \operatorname*{argmax}_{q \in \mathcal{Q}} J_Y(\theta, q)$$
$$J_Y(\theta, q) := \log p_\theta(Y) - KL\left[q(Z) \| p_\theta(Z \mid Y)\right]$$

where KL is the Kullback-Leibler divergence between two probabilities. We can rewrite the the ELBO as :

$$\begin{aligned}
J_Y(\theta, q) &= \log p_\theta(Y) - KL\left[q(Z) \| p_\theta(Z \mid Y)\right] \\
&= \log p_\theta(Y) - \mathbb{E}_q\left[\log\left[q(Z)/p_\theta(Z \mid Y)\right]\right] \\
&= \log p_\theta(Y) - \mathbb{E}_q\left[\log \frac{q(Z)p_\theta(Y)}{p_\theta(Y, Z)}\right] \\
&= \log p_\theta(Y) - \mathbb{E}_q[\log q(Z)] - \mathbb{E}_q\left[\log p_\theta(Y)\right] + \mathbb{E}_q\left[\log p_\theta(Y, Z)\right] \\
&= \mathbb{E}_q\left[\log p_\theta(Y, Z)\right] \underbrace{-\mathbb{E}_q[\log q(Z)]}_{\text{entropy } \mathcal{H}(q)}
\end{aligned}$$

The Variational EM (VEM) consists in alternating between two steps :

- VE step: update q

$$q^{(h+1)} = \arg\max_{q \in \mathcal{Q}} J_Y(\theta^{(h)}, q) = \arg\min_{q \in \mathcal{Q}} KL\left[q(Z) \| p_{\theta^{(h)}}(Z \mid Y)\right]$$

- M step: update θ

$$\theta^{(h+1)} = \arg\max_{\theta} J_Y(\theta, q^{(h+1)}) = \arg\max_{\theta} \mathbb{E}_{q^{(h+1)}}\left[\log p_\theta(Y, Z)\right]$$

2.3 Variational EM for PLN

2.3.1 Choice of the class distribution

To apply Variational EM for the PLN model, we only need to define the set of distributions $\mathcal{Q}$ and derive the corresponding VE and M steps. Intuitively, should be complicated enough to approximate the data, but easy enough to have a computable ELBO. The Z_i are gaussians in the latent space, so it stands to reason that $Z_i|Y_i$ would be well Gaussian-approximated. We thus consider

$$\mathcal{Q}_{\text{Gauss}} = \{q = (q_1, \dots q_n), q_i \sim \mathcal{N}(M_i, \text{diag}(S_i \odot S_i), M_i \in \mathbb{R}^p, S_i \in \mathbb{R}^p)\} \tag{4}$$

where $\text{diag}(S_i \odot S_i)$ denotes a diagonal matrix where the diagonal is $S_i \odot S_i$. Choosing a diagonal covariance in (4) implies that the $(Z_{i,j})_{1 \leqslant j \leqslant p}$ are independant under q_i. It is natural to have this property since it is already the case under $p_\theta(Z_i)$. However, the dependency inherent to Y_i may incur dependency in $Z_i|Y_i$, but adding dependencies inside the variational parameters would be considerably more costly and wouldn't improve much the accuracy of the approximation.

Fig 1 displays the density of the posterior for some random parameter C and β, to show that a Gaussian approximation would fit. We can do so since we can derive the density distribution, ignoring a multiplication factor. Let $1 \leqslant i \leqslant n$. Thanks to the Bayes rule :

$$\begin{aligned} p_\theta(Z_i \mid Y_i) &= \frac{p_\theta(Y_i \mid Z_i)\, p(Z_i)}{p_\theta(Y_i)} \\ &\propto p_\theta(Y_i \mid Z_i)\, p(Z_i) \\ &\propto \exp\left(\sum_{j=1}^{p} -\exp(O_{ij} + Z_{ij}) + Y_{ij}(O_{ij} + Z_{ij}) - \frac{1}{2}(Z_i - \beta^\top X_i)^\top \Sigma^{-1}(Z_i - \beta^\top X_i)\right) \end{aligned}$$

2.3.2 Explicit expressions

2.3.2.1 ELBO Once we have chosen the class $\mathcal{Q}$, we only need to derive the M et VE step. To derive the M (resp. VE) step, we will compute the gradients of the ELBO and look for the maximum when q (resp. θ) is fixed. Let $\theta = (\Sigma, \beta), \Sigma \in \mathbb{S}_p, \beta \in \mathbb{R}^{d \times p}$. Let's compute the ELBO $J_{\theta,q}(Y)$. We split the ELBO in three terms :

$$J_{\theta,q}(Y) = \underbrace{\mathbb{E}_q\left[\log p_\theta(Y \mid Z)\right]}_{J_1} + \underbrace{\mathbb{E}_q\left[\log p_\theta(Z)\right]}_{J_2} + \underbrace{H(q)}_{J_3}$$

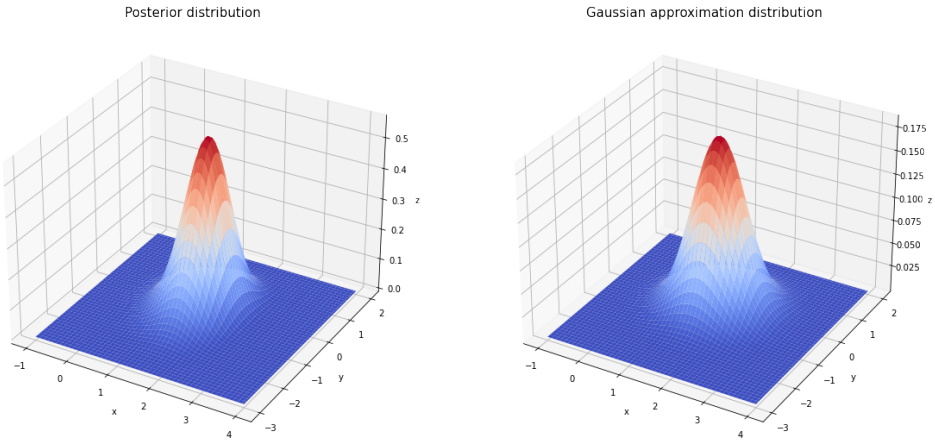

Figure 1: Comparison of the posterior distribution and Gaussian approximation in two dimensions. On the left the posterior distribution, on the right, the Gaussian approximation. We took $p = q = 2$.

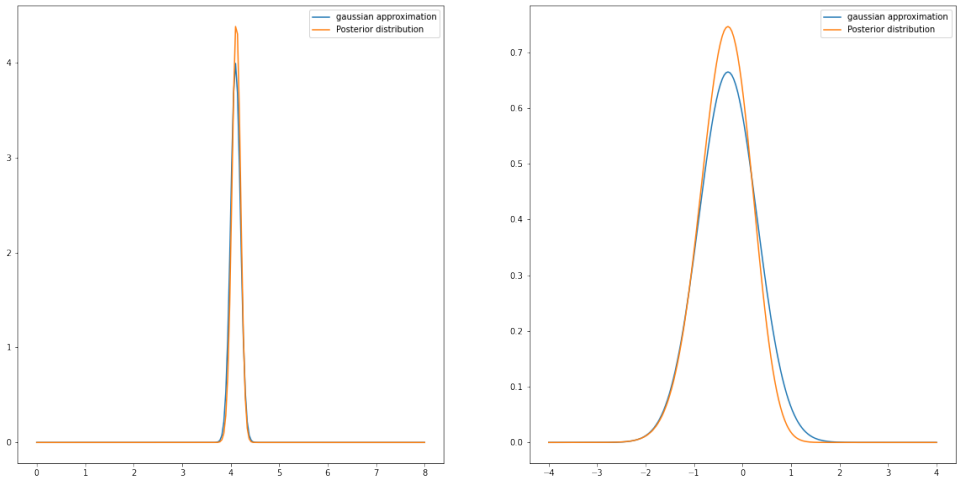

Figure 2: Two examples of Gaussian approximation in one dimension. On the left, the density is very spiked, on the right it is more smooth. We took $p = q = 1$ and used scipy integration (Virtanen et al., 2020) to compute the normalizing factor for similar scales in both plots.

$$\begin{aligned} J_1 &= \sum_i \mathbb{E}_q \left[\log p_\theta \left(Y_i \mid Z\right)\right] \\ &= \sum_i \mathbb{E}_q \left[\log p_\theta \left(Y_i \mid Z_i\right)\right] \\ &= \sum_{i,j} \mathbb{E}_q \left[\log p_\theta \left(Y_{ij} \mid Z_{ij}\right)\right] \\ &= \sum_{i,j} \mathbb{E}_q \left[Y_{ij}\left(o_{ij} + Z_{ij}\right) - \exp\left(o_{ij} + Z_{ij}\right)\right] + cst \end{aligned}$$

where *cst* denotes some amount that does not depend on the parameter θ or q.

Since under q, $Z_{ij} \sim \mathcal{N}(M_{ij}, S_{ij}^2)$.

$$\mathbb{E}_q \left[Z_{ij}\right] = M_{ij} \qquad \mathbb{E}_q \left[\exp\left(Z_{ij}\right)\right] = \frac{1}{2}\exp\left(M_{ij} + \frac{(S_{ij})^2}{2}\right)$$

So that

$$\begin{aligned} J_1 &= \sum_{i,j} Y_{ij}\left(o_{ij} + M_{ij}\right) - \frac{1}{2}\exp\left(o_{ij} + M_{ij} + \frac{(S_{ij})^2}{2}\right) + cst \\ &= \mathbf{1}_n^\top \left(Y \odot (O + M) - \frac{1}{2}\exp\left(O + M + \frac{S \odot S}{2}\right)\right)\mathbf{1}_p + cst \end{aligned}$$

Where we have denoted $M = (M_{ij})_{1 \leqslant i \leqslant n, 1 \leqslant j \leqslant p}$, $S = (S_{ij})_{1 \leqslant i \leqslant n, 1 \leqslant j \leqslant p}$., and $\mathbf{1}_n$ is a vector of dimension n full of ones. The exponential is applied component-wise on the last equation.

To compute (3), note that if $X \sim \mathcal{N}_N(\mu, \Sigma)$, and f its probability density, then

$$H(f) := \mathbb{E}_f \left[\log f(X)\right] = \log\left(\sqrt{(2\pi e)^N |\Sigma|}\right)$$

So that

$$\begin{aligned} J_3 = H(q) &= \sum_i H\left(q_i\right) \\ &= \sum_i \log\left(\sqrt{(2\pi e)^p \left|\mathrm{diag}(S_i \odot S_i)\right|}\right) \\ &= \frac{1}{2}\sum_i \log\left|\mathrm{diag}(S_i \odot S_i)\right| + cst \\ &= \frac{1}{2}\sum_{ij} \log S_{ij}^2 + cst \\ &= \frac{1}{2}\mathbf{1}_n^\top \log(S \odot S)\mathbf{1}_p + cst \end{aligned}$$

where the log is applied component-wise on the last equation.

$$\begin{aligned} J_2 = \mathbb{E}_q \left[\log p_\theta(Z)\right] &= \sum_i \mathbb{E}_q \left[\log p_\theta \left(Z_i\right)\right] \\ &= -\frac{n}{2}\log|\Sigma| + \sum_i \mathbb{E}_q \left[-\frac{1}{2}(Z_i - \beta^\top X_i)^\top \Sigma^{-1}(Z_i - \beta^\top X_i)\right] + cst \end{aligned}$$

We introduce $V \sim \mathcal{N}(\mu, \Lambda), \mu \in \mathbb{R}^p, \Lambda \in \mathcal{S}_p^{++}$ and we want to compute $\mathbb{E}\left[V^\top \Sigma^{-1} V\right]$ We denote $\Sigma^{-1/2}$ the square root Matrix of Σ^{-1}. It exists since $\Sigma^{-1} \in \mathcal{S}_p^{++}$.

$$\begin{aligned}
\mathbb{E}\left[V^\top \Sigma^{-1} V\right] &= \mathbb{E}\left[V^\top \Sigma^{-1/2}\Sigma^{-1/2} V\right] \\
&= \mathbb{E}\left[\left(\Sigma^{-1/2}V\right)^\top \left(\Sigma^{-1/2}V\right)\right] \\
&= \mathbb{E}\|\Sigma^{-1/2}V\|_2^2
\end{aligned}$$

Let $\tilde{V} = \Sigma^{-1/2}V, \quad \tilde{V} \sim \mathcal{N}\left(\Sigma^{-1/2}\mu, \Sigma^{-1/2}\Lambda\Sigma^{-\frac{1}{2}}\right)$

$$\begin{aligned}
\mathbb{E}\left[V^\top \Sigma^{-1} V\right] &= \mathbb{E}\|\tilde{V}\|_2^2 \\
&= \sum_j \mathbb{E}\tilde{V}_j^2 \\
&= \sum_j \mathrm{var}\left(\tilde{V}_j\right)^2 + \mathbb{E}\left[\tilde{V}_j\right]^2 \\
&= \sum_j \left(\Sigma^{-1/2}\Lambda\Sigma^{-1/2}\right)_{jj} + \left(\Sigma^{-1/2}\mu\right)_j^2 \\
&= \mathrm{tr}\left(\Sigma^{-1/2}\Lambda\Sigma^{-1/2}\right) + \sum_j \left(\left(\Sigma_{j,\cdot}^{-1/2}\right)^\top \mu\right)^2 \\
&= \mathrm{tr}\left(\Sigma^{-1}\Lambda\right) + \sum_j \left(\left(\Sigma_{j,\cdot}^{-1/2}\right)^\top \mu\right)^2
\end{aligned}$$

Since under q, $Z_i - X_i^\top \beta \sim \mathcal{N}(M_i - \beta^\top X_i, S_i \odot S_i)$

$$\begin{aligned}
J_2 &= -\frac{1}{2}\sum_i \mathrm{tr}\left(\Sigma^{-1}(S_i \odot S_i)\right) - \frac{1}{2}\sum_{i,j}\left(\left(\Sigma_{j,\cdot}^{-1/2}\right)^\top (M_i - \beta^\top X_i)\right)^2 - \frac{n}{2}\log|\Sigma| + cst \\
&= -\frac{1}{2}\mathrm{tr}\left(\Sigma^{-1}\left(\sum_i S_i \odot S_i\right)\right) - \frac{1}{2}\sum_{i,j}\left(\Sigma^{-1/2}(M - X\beta)\right)_{j,i}^\top \left(\Sigma^{-1/2}(M - X\beta)\right)_{i,j} - \frac{n}{2}\log|\Sigma| + cst \\
&= -\frac{1}{2}\mathrm{tr}\left(\Sigma^{-1}\left(\sum_i S_i \odot S_i\right)\right) - \frac{1}{2}\mathrm{tr}\left(\Sigma^{-1/2}(M - X\beta)^\top (M - X\beta)\Sigma^{-1/2}\right) - \frac{n}{2}\log|\Sigma| + cst \\
&= -\frac{1}{2}\mathrm{tr}\left(\Sigma^{-1}\left(\mathrm{diag}(\mathbf{1}_n^\top (S \odot S)) + (M - X\beta)^\top (M - X\beta)\right)\right) - \frac{n}{2}\log|\Sigma| + cst
\end{aligned}$$

We then have :

$$\begin{aligned}
J_{\theta,q}(Y) = \mathbf{1}_n^\top &\left(Y \odot (O + M) - \frac{1}{2}\exp\left(O + M + \frac{S \odot S}{2}\right) + \frac{1}{2}\log(S \odot S)\right)\mathbf{1}_p \\
&- \frac{1}{2}\mathrm{tr}\left(\Sigma^{-1}\left(\mathrm{diag}(\mathbf{1}_n^\top (S \odot S)) + (M - X\beta)^\top (M - X\beta)\right)\right) - \frac{n}{2}\log|\Sigma| + cst
\end{aligned} \tag{5}$$

This function is concave with respect to θ when (M, S) are fixed, and concave with respect to (M, S) when θ is fixed. See Chiquet et al. (2018) for a proof.

2.3.2.2 M and VE steps To derive the VE and M step, we thus need to compute the gradients and perform a gradient ascent if we do not have an analytical form.

The parameters of this function are (Σ, β, M, S). Taking S as parameter instead of $S^2 = S \odot S$ allows us to avoid constraints of positiveness.

The M-step is straightforward as we have analytical forms for β and Σ:

$$\Sigma^{(t+1)} = \frac{1}{n} \sum_i \left((M^{(t)} - X\beta)_i (M^{(t)} - X\beta)_i^\top + S_i^{(t)} \right)$$
$$\beta^{(t+1)} = (X^\top X)^{-1} X^\top M^{(t)}$$

We don't have any closed form for the VE step. Here are the gradients with respect to the variational parameters.

$$\nabla_M J = -M\Sigma^{-1} - \exp\left(O + M + \frac{S \odot S}{2}\right) + Y$$
$$\nabla_S J = -\frac{1}{2} \mathbf{1}_n \mathbf{1}_p^T D_{\Sigma^{-1}} - S \odot \exp\left(O + M + \frac{S \odot S}{2}\right) + \frac{1}{2} \frac{1}{S}$$

We denoted $D_{\Sigma^{-1}}$ the diagonal matrix composed of the diagonal of Σ^{-1}. The exponential is applied component-wise, as well as the division $\frac{1}{S}$.

2.3.3 Different parametrisations

We can define another model equivalent to (1) :

$$\begin{aligned} W_i &\sim \mathcal{N}(0, I_q) \\ Z_i &= CW_i \\ Y_{ij} \mid Z_{ij} &\sim \mathcal{P}\left(\exp\left(O_{ij} + \beta_{:,j}^\top X_i + Z_{ij}\right)\right) \end{aligned} \tag{6}$$

This model is stricly equivalent from a mathematical point of view, but different from an algorithmic point of view. Indeed, the variational approximation is done on the variable Z_i, and it can either be centered in 0 or $\beta^\top X_i$. In other words, the variational parameter M does not approximate the same mean. Here is the ELBO for Parametrisation (6) (the computation is the same as in Parametrisation (1)):

$$\begin{aligned} J_{\theta,q}(y) = &-\frac{n}{2} \log|\Sigma| - \frac{1}{2} \sum_i m_i^\top \Sigma^{-1} m_i + \operatorname{tr}\left(\Sigma^{-1} S_i\right) \\ &+ \sum_{i,j} -\frac{1}{2} \exp\left(o_{ij} + x_i^T \beta_j + m_{ij} + [S_i]_{jj}^2 / 2\right) + Y_{ij}\left(o_{ij} + x_i^T \beta_j + m_{ij}\right) \\ &+ \frac{1}{2} \sum_i \log|(S \odot S)_i| + cst. \end{aligned} \tag{7}$$

One major difference that comes up is the non-existence of an analytic form for β. Thus, if we chose the parametrisation centered in 0_p, the M-step must be done with a (costly) gradient ascent. We saw here two differents parametrisations, but we can also change the

arguments of the ELBO as (C, β, M, S) instead of (Σ, β, M, S). If we do so, we can have another parametrisation :

$$\begin{aligned} Z_i &\sim \mathcal{N}(0_p, I_p) \\ Y_{ij} \mid Z_i &\sim \mathcal{P}\left(\exp\left(o_{ij} + x_i^\top \beta_j + C_j^\top Z_i\right)\right) \end{aligned} \tag{8}$$

This parametrisation does not admit any analytical forms.

2.3.4 Analytical forms importance and Gradient Ascent

Depending on the parametrisation and the parameter choice, we can have none, one or two analytical forms. We want to find the faster and more accurate one.

The function defined in (5) is only jointly concave. We can show that it is the case for any parametrisation. This being said, we did not know if we would find a maximum and if we do, would it be a global one ? Playing with all the different parametrisations, we were hoping to find one that finds better local maxima. After some tests, we found out that a rigorous VEM was quite inefficient. Doing each VE and M step very accurately is not productive. Doing between 1 and 15 gradient iterations for each step is enough, we do not need to spend precious time finding the maximum at each step, a rough estimate works well. As a result, we added an hyper-parameter that controls the number of gradient iterations we do at each step.

One interesting model would consists in updating all parameters simultaneously, completely forgetting the alternation of steps. We believed that the smooth trajectory of a gradient ascent would be more efficient. In other words, we believed that the analytical forms updates would be too jerky and end up in a bad local maxima.

2.3.5 Framework used

We used Pytorch Framework (Paszke et al., 2017) to compute the gradients using automatic differentiation. However we also implemented the gradients ourselves as a sanity check. Once the algorithms were implemented, we needed an optimiser, such as the Stochastic Gradient Descent. There exists more sophisticated optimisers that are faster than others, depending on the data. We had to find the better one. We tried every optimiser of Pytorch and found out that Rprop (Riedmiller and Braun, 1993) was the faster one for every parametrisation.

2.4 Data simulation and evaluation

2.4.1 Data simulation

We simulated data according to (1). The latent variables are thus available since simulated, which is not the case in practice:

$$Z_i \sim \mathcal{N}(\beta^\top X_i, CC^\top)$$

The Maximum Likelihood Estimator (MLE) in the latent layer of the model parameters, namely $\beta_{MLE,\text{lat}}$ and $\Sigma_{MLE,\text{lat}}$, are thus available. Here are the explicit expressions :

$$\beta_{MLE,\text{lat}} = (X^\top X)^{-1} X^\top Z$$

$$\Sigma_{MLE,\text{lat}} = \frac{1}{n}\sum_{i=1}^{n}(Z_i - \mu_{MLE})(Z_i - \mu_{MLE,\text{lat}})^\top$$

where $\mu_{MLE,\text{lat}} := \frac{1}{n}\sum_{i=1}^{n} Z_i$. The simulated data coming from (1) is only noisy latent variables. Thus maximising the likelihood of the simulated data will probably end up with parameters further from the true parameter than $(\beta_{MLE,\text{lat}}, \Sigma_{MLE,\text{lat}})$. Moreover, the exponential makes it more difficult to distinguish low latent variables values due to the exponential. For example, let $1 \leqslant i \leqslant n, 1 \leqslant j_0 < j_1 \leqslant p$:

$$\mathbb{P}(Y_{i,j_0} = 0, Y_{i,j_1} = 0 | Z_{i,j_0} = -3, Z_{i,j_1} = -12) = \exp(-\exp(-3) - \exp(-12)) \approx 0.95$$

However, the accuracy is improved for large latent variables values. For example, it is easier to distinguish 3 and 3.5 after an exponential transformation :

$$\mathbb{E}[Y_{ij}|Z_{ij} = 3] \approx 20$$
$$\mathbb{E}[Y_{ij}|Z_{ij} = 3.5] \approx 33$$

2.4.2 Evaluation of performances

We choose the Mean Squared Error (MSE) between our estimate $\hat{\theta}$ and the true parameters $\theta^\star$ to evaluate the estimates :

$$MSE(\beta^\star - \hat{\beta}) = \frac{1}{dp}\sum_{k=1}^{d}\sum_{j=1}^{p}(\beta^\star_{kj} - \hat{\beta}_{kj})^2$$

$$MSE(\Sigma^\star - \hat{\Sigma}) = \frac{1}{p^2}\sum_{k=1}^{p}\sum_{j=1}^{p}(\Sigma^\star_{kj} - \hat{\Sigma}_{kj})^2$$

Note that when the data is not simulated, we cannot compute those errors since we don't know $\theta^\star = (\beta^\star, \Sigma^\star)$. A more natural metric would be the log likelihood. we cannot compute it but we can estimate it, as we will see later in 18.

2.5 Results and discussion

We discriminate two models comparing their ELBO. We tried 6 different models in total. Note that only the model number 1 does the M step perfectly, as it updates the model parameters with an analytical form. Also, the number 2 and 5 update the four parameters alltogether, forgeting the alternation of VE and M step. The other ones uses between 1 and 15 gradient steps for each M and VE steps. Here is the summary of our trials :

Parametrisation	Model ID	Mean of the latent variable	Variance of the latent variables	Argument for the variance	Analytical form for β	Analytical form
1	both closed, $X\beta, \Sigma$	$X\beta$	$CC^\top$	Σ	✓	✓
	No closed, $X\beta, C$	$X\beta$	$CC^\top$	C	×	×
	Σ closed, $0, \Sigma$	0	$CC^\top$	Σ	✓	×
6	Σ closed, $X\beta, \Sigma$	$X\beta$	$CC^\top$	Σ	✓	×
	No closed, $0, I_p$	0	I_p	C	×	×
8	β closed, $X\beta, C$	$X\beta$	$CC^\top$	C	×	✓

We launched all the model first for $n = 1000, p = q = 200$ to compare them. We first needed to choose the right hyperparameters for each model such as :

- optimiser
- Learning rate
- Number of gradients steps required for each M and VE steps (if needed).

Fig 3 displays the result of the algorithm for each optimiser. The optimiser choice is straightforward since we found that Rprop (Riedmiller and Braun, 1993) was much faster than the other ones for every model. We then performed a grid search to find both other best hyperparameters.

Fig 4 displays the result of the algorithm for each parametrisation. Looking at the first two rows, we see clearly two groups. It turns out that using a parametrisation centered in $X\beta$ is much more efficient. Every model that uses this parametrisation converges in more or less the same amount of time.

On the one hand, bad estimation of the parameter does not mean low ELBO. Indeed, we can see a much higher ELBO for some models ("Σ closed, $0, \Sigma$" and "No closed, $0, I_p$" at about 55 s) than other ("No closed, $X\beta, C$", "Σ closed, $X\beta, \Sigma$" and " β closed, $X\beta, C$ " at about 4s), whereas the MSE is much lower for the second group. On the other hand, if models share the same parametrisation, then high ELBO means good estimation of the parameters. This means that we can compare the ELBO between models that shares the same parametrisation.

As we can see, using analytical forms for both Σ and β performs very well. The one that uses a gradient ascent on all the parameters and thus no analytical forms performs badly. We were expecting the opposite. On (a) we see clearly the analytical updates, since "Σ closed, $X\beta, \Sigma$" does a big step towards the true parameter. It is also true for β in (b) for "β closed, $X\beta, C$". As a result, "both closed, $X\beta, \Sigma$" is very fast since doing both big steps. It does not fall into bad local maxima. Note that if we launch the models for 500 more seconds, the parameters of each model converges close to the true parameter. We launch the same models for $n = 10000$ and $p = 2000$, but most of them were not very robust to large p, so we decided to plot only the one that performs best (Fig 6) ("Σ closed, $X\beta, \Sigma$"). "both closed, $X\beta, \Sigma$" performs very well for large n and large p as we can see on Fig 6. It reaches rapidly a local maximum, close to the MLE in the latent layer. The time of convergence is about a hundred seconds, which is great for this amount of data. Note that a GPU has been used.

2.6 PCA parametrisation

When p becomes large, computing and inverting a $p \times p$ matrix becomes costly. Switching to a PCA parametrisation with $q < p$ seems to be a good idea to bypass this issue. However, from all the models, only "No closed, $0, I_p$" is easily adaptable to the PLN-PCA case ($p < q$). Indeed, all the other ones requires $CC^\top$ invertible, and thus preventing us to compute the log of the determinant (such as in (5)) since $\Sigma = CC^\top$ has rank at most q and is thus not invertible. Unfortunately, this model converges to the true parameter very slowly, even when p is low. We thus need to change the inference approaches. Moreover, the variational approach does not try to infer the MLE, but something close to the MLE (in practice, since we don't have any theoretical results of convergence for variational inference). Thus, we cannot have any confidence intervals. Looking for the MLE would allow us to do so.

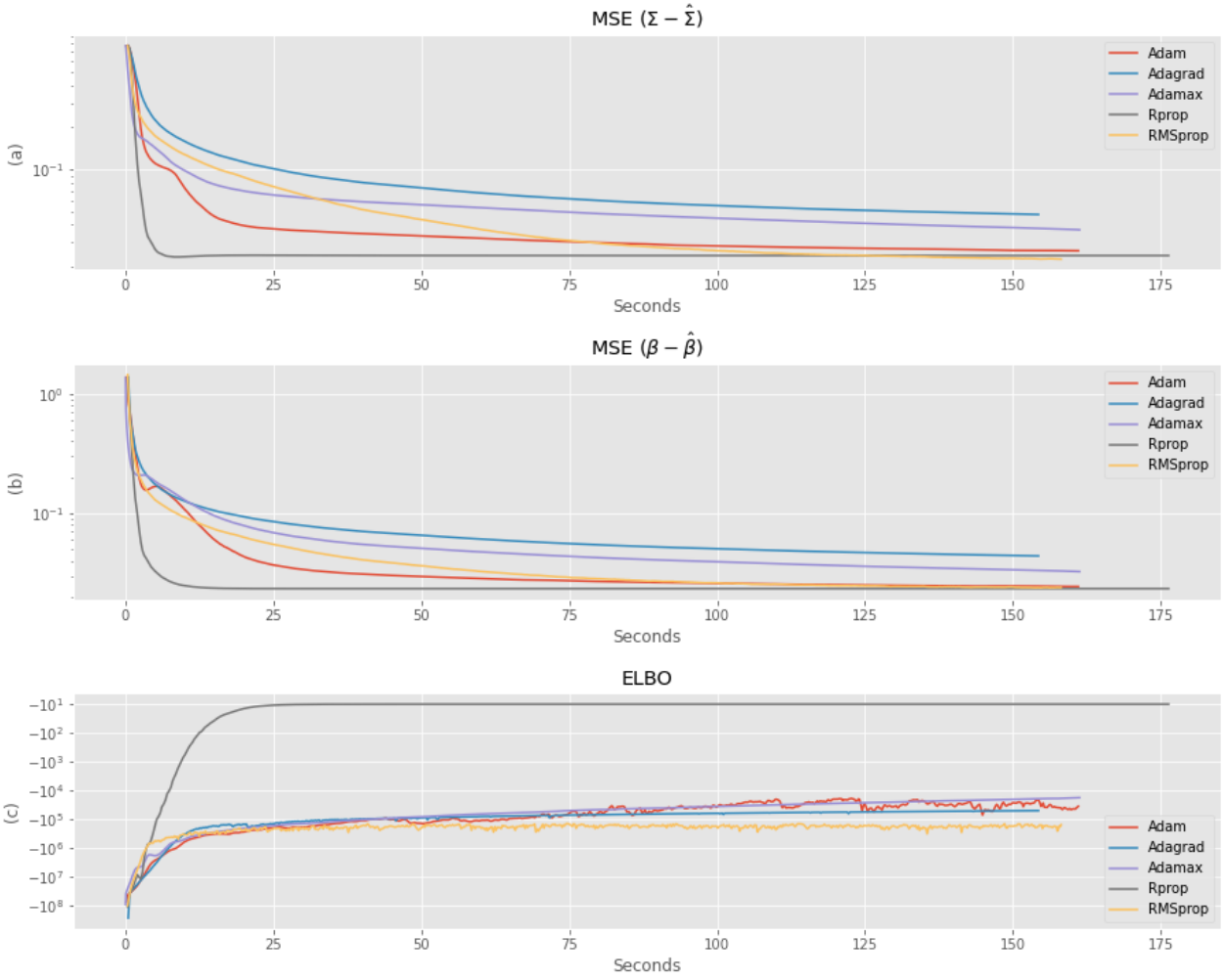

Figure 3: Simulation with $n = 1000, p = q = 200, d = 2$ with the parametrisation 'both closed, $X\beta, \Sigma$' with different optimiser.The learning rate of each optimiser has been tuned. We see that Rprop performs better than the others.

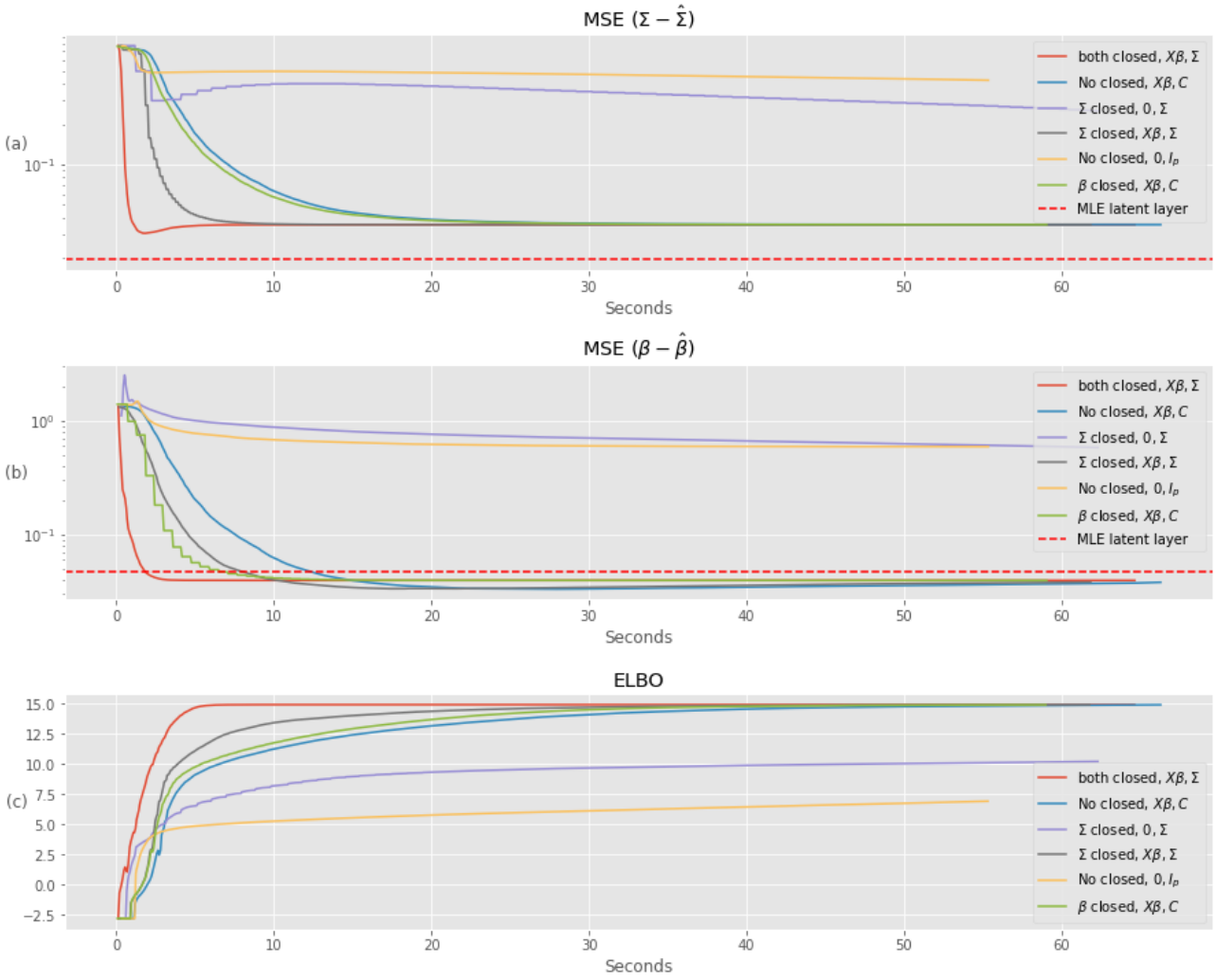

Figure 4: Comparison of all the different models for simulated data ($n = 1000, p = 200, d = 2$). We took blockwise symmetric matrices for Σ (see Fig 5), and Gaussians random numbers for β. (a) and (b) show the error evolution evolution with respect to the model parameters. (c) shows the ELBO, the only metric available in practice. We applied a log transformation for a clear plot. A GPU has been used.

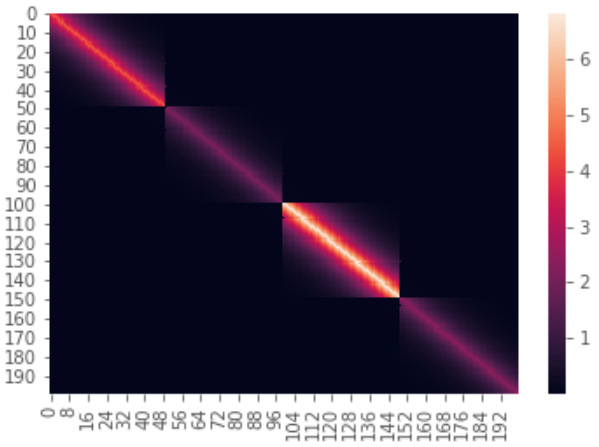

Figure 5: True parameter Σ took for the simulation of the data. In practice the matrix contains blocks, similar to this matrix.

3 Gradient methods

In this section, we will try to infer the MLE, that is :

$$\underset{\theta}{\operatorname{argmax}} \log p_\theta(Y) = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} \log p_\theta(Y_i) \tag{9}$$

We will infer the MLE using gradient methods, based on the estimation of the gradients of the objective function.

3.1 Gradient computation

We set

$$\begin{aligned} \tilde{p}_\theta \ : \ & \mathbb{R}^q \to \mathbb{R}^+ \\ & W_i \mapsto p_\theta(W_i|Y_i) \end{aligned}$$

the posterior law. In the following, $\|.\|$ denotes $\|.\|_2$.

Proposition 3.1. *The gradient of the log likelihood has the following form :*

$$\nabla_\theta \log p_\theta(Y_i) = \begin{pmatrix} X_i \left[Y_i - \exp\left(O_i + \beta^\top X_i\right) \mathbb{E}_{\tilde{p}_\theta}\left[e^{CW_i}\right]\right]^\top \\ Y_i \mathbb{E}_{\tilde{p}_\theta}\left[W_i^\top\right] - \exp(O_i + \beta^\top X_i) \mathbb{E}_{\tilde{p}_\theta}\left[\exp\left(CW_i\right) W_i^\top\right] \end{pmatrix}$$

Proof 3.1. *We assume we can swap integral and gradient in the following computation.*

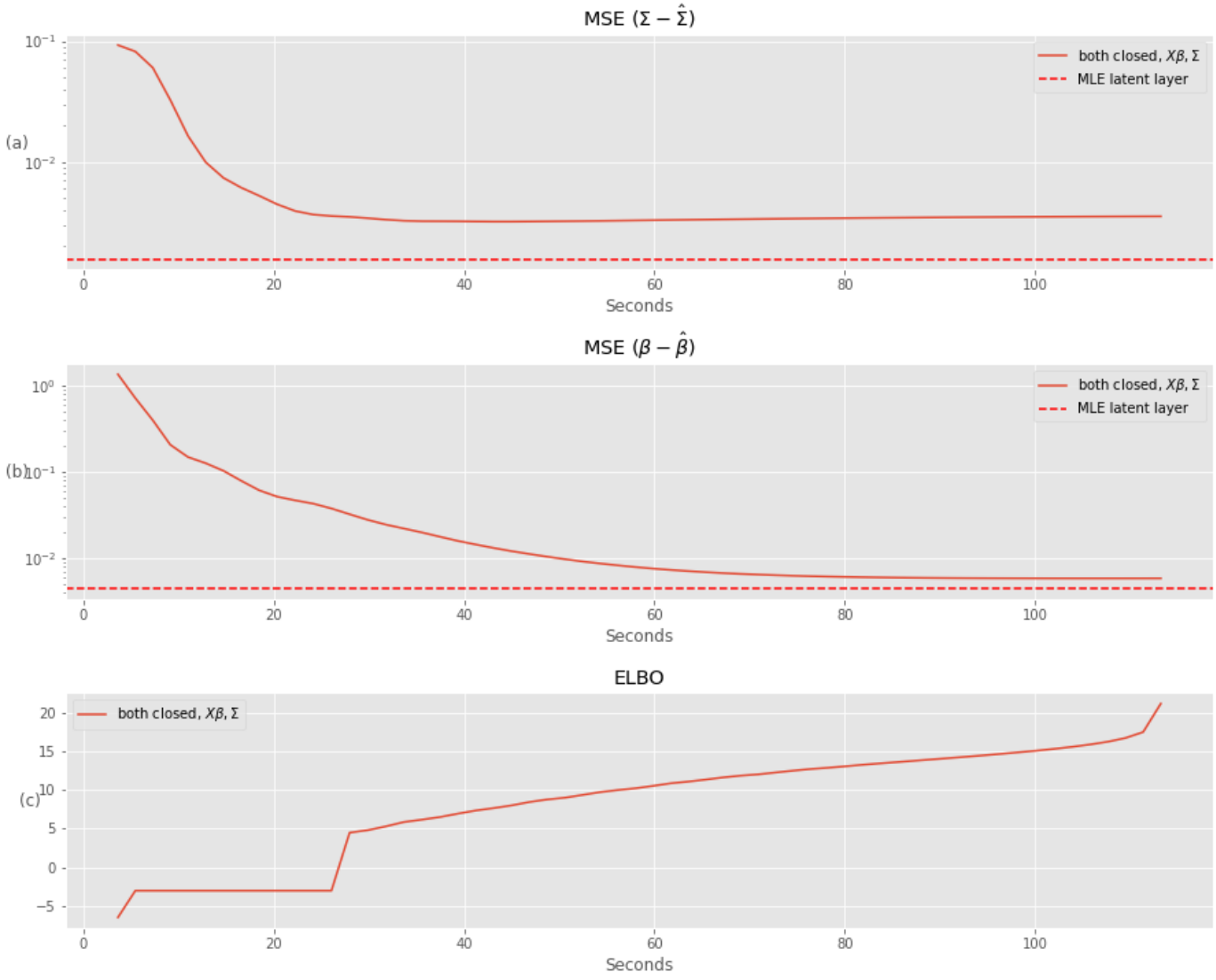

Figure 6: Simulation for $n = 10000, p = 2000$ for "Σ closed, $X\beta, \Sigma$". A log transformation has been applied on (c).

$$\begin{aligned}
\nabla_\theta \log p_\theta(Y_i) &= \frac{\nabla_\theta p_\theta(Y_i)}{p_\theta(Y_i)} \\
&= \frac{\nabla_\theta \int p_\theta(Y_i \mid W_i) p(W_i) dW}{p_\theta(Y_i)} \\
&= \frac{\int \nabla_\theta p_\theta(Y_i|W_i) p(W_i) dW_i}{p_\theta(Y_i)} \\
&= \frac{\int (\nabla_\theta \log p_\theta(Y_i \mid W_i))\, p_\theta(Y_i|W_i) p(W_i) dW_i}{p_\theta(Y_i)} \\
&= \int (\nabla_\theta \log p_\theta(Y_i \mid W_i)) \frac{p_\theta(Y_i|W_i) p(W_i) dW_i}{p_\theta(Y_i)} \\
&= \int \nabla_\theta \log p_\theta(Y_i \mid W_i) \tilde{p}_\theta(W_i) dW_i \\
&= \mathbb{E}_{\tilde{p}_\theta} \left[\nabla_\theta \log p_\theta(Y_i \mid W_i) \right]
\end{aligned}$$

We used the Bayes rules at the second last line. The gradients with respect to θ is only the concatenation of the gradients with respect to β and C.

$$\begin{aligned}
\nabla_\theta \log p_\theta(Y_i) &= \begin{pmatrix} \nabla_\beta \log p_\theta(Y_i) \\ \nabla_C \log p_\theta(Y_i) \end{pmatrix} \\
&= \begin{pmatrix} \mathbb{E}_{\tilde{p}_\theta} \left[\nabla_\beta \log p_\theta(Y_i \mid W_i) \right] \\ \mathbb{E}_{\tilde{p}_\theta} \left[\nabla_C \log p_\theta(Y_i \mid W_i) \right] \end{pmatrix}
\end{aligned}$$

Let us justify the swap between integral and gradients. We need to justify two swaps :

$$\nabla_\beta \int p_\theta(Y_i \mid W_i) p(W_i) dW_i = \int \nabla_\beta p_\theta(Y_i \mid W_i) p(W_i) dW_i \tag{10}$$

$$\nabla_C \int p_\theta(Y_i \mid W_i) p(W_i) dW_i = \int \nabla_\theta p_C(Y_i \mid W_i) p(W_i) dW_i \tag{11}$$

Now, note that

$$p_\theta(Y_i \mid W_i) p(W_i) = \tilde{C} \exp\left(-\frac{1}{2} \|W_i\|^2 - \mathbf{1}_p^\top \exp(O_i + \beta^\top X_i + C W_i) + Y_i^\top (O_i + \beta^\top X_i + C W_i) \right)$$

where $\tilde{C} \in \mathbb{R}^+$. We will use the dominated convergence theorem to justify the swap. Since $\theta \mapsto p_\theta(Y_i \mid W) p(W)$ is C^1 for all $W \in \mathbb{R}^q$ (all the functions are either linear or exponential),to justify (10) (the same is true for (11), substituting β with C in the following inequation), we only need to prove the existence of a function $h \in L^1$ such that :

$$p_\theta(Y_i \mid W) p(W) \leqslant h(W) \ \forall \beta, W$$

$$\begin{aligned}
\log(p_\theta(Y_i|W) p(W)) + \frac{1}{2}\|W\|^2 &= \log(\tilde{C}) - \mathbf{1}_p^\top \exp(O_i + \beta^\top X_i + CW) + Y_i^\top (O_i + \beta^\top X_i + CW) \\
&= \log(\tilde{C}) + \sum_{j=1}^{p} Y_{ij} (O_{ij} + X_i^\top \beta_{.j} + C_j^\top W) - \exp(O_{ij} + X_i^\top \beta_{.j} + C_j^\top W)
\end{aligned}$$

For $\alpha \geqslant 0$, the function $x \mapsto \alpha x - \exp(x)$ being bounded from above on $\mathbb{R}$ by some constant K_α , we get (setting $x = O_{ij} + X_i^\top \beta_{:,j} + C_j^\top W, \alpha = Y_{ij}$) :

$$p_\theta(Y_i|W) p(W) \leqslant \tilde{C} \exp(\sum_{j=1}^{p} K_{Y_{ij}} - \frac{1}{2}\|W\|^2) \in L^1$$

so that (10) (and (11)) holds.

Now, we need to derive the log likelihood of Y_i given W_i in order to compute its gradient and then the expectation. First,

$$\begin{aligned} p_\theta\left(Y_i \mid W_i\right) &= \prod_{j=1}^{p} p_\theta\left(Y_{ij} \mid W_i\right) \\ &= \prod_{j=1}^{p} \frac{1}{Y_{ij}!} \exp\left(-\exp\left(O_{ij}+Z_{ij}\right)\right) \exp\left(O_{ij}+Z_{ij}\right)^{Y_{ij}} \end{aligned}$$

So that the log likelihood given the latent variable is the following :

$$\log p_\theta\left(Y_i \mid W_i\right) = \sum_{j=1}^{p} -\log\left(Y_{ij}!\right) - \exp\left(0_{ij}+Z_{ij}\right) + Y_{ij}\left(O_{ij}+Z_{ij}\right)$$

Let's compute the gradient with respect to β. We see the vectors of size n as a matrix of dimension $(n,1)$.

We set $A = O_i + CW_i$ and

$$h : \beta \mapsto \mathbf{1}_p^\top \exp\left(\beta^\top X_i + A\right)$$

Let $1 \leqslant k \leqslant d, 1 \leqslant l \leqslant p$

$$\begin{aligned} \frac{\partial h}{\partial \beta_{kl}}(\beta) &= \frac{\partial}{\partial \beta_{kl}}\left(\sum_j \exp\left(X_i^\top \beta_{.j} + A_j\right)\right) \\ &= \frac{\partial}{\partial \beta_{k,l}} \exp\left(X_i^\top \beta_{.l} + A_l\right) \\ &= \frac{\partial}{\partial \beta_{kl}} \exp\left(A_l + \sum_s X_{is}\beta_{sl}\right) \\ &= \left(\frac{\partial}{\partial \beta_{kl}}\left(A_l + \sum_s X_{is}\beta_{sl}\right)\right) \exp\left(A_l + \sum_s X_{is}\beta_{sl}\right) \\ &= X_{ik} \exp\left(A_l + \sum_s X_{is}\beta_{sl}\right) \\ &= X_{ik} \exp\left(X_i^\top \beta_{.l} + A_l\right) \\ &= \left(X_i \exp\left(\beta^\top X_i + A\right)^T\right)_{kl} \end{aligned}$$

So that

$$\nabla_\beta h = X_i \exp\left(O_i + \beta^\top X_i + CW_i\right)^\top$$

where the exponential is applied component-wise.

A similar computation for

$$\tilde{h} : \beta \mapsto \sum_j Y_{ij}\left(O_{ij} + X_i^\top \beta_{:,j} + W_i C_{:,j}^\top\right)$$

shows that

$$\nabla_\beta \tilde{h} = X_i Y_i^\top$$

So that :

$$\nabla_\beta \log p_\theta\left(Y_i \mid W_i\right) = \nabla_\beta \tilde{h} - \nabla_\beta h = X_i\left[Y_i - \exp\left(O_i + \beta^\top X_i + CW_i\right)\right]^\top$$

A similar computation for C shows that :

$$\nabla_C \log p_\theta\left(Y_i \mid W_i\right) = \left[Y_i - \exp\left(O_i + \beta^\top X_i + CW_i\right)\right] W_i^\top$$

Now,

$$\begin{aligned}
\nabla_\theta \log p_\theta(Y_i) &= \begin{pmatrix} \mathbb{E}_{\tilde{p}_\theta}\left[\nabla_\theta \log p_\theta(Y_i \mid W_i)\right] \\ \mathbb{E}_{\tilde{p}_\theta}\left[\nabla_\theta \log p_\theta(Y_i \mid W_i)\right] \end{pmatrix} \\
&= \begin{pmatrix} \mathbb{E}_{\tilde{p}_\theta}\left[X_i\left[Y_i - \exp\left(O_i + \beta^\top X_i + CW_i\right)\right]^\top\right] \\ \mathbb{E}_{\tilde{p}_\theta}\left[\left[Y_i - \exp\left(O_i + \beta^\top X_i + CW_i\right)\right] W_i^\top\right] \end{pmatrix} \\
&= \begin{pmatrix} X_i\left[Y_i - \exp\left(0_i + \beta^\top X_i\right) \mathbb{E}_{\tilde{p}_\theta}\left[e^{CW_i}\right]\right]^\top \\ Y_i \mathbb{E}_{\tilde{p}_\theta}\left[W_i^\top\right] - \exp(O_i + \beta^\top X_i) \mathbb{E}_{\tilde{p}_\theta}\left[\exp\left(CW_i\right) W_i^\top\right] \end{pmatrix} \quad (12)
\end{aligned}$$

which concludes the proof.

Thus, computing the gradients boils down to compute those 3 expectations

$$\mathbb{E}_{\tilde{p}_\theta}\left[e^{CW_i}\right] \qquad \mathbb{E}_{\tilde{p}_\theta}\left[\exp\left(CW_i\right) W_i^\top\right] \qquad \mathbb{E}_{\tilde{p}_\theta}\left[W_i^\top\right] \tag{13}$$

Each of these expectations are multi-dimensional. The dimensions are p, (p, q) and q respectively. We look closer to the first integral as an example :

$$\mathbb{E}_{\tilde{p}_\theta}\left[e^{CW_i}\right] = \begin{pmatrix} \mathbb{E}_{\tilde{p}_\theta}\left[e^{(CW_i)_1}\right] \\ \vdots \\ \mathbb{E}_{\tilde{p}_\theta}\left[e^{(CW_{i,k})_p}\right] \end{pmatrix} = \left(\mathbb{E}_{\tilde{p}_\theta}\left[e^{u_m^\top CW_i}\right]\right)_{1 \leqslant m \leqslant p}$$

where u_m is the m^{th} canonical vector of $\mathbb{R}^p$. Each of those expectations are intractable, once again because of the unknown posterior $\tilde{p}_\theta$. We want to use Monte Carlo simulation to get an estimate but we neither know how to sample from the posterior nor its density. What we do know is the unnormalized density $\tilde{p}_\theta^{(u)}$:

$$\begin{aligned}
\tilde{p}_\theta^{(u)} \;:\; & \mathbb{R}^q \to \mathbb{R}^+ \\
& W \mapsto p_\theta(Y_i|W)p(W)
\end{aligned}$$

We can use the self-normalize importance sampling to get an estimate of this integral, which we introduce in the following section.

3.2 Importance sampling

3.2.1 Basic importance sampling

Let ϕ be a measurable function. Let $\tilde{p}_\theta$ and g be two probability densities such that $\text{supp}(g) = \text{supp}(\tilde{p}_\theta) = \mathbb{R}^q$. We want to find $\mu = \mathbb{E}_{\tilde{p}_\theta}[\phi(X)]$.

$$\mu = \int_{\mathbb{R}^q} \phi(x)\tilde{p}_\theta(x)\mathrm{d}x = \int_{\mathbb{R}^q} \frac{\phi(x)\tilde{p}_\theta(x)}{g(x)} g(x)\mathrm{d}x = \mathbb{E}_g\left(\frac{\phi(X)\tilde{p}_\theta(X)}{g(X)}\right)$$

Let $n_s \in \mathbb{N}$, n_s is called the sampling efforts. We denote $(V_k)_{1 \leqslant k \leqslant n_s} \overset{iid}{\sim} g$.

We set

$$w(X) := \frac{\tilde{p}_\theta(X)}{g(X)},\ X \in \mathbb{R}^q$$

$w(V_k)$ is called the weight associated to sample V_k. Note that

$$\mathbb{E}_g[w(X)] = \int_{\mathbb{R}^q} \frac{\tilde{p}_\theta(x)}{g(x)} g(x)\mathrm{d}x = \int_{\mathbb{R}^q} \tilde{p}_\theta(x)\mathrm{d}x = 1$$

The estimate of μ is

$$\hat{I}_{n_s,g} = \frac{1}{n_s}\sum_{k=1}^{n_s} w(V_k)\phi(V_k)$$

Since $\mathbb{E}_g[\phi(X)w(X)] = \mu$, from the large law of numbers :

$$\hat{I}_{n_s,g} \xrightarrow[n_s\to+\infty]{\text{a.s}} \mu$$

The importance law g is very important since the variance of the estimator is directly linked to it : $\text{var}(\hat{I}_{n_s,g}) = \frac{\sigma_g^2}{n_s}$ where

$$\sigma_g^2 = \int_{\mathbb{R}^q} \frac{(\phi(x)\tilde{p}_\theta(x))^2}{g(x)}\mathrm{d}x - \mu^2$$

Choosing the law g is the difficult part. How do we choose a good importance law ? Assuming ϕ to be non negative, we find one optimal law.

$$g_{opt}(x) = \frac{\phi(x)\tilde{p}_\theta(x)}{\mu}, x \in \mathbb{R}^q$$

The estimator based on this law is called the zero variance estimator since the variance of the estimator associated to g_{opt} is null. Indeed,

$$\begin{aligned}\sigma_{g_{opt}}^2 &= \int_{\mathbb{R}^q} \frac{(\phi(x)\tilde{p}_\theta(x))^2}{g_{opt}(x)}\mathrm{d}x - \mu^2 \\ &= \mu\int \phi(x)\tilde{p}_\theta(x)\mathrm{d}x - \mu^2 \\ &= \mu^2 - \mu^2 = 0\end{aligned}$$

Finding such a law is difficult since the denominator is what we are trying to reach. However, we can try to find a law that is close to this. If ϕ has non constant sign, then one can still get a zero variance estimator. We only need to get the zero variance estimator for

$$\begin{aligned}\phi_+(x) &= \max(\phi(x), 0) \\ \phi_-(x) &= \max(-\phi(x), 0)\end{aligned}$$

which are non negative and then using

$$\mathbb{E}_{\tilde{p}_\theta}\left[\phi(X)\right] = \mathbb{E}_{\tilde{p}_\theta}\left[\phi_+(X)\right] - \mathbb{E}_{\tilde{p}_\theta}\left[\phi_-(X)\right]$$

3.2.2 Self-normalized importance sampling

Basic importance sampling is not always reachable, especially when we don't know $\tilde{p}_\theta$ but only an unnormalized version of it, $\tilde{p}_\theta^{(u)}(x) = C \times \tilde{p}_\theta(x)$, where $C = \int_{\mathbb{R}^q} \tilde{p}_\theta^{(u)}(x)\mathrm{d}x$. Estimating this constant is an alternative that allows us to use basic importance sampling. However, we can avoid this estimation and use the self-normalized importance sampling. We define for $1 \leqslant k \leqslant n_s$:

$$w_k^{(u)} = \frac{\tilde{p}_\theta^{(u)}(V_k)}{g(V_k)} \tag{14}$$

$$\tilde{w}_k^{(u)} = \frac{w_k^{(u)}}{\sum_{\ell=1}^{n_s} w_\ell^{(u)}} \tag{15}$$

Note that mutliplying all the $w_k^{(u)}$ by some constant in (14) does not change the result since this constant vanishes when we take the ratio in (15). Then,

$$\hat{I}^u_{n_s,g} := \sum_{k=1}^{n_s} \tilde{w}_k^{(u)} \phi(V_{i,k}) \xrightarrow[n_s \to +\infty]{\mathbb{P}} \int \phi(W) \tilde{p}_\theta(W) \mathrm{d}W = \mathbb{E}_{\tilde{p}_\theta}[\phi(W)]$$

See Owen (2013) chapter 9 page 8 for a proof.
This estimator is not unbiased in general. Indeed,

$$\begin{aligned}
\mathbb{E}_g[\hat{I}^u_{n_s,g}] = \mathbb{E}_g\left[\sum_{k=1}^{n_s} \tilde{w}_k^{(u)} \phi(V_k)\right] &= n_s \mathbb{E}_g\left[\frac{w(V_1)\phi(V_1)}{\sum_{k=1}^{n_s} w_k(V_k)}\right] \\
&\neq n_s \frac{\mathbb{E}_g[w_k(V_1)\phi(V_1)]}{\mathbb{E}_g[\sum_{k=1}^{n_s} w_k(V_k)]} \\
&= \mathbb{E}_g[w_k(V_1)\phi(V_1)] \\
&= \mu
\end{aligned}$$

The variance of the estimator is given by

$$\begin{aligned}
\mathrm{var}\left(\hat{I}^u_{n_s,g}\right) = \frac{1}{n_s} \frac{\mathbb{E}_q\left[(\phi(X)w(X) - \mu w(X))^2\right]}{\mathbb{E}_g\left[w(X)\right]^2} &= \frac{\sigma^2_{g,\mathrm{sn}}}{n_s} \\
\sigma^2_{g,\mathrm{sn}} = \mathbb{E}_g\left[w(X)^2(\phi(X) - \mu)^2\right]&
\end{aligned}$$

We used $\mathbb{E}_g[w(X)] = 1$.The optimal density has the form

$$g_{\mathrm{opt},sn}(x) = \frac{|\phi(x) - \mu|\tilde{p}_\theta(x)}{\int_{\mathbb{R}^q} |\phi(x) - \mu|\tilde{p}_\theta(x)\mathrm{d}x}, \ x \in \mathbb{R}^q$$

Unfortunately, this density does not give a zero variance estimator. It is even more difficult to get close to this optimal density since we don't know the numerator (we know the numerator of g_{opt} but do not know the numerator of $g_{\mathrm{opt,\ sn}}$). However, we will be able to get a rough estimate. Note that taking g_{opt} instead of $g_{\mathrm{opt,sn}}$ in self-normalized importance sampling does not change much the variance, so that getting a rough estimate will be enough.

3.2.3 Importance law choice

As we said, the main difficulty in importance sampling is to find an importance law which yields a zero variance estimator. We know which density to look for but we don't know how to sample from it. However, we know that the posterior is well gaussian approximated, it stands to reason that the target density $g^\star$ too (the target density is either $g_{\mathrm{opt},sn}$ in self-normalize importance sampling or g_{opt} in basic importance sampling). We will thus sample from a gaussian distribution, but how do we choose the mean and covariance ? For the mean, we will take the mode of the target density (i.e. the argmax of the density). We will find it through optimization techniques. For the variance, we will take (minus) the inverse of the hessian of the log target density, evaluated in the mode (this process has been found in Owen (2013) chapter 9). If $g^\star$ has mode m, then

$$\Sigma^{\star -1} = -\frac{\partial^2}{\partial x \partial x^\top} \log g(x) \mid_{x=m}$$

Indeed, if q is a gaussian density with mean μ and covariance Σ, then taking the hessian of the log density gives (minus) the inverse covariance matrix :

$$-\frac{\partial^2}{\partial x \partial x^\top} \log q(x) = -\frac{\partial^2}{\partial x \partial x^\top}\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right) = \Sigma^{-1}$$

If the density is almost gaussian, we can expect that it would be a good approximation when evalutated in the mode.

3.3 Application

We can use self-importance sampling to estimate the integrals in (12). We need to estimate $\mathbb{E}_{\tilde{p}_\theta}[\phi(W)]$ for ϕ that takes the following forms

1. $\phi_{1,m}(x) = \exp(u_m^\top C x), \quad 1 \leqslant m \leqslant p$
2. $\phi_{2,m,l}(x) = \exp(u_m^\top C x) u_l^\top x, \quad 1 \leqslant m \leqslant p, 1 \leqslant l \leqslant q$
3. $\phi_{3,l}(x) = u_l^\top x, \quad 1 \leqslant l \leqslant q$

where $x \in \mathbb{R}^q$

3.3.1 Estimation of μ

When trying to reach the optimal density $g_{\text{opt},sn}$, we need to know μ, or at least get an estimate. If $g_{\text{opt},\mathcal{N}}$ is a gaussian that is close to $\tilde{p}_\theta$, we know that :

$$\mu = \mathbb{E}_{\tilde{p}_\theta}[\phi(X)] \approx \mathbb{E}_{g_{\text{opt},\mathcal{N}}}[\phi(X)]$$

To get a good gaussian estimate we will do as in 3.2.3 for p_θ (i.e. taking the mode of the unnormalized density for the mean and the hessian of the log density evaluated in the mode for the covariance matrix).

3.3.2 Gradient ascent of the log likelihood

We can now estimate the integrals, compute the gradients of the log likelihood and perform a gradient ascent. The resulting algorithm is quite costly. Indeed, we need to estimate $(p+1)(q+1)$ integrals. For each integral, we need to perform a gradient ascent on the likelihood to reach the mode and compute the covariance matrix, which is very costly. We simulated data according to 1 for $n = 100, p = 4, q = 3, d = 2$. Fig 7 displays the result of the algorithm, Fig 9 displays the covariance matrix found by the algorithm and the true covariance matrix. Fig 7 displays the normalized variance of the weights used to estimate the integral required for the gradient of β. A low variance means that our importance law targets the right area. We want the variance as low as possible. If $(w_{i,k})_{1 \leqslant k \leqslant N_{\text{samples}}}$ are the weights used to estimate the i^{th} coordinate of the integral, then the normalized variance V_β is

$$V_\beta := \frac{1}{p} \sum_{i=1}^{p} \sigma_i \tag{16}$$

$$\sigma_i = \frac{1}{N_{\text{samples}}^2} \sum_{i=1}^{N_{\text{samples}}} (w_{i,k} - \bar{w_i})^2 \tag{17}$$

It almost finds the structure back as we see in Fig 9.The sample size grows with q, so that the algorithm does not work for $q > 3$ for reasonable sampling effort (about 25000). It fails to find the parameter or even the structure of the covariance matrix.

We took 25000 samples to estimate each integral. This is a lot, but necessary. Indeed, we show in 8 the result of the algorithm when taking less samples (100 samples). The MSE is greater, and the log likelihood is lower. This may explain why the algorithm does not work for larger q. The larger q, the spiker the density. We need 25000 samples to get a good estimate for $q = 3$, thus it stands to reason that we need more samples for $q > 3$. But getting more samples is very costly. An alternative would be to estimate only $p_\theta(Y)$ and derive the gradients directly from it, which we do in the following section.

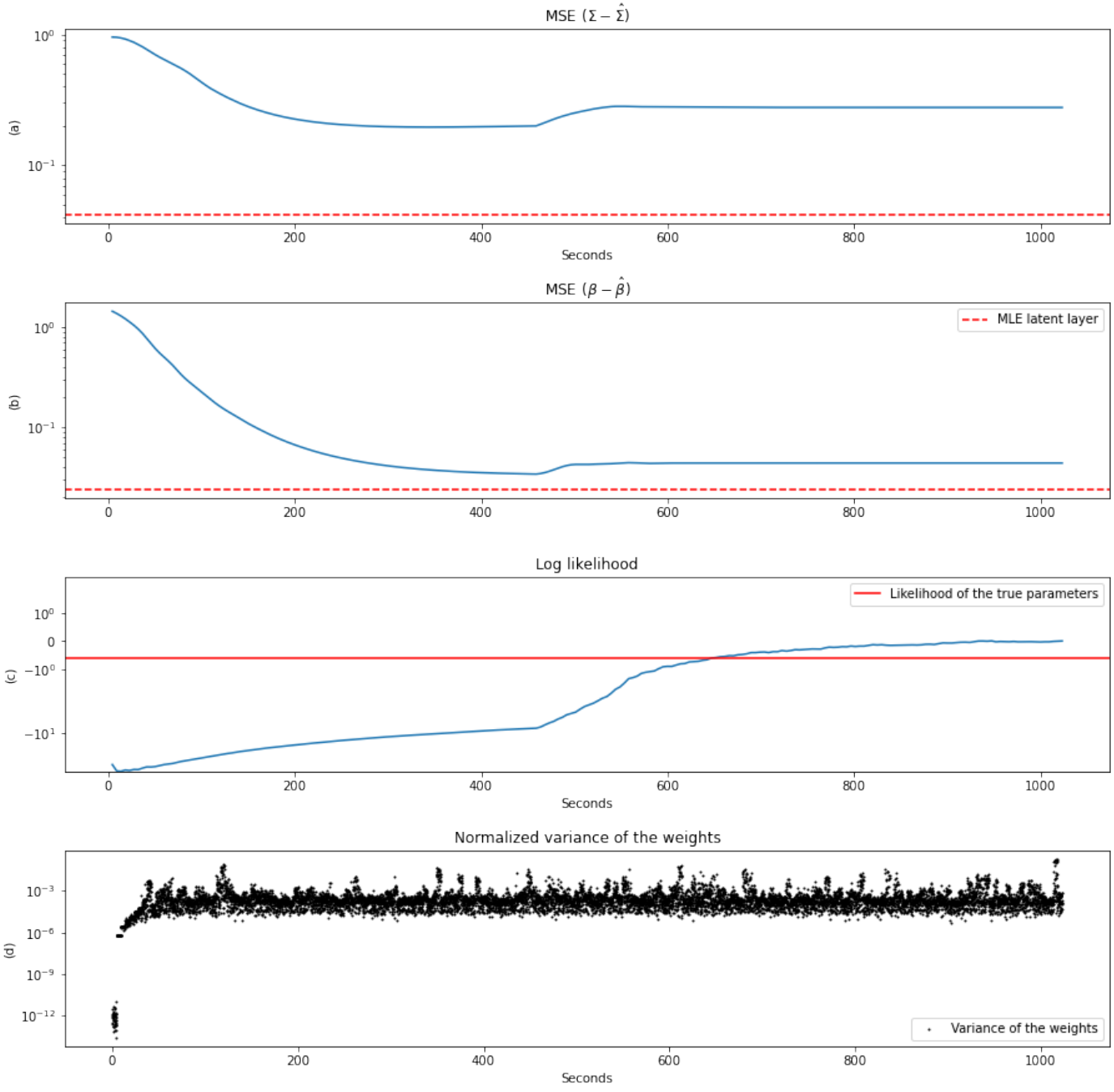

Figure 7: Simulation for $n = 100, p = 4, q = 3, d = 2$. Number of samples used for each integral : 25000. Estimation of $(p+1)(q+1)$ integrals. We remove the max on (c) that we computed the MSE and the log likelihood with the mean of the last 100 parameters for a clearer plot.

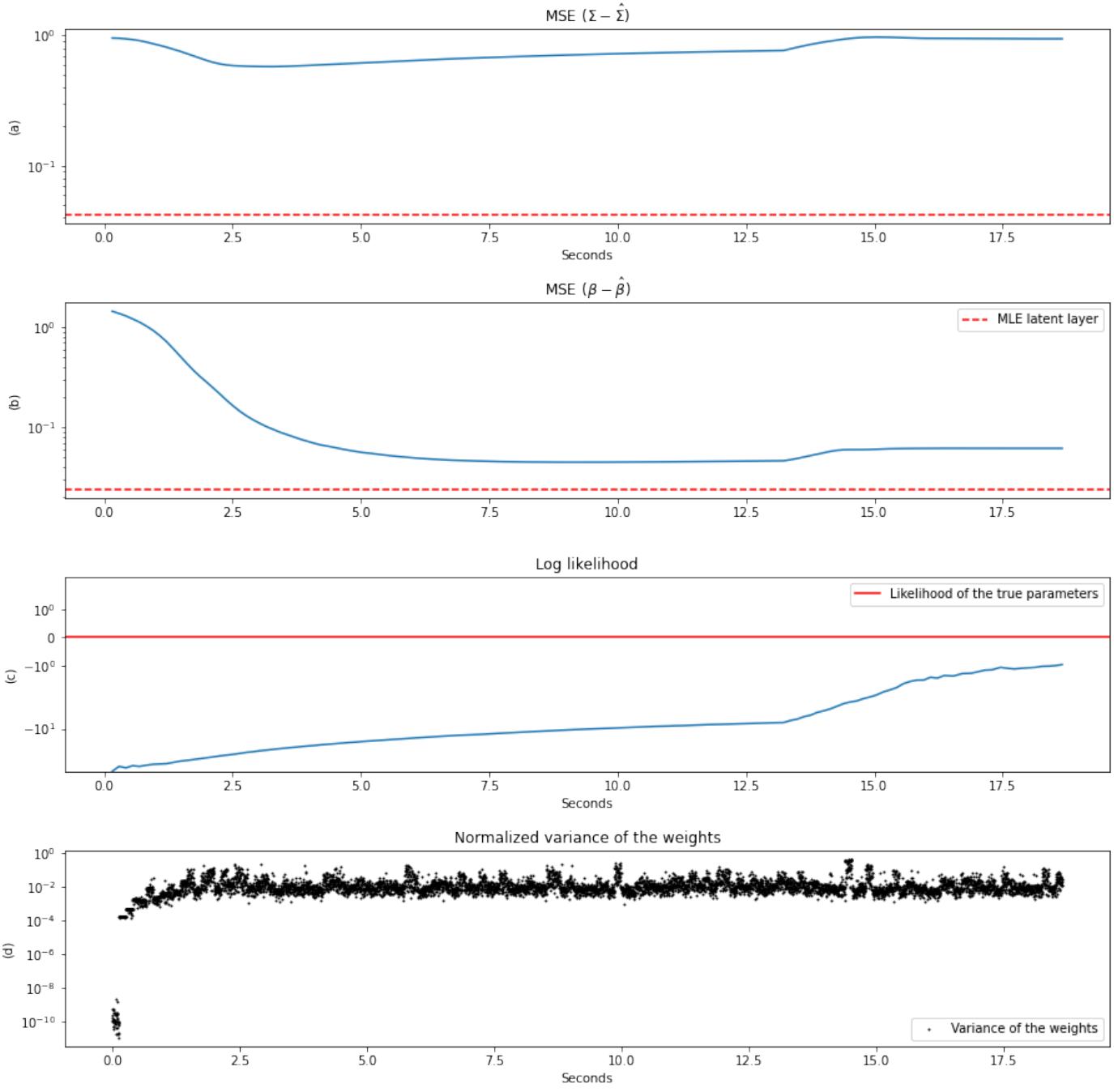

Figure 8: Simulation for $n = 100, p = 4, q = 3, d = 2$. Number of samples used for each integral : 100. Estimation of $(p + 1)(q + 1)$ integrals. We remove the max on (c) and computed the MSE and the log likelihood with the mean of the last 100 parameters for a clearer plot.

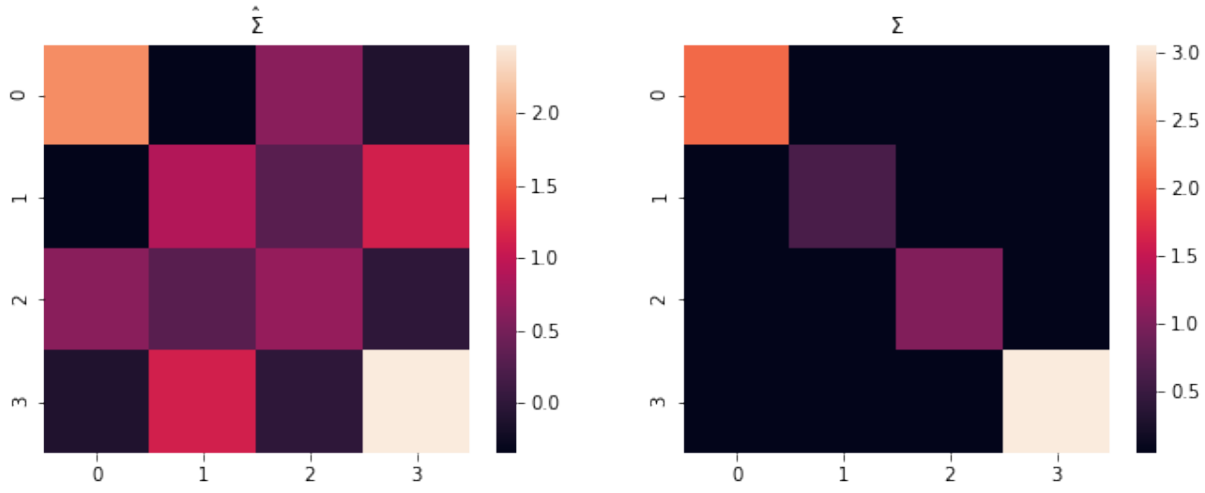

Figure 9: On the left, the covariance matrix estimated when the model is launched for $n = 1000, p = 30, q = 10$, the gradients were estimated with 13. On the right, the true parameter.

3.4 Approximating the likelihood with importance sampling

Estimating $(p+1)(q+1)$ integrals costs too much, and the integrals in 13 need too much samples to get a good approximation. An alternative is to estimate $p_\theta(Y_i)$ with importance sampling :

$$p_\theta(Y_i) = \int \tilde{p}_\theta^{(u)}(W)\mathrm{d}W \approx \frac{1}{n_s}\sum_{k=1}^{n_s} \frac{\tilde{p}_\theta^{(u)}(V_k)}{g(V_k)}, \quad (V_k)_{1\leqslant k\leqslant n_s} \overset{iid}{\sim} g$$

We then take the log of this approximation and derive the gradients. The approximation is the following :

$$\nabla_\theta \log p_\theta(Y_i) \approx \nabla_\theta \log\left(\frac{1}{n_s}\sum_{k=1}^{n_s} \frac{\tilde{p}_\theta^{(u)}(V_k)}{g(V_k)}\right) \tag{18}$$

3.4.1 Importance law choice

The function $\tilde{p}^{(u)}(.)$ is a product of two densities which results in an unnormalized density. However, here we only need to estimate the normalization factor. Thus we can (only) use importance sampling to estimate it. We need to find the right importance law. We will do as in 3.2.3, with $\phi(x) \equiv 1$. A major improvement is the zero variance estimator. Indeed, if we take the mean and covariance matrix as in 3.2.3, we are very close to the zero variance estimator. As a result, taking only 10 samples gives a very good estimation.

To sum up, this alternative has two benefits:

1. Estimation of only one integral.
2. Zero variance estimator reached.

However, it has one major disadvantage. We need to estimate $p_\theta(Y_i)$ which can be very low, and take the log after. we cannot estimate the logarithm directly due to the integral (we cannot swap logarithm and integral). This being said, $p_\theta(Y_i)$ becomes lower as p grows, and becomes numerical

zero for $p > 30$. We thus can't take a large p, but at least we can infer the MLE for low p, which was not possible with variational inference.

3.4.2 Application

We simulate data according to 1 and perform a gradient ascent, taking as optimiser Rprop. The result is displayed in Fig 10. We did not plot the MSE of Σ in the latent layer since it is a full rank matrix. We did not want to compare a full rank matrix with a rank q matrix. However, we plot the structure of the matrix in 11. It finds the structure back, even with a low rank estimation and β is well approximated, not far from the MLE in the latent layer. Note that the MSE of $\hat{\Sigma}$ is quite high but the structure is the right one. This may be explain by the low rank approximation. It is reasonable to think that we will have a better approximation of Σ when taking $p = q$. Note that the variance of the weights are very low (the logarithm has order -10^2), which indicates that we are reaching the zero variance estimator.

4 Conclusion and discussion

4.1 Conclusion

We managed to infer quickly the parameters for large n and large p (about 2000) with variational inference. However, for very large p (> 100000), there is a need to perform a dimension reduction as a matrix of size (p, p) would be too demanding, both in time calculus and memory. But variational inference does not adjust very well to dimension reduction as seen in 2.6. Moreover, it does not try to infer the MLE, which is not very convenient to get confidence intervals. We switched to a Monte-Carlo approach to fix those two problems. Unfortunately, we could only infer the MLE for low p (< 30).

4.2 Discussion

Variational inference does not adjust very well to dimension reduction only because it cannot compute the log of a determinant since the latter is null. Modifying this determinant to make it non-zero would be a good lead. For the importance sampling part, we cannot estimate $p_\theta(Y_i) = \int \tilde{p}_\theta(W)\mathrm{d}W$ when p becomes large. However, we may be able to estimate $C \times p_\theta(Y_i)$ for the right constant C. We need to tune this constant C in order to get a computable probability. Else, the importance law used to estimate the $(p+1)(q+1)$ integrals

$$\mathbb{E}_{\tilde{p}_\theta}\left[e^{CW_i}\right] \qquad \mathbb{E}_{\tilde{p}_\theta}\left[\exp\left(CW_i\right) W_i^\top\right] \qquad \mathbb{E}_{\tilde{p}_\theta}\left[W_i^\top\right] \tag{19}$$

does not seem to be very good since we need 25000 samples to get a good estimation. Finding a better one would be a good idea.

References

J. Aitchison and C. H. Ho. The multivariate Poisson-log normal distribution. *Biometrika*, 76(4): 643–653, 12 1989. ISSN 0006-3444. doi: 10.1093/biomet/76.4.643. URL https://doi.org/10.1093/biomet/76.4.643.

J. Chiquet, M. Mariadassou, and S. Robin. Variational inference for probabilistic poisson pca. 2018.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 39:1–38, 1977. URL http://web.mit.edu/6.435/www/Dempster77.pdf.

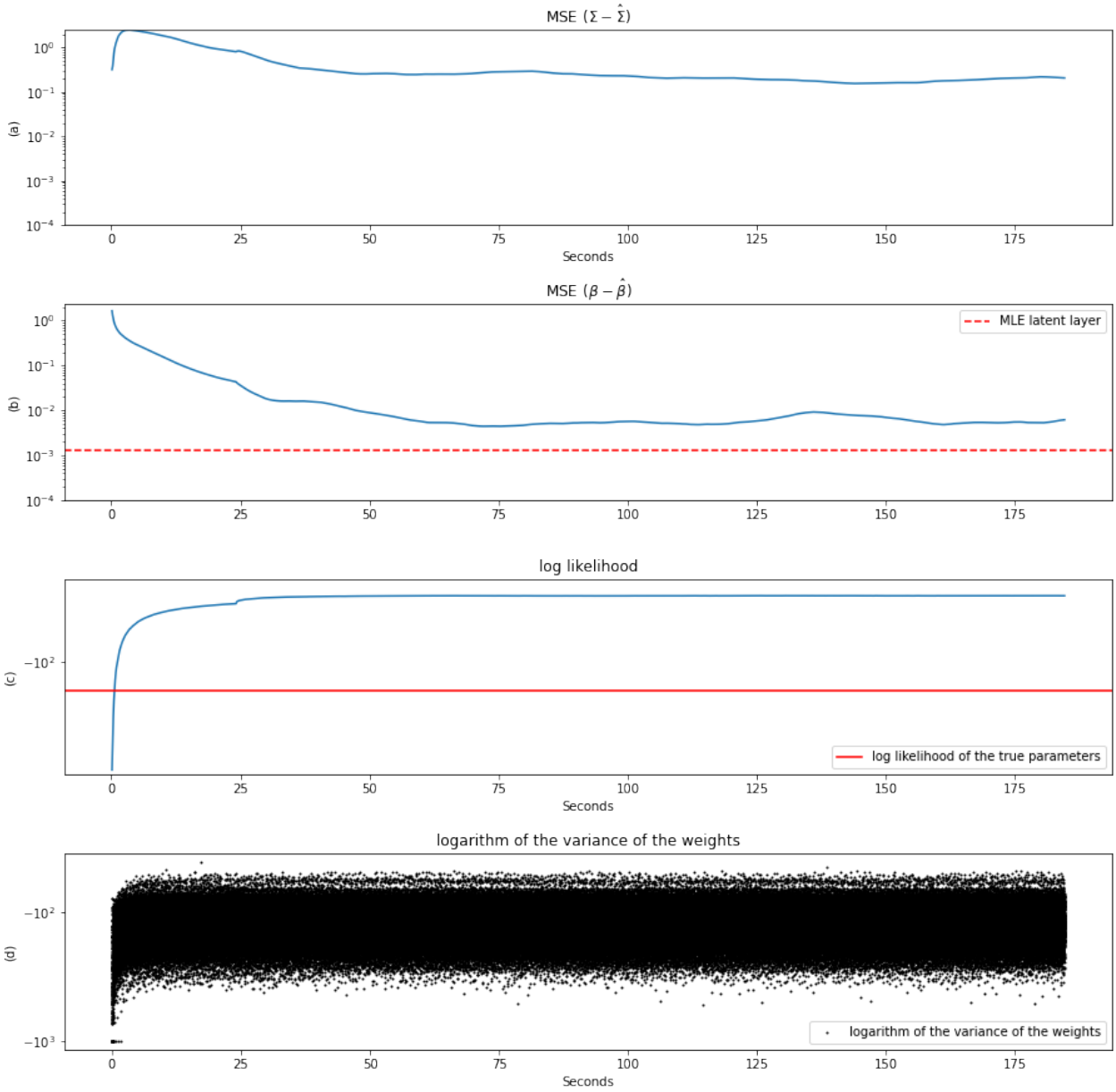

Figure 10: Simulation for $n = 1000,\ p = 30,\ q = 10,\ d = 2$. Number of samples used for each integral: 20. We estimated the gradients with 18. We computed the MSE and the log likelihood with the mean of the last 100 parameters for a clearer plot.

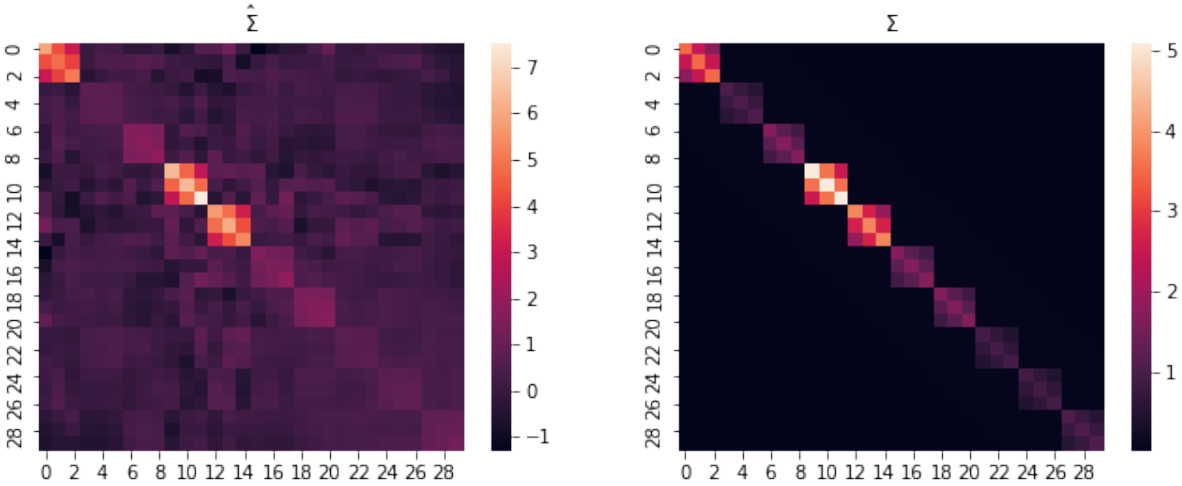

Figure 11: On the left, the covariance matrix estimated when the model is launched for $n = 1000$, $p = 30$, $q = 10$ when estimating the gradients with 18. On the right, the true parameter.

P. Hall, J. Ormerod, and M. Wand. Theory of gaussian variational approximation for a poisson mixed model. *Centre for Statistical Survey Methodology Working Paper Series*, 21, 01 2011.

R. Izsák. Maximum likelihood fitting of the poisson lognormal distribution. *Environ. Ecol. Stat.*, 15:143–156, 01 2008. doi: 10.1007/s10651-007-0044-x.

M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 01 1999. doi: 10.1023/A:1007665907178.

D. Karlis. Em algorithm for mixed poisson and other discrete distributions. *ASTIN Bulletin*, 35: 3–24, 05 2005. doi: 10.1017/S0515036100014033.

D. Lee and H. Seung. Algorithms for non-negative matrix factorization. *Adv. Neural Inform. Process. Syst.*, 13, 02 2001.

J. Li and D. Tao. Simple exponential family pca. *Journal of Machine Learning Research - Proceedings Track*, 9:453–460, 01 2010. doi: 10.1109/TNNLS.2012.2234134.

N. B. N. L. Johnson, S. Kotz. *Discrete multivariate distributions, volume 165*. 08 1997. ISBN 9780471667193. doi: 10.1002/0471667196.ess5099.pub2.

J. F. Nelson. Multivariate gamma-poisson models. *Journal of the American Statistical Association*, 80(392):828–834, 1985. doi: 10.1080/01621459.1985.10478190.

A. B. Owen. *Monte Carlo theory, methods and examples.* 2013.

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

M. A. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: the rprop algorithm. *IEEE International Conference on Neural Networks*, pages 586–591 vol.1, 1993.

J. a. Royle and C. Wikle. Efficient statistical mapping of avian count data. *Environmental and Ecological Statistics*, 12:225–243, 06 2005. doi: 10.1007/s10651-005-1043-4.

H. Rue, S. Martino, and N. Chopin. Approximate bayesian inference for latent gaussian models by using integrated nested laplace approximations. *Journal of the Royal Statistical Society Series B*, 71:319–392, 04 2009. doi: 10.1111/j.1467-9868.2008.00700.x.

A. Silva, S. Rothstein, P. McNicholas, and S. Dang. A multivariate poisson-log normal mixture model for clustering transcriptome sequencing data. *BMC Bioinformatics*, 20, 07 2019. doi: 10.1186/s12859-019-2916-0.

S. Srivastava and L. Chen. A two-parameter generalized poisson model to improve the analysis of rna-seq data. *Nucleic acids research*, 38:e170, 09 2010. doi: 10.1093/nar/gkq670.

P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

M. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1:1–305, 01 2008. doi: 10.1561/2200000001.

K. Wang, T. Bhowmik, S. Zhao, N. Eluru, and E. Jackson. Highway safety assessment and improvement through crash prediction by injury severity and vehicle damage using multivariate poisson-lognormal model and joint negative binomial-generalized ordered probit fractional split model. *Journal of Safety Research*, 76, 12 2020. doi: 10.1016/j.jsr.2020.11.005.

M. S. Williams and E. D. Ebel. Methods for fitting the poisson-lognormal distribution to microbial testing data. *Food Control*, 27(1):73–80, 2012. ISSN 0956-7135. doi: https://doi.org/10.1016/j.foodcont.2012.03.007. URL https://www.sciencedirect.com/science/article/pii/S0956713512001260.