Notes: MPLN model

1 Multivariate Poisson Lognormal (MPLN) model

The MPLN model (**?**) relates some p-dimensional observation vectors $\boldsymbol{Y}_i$ to some p-dimensional vectors of Gaussian latent variables Z_i as follows

$$\begin{aligned}&\boldsymbol{Y}_i \mid \boldsymbol{Z}_i \sim Poisson(e^{\boldsymbol{Z}_i}) \quad , \quad Y_{ij} \mid Z_{ij} \; indep.\\ &\boldsymbol{Z}_i \sim \mathcal{N}(\boldsymbol{X}_i^\intercal \boldsymbol{B}_{d\times p}, \Sigma_{p\times p})\end{aligned}$$

where

- Y_{ij} represents the jth code of ith patient with $i = 1, \cdots, n$ and $j = 1, \cdots, p$, $\boldsymbol{Y} = (\boldsymbol{Y}_1, \cdots, \boldsymbol{Y}_n)^\intercal \in \mathbb{R}^{n\times p}$,
- $\boldsymbol{Z}_i$ is the patient-level embedding of ith patient, $\boldsymbol{Z} = (\boldsymbol{Z}_1, \cdots, \boldsymbol{Z}_n)^\intercal \in \mathbb{R}^{n\times p}$,
- $\boldsymbol{X}_i$ is the covariates of ith patient, $\boldsymbol{X} = (\boldsymbol{X}_1, \cdots, \boldsymbol{X}_n)^\intercal \in \mathbb{R}^{n\times d}$

2 Leveraging external information for Σ

In the MPLN model, $\boldsymbol{\Sigma}$ describes the underlying residual relationships between the p codes. We may decompose $\boldsymbol{\Sigma} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^\intercal$, where $\boldsymbol{V} \in \mathbb{R}^{p\times q}$ is orthonormal and the rank $q \leq p$, $\boldsymbol{\Lambda} \in \mathbb{R}^{q\times q}$ is diagonal. Assuming that across different institution/dataset, the orientation/rotation ($\boldsymbol{V}$) is the same but the shape/scaling $\boldsymbol{\Lambda}$ can be different, we may borrow $\boldsymbol{V}$ that is obtained from external sources for model estimation. In that case, instead of estimating the full covariance matrix $\boldsymbol{\Sigma}$, we only need to estimate the diagnol matrix $\boldsymbol{\Lambda}$.

An alternative way to write the model:

$$\begin{aligned}&\boldsymbol{Y}_i \mid \boldsymbol{Z}_i \sim Poisson(e^{\boldsymbol{Z}_i}) \quad , \quad Y_{ij} \mid Z_{ij} \; indep.\\ &\boldsymbol{Z}_i = \boldsymbol{X}_i^\intercal \boldsymbol{B} + \boldsymbol{V}\boldsymbol{\Lambda}^{1/2}\boldsymbol{W}_i \quad ,\\ &\boldsymbol{W}_i \sim \mathcal{N}(0_q, \boldsymbol{I}_{q\times q}) \quad iid\end{aligned}$$

2.1 Likelihood

In matrix notation, $\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{B} + \boldsymbol{W}\boldsymbol{\Lambda}^{1/2}\boldsymbol{V}^\intercal$. The observation matrix $\boldsymbol{Y}$ only depends on $\boldsymbol{Z}$ through $\boldsymbol{B}$, $\boldsymbol{\Lambda}$ and $\boldsymbol{W}$, and the complete log-likelihood is therefore

$$\begin{aligned}\log p(\boldsymbol{Y}, \boldsymbol{W}; \boldsymbol{B}, \boldsymbol{\Lambda}) &= \sum_{i=1}^{n} \log p(\boldsymbol{Y}_i \mid \boldsymbol{W}_i; \boldsymbol{B}, \boldsymbol{\Lambda}) + \log p(\boldsymbol{W}_i)\\ &= \mathbf{1}_n^\intercal \left\{ \boldsymbol{Y} \odot (\boldsymbol{X}\boldsymbol{B} + \boldsymbol{W}\boldsymbol{\Lambda}^{1/2}\boldsymbol{V}^\intercal) - \exp(\boldsymbol{X}\boldsymbol{B} + \boldsymbol{W}\boldsymbol{\Lambda}^{1/2}\boldsymbol{V}^\intercal) \right\} \mathbf{1}_p\\ &\quad - \frac{||\boldsymbol{W}||_F^2}{2} - \frac{nq}{2}\log(2\pi) - \mathbf{1}_n^\intercal \log(\boldsymbol{Y}!)\mathbf{1}_p\end{aligned}$$

2.2 Variational bound of the likelihood

Maximize the evidence lower bound:

$$\begin{aligned}\mathcal{J}(\boldsymbol{B}, \boldsymbol{\Lambda}, q) &= \log p(\boldsymbol{Y}; \boldsymbol{B}, \boldsymbol{\Lambda}) - KL[q(\boldsymbol{W}), p(\boldsymbol{W} \mid \boldsymbol{Y}; \boldsymbol{B}, \boldsymbol{\Lambda}))] \\ &= E_q[\log p(\boldsymbol{Y}, \boldsymbol{W}; \boldsymbol{B}, \boldsymbol{\Lambda})] - E_q[\log q(\boldsymbol{W})]\end{aligned}$$

Choose the set $\mathcal{Q}$ of product distribution of q-dimensional multivariate Gaussian with diagonal covariance matrices:

$$\mathcal{Q} = \left\{ q : q(\boldsymbol{Z}) = \prod_i q_i(\boldsymbol{Z}_i), q_i(\boldsymbol{Z}_i) = \mathcal{N}(\boldsymbol{Z}_i; \boldsymbol{m}_i, diag(\boldsymbol{s}_i \circ \boldsymbol{s}_i))), (\boldsymbol{m}_i, \boldsymbol{s}_i) \in \mathbb{R}^q \times \mathbb{R}_+^q \right\}$$

Let $\boldsymbol{M} = (\boldsymbol{m}_1, \cdots, \boldsymbol{m}_n)^\intercal \in \mathbb{R}^{n \times q}$ and $\boldsymbol{S} = (\boldsymbol{s}_1, \cdots, \boldsymbol{s}_n)^\intercal \in \mathbb{R}^{n \times q}$. Results on first- and second-order moments of multivariate Gaussian show that

$$\begin{aligned}\mathcal{J}(\boldsymbol{B}, \boldsymbol{\Lambda}, \boldsymbol{M}, \boldsymbol{S}) = \mathbf{1}_n^\intercal \left[\boldsymbol{Y} \odot (\boldsymbol{X}\boldsymbol{B} + \boldsymbol{M}\boldsymbol{\Lambda}^{1/2}\boldsymbol{V}^\intercal) - \mathbb{E}_q \left\{ \exp(\boldsymbol{X}\boldsymbol{B} + \boldsymbol{W}\boldsymbol{\Lambda}^{1/2}\boldsymbol{V}^\intercal) \right\} \right] \mathbf{1}_p \\ - \frac{1}{2} \mathbf{1}_n^\intercal \left\{ \boldsymbol{M} \odot \boldsymbol{M} + \boldsymbol{S} \odot \boldsymbol{S} - 2\log(\boldsymbol{S}) - \mathbf{1}_{n \times q} \right\} \mathbf{1}_q - \mathbf{1}_n^\intercal \log(\boldsymbol{Y}!) \mathbf{1}_p\end{aligned}$$

where

$$\mathbb{E}_q \left\{ \exp(\boldsymbol{X}\boldsymbol{B} + \boldsymbol{W}\boldsymbol{\Lambda}^{1/2}\boldsymbol{V}^\intercal) \right\} = \exp \left\{ \boldsymbol{X}\boldsymbol{B} + \boldsymbol{M}\boldsymbol{\Lambda}^{1/2}\boldsymbol{V}^\intercal + \frac{1}{2}(\boldsymbol{S} \odot \boldsymbol{S})(\boldsymbol{\Lambda}^{1/2}\boldsymbol{V}^\intercal \odot \boldsymbol{\Lambda}^{1/2}\boldsymbol{V}^\intercal) \right\} =: \boldsymbol{A}$$

2.3 Blockwise gradient

Find blockwise gradients of $\mathcal{J}(\boldsymbol{B}, \boldsymbol{\Lambda}, \boldsymbol{M}, \boldsymbol{S})$:

$$\begin{aligned}\frac{\partial \mathcal{J}}{\partial \boldsymbol{B}} &= (\boldsymbol{Y} - \boldsymbol{A})^\intercal \boldsymbol{X} \\ \frac{\partial \mathcal{J}}{\partial \boldsymbol{\Lambda}^{1/2}} &= \boldsymbol{V}^\intercal \left[(\boldsymbol{Y} - \boldsymbol{A})^\intercal \boldsymbol{M} - \{\boldsymbol{A}^\intercal (\boldsymbol{S} \odot \boldsymbol{S})\} \odot \boldsymbol{V}\boldsymbol{\Lambda}^{1/2} \right] \in \mathbb{R}^{q \times q} \\ \frac{\partial \mathcal{J}}{\partial \boldsymbol{M}} &= (\boldsymbol{Y} - \boldsymbol{A})\boldsymbol{V}\boldsymbol{\Lambda}^{1/2} - \boldsymbol{M} \\ \frac{\partial \mathcal{J}}{\partial \boldsymbol{S}} &= \boldsymbol{S}^{\oslash} - \boldsymbol{S} - 2\boldsymbol{A}(\boldsymbol{V}\boldsymbol{\Lambda}^{1/2} \odot \boldsymbol{V}\boldsymbol{\Lambda}^{1/2}) \odot \boldsymbol{S}\end{aligned} \tag{1}$$

where $\boldsymbol{S}^{\oslash}$ is the element-wise inverse of $\boldsymbol{S}$, $S_{ij}^{\oslash} = S_{ij}^{-1}$. Unlike the standard MPLN, there is no closed form solution for $\boldsymbol{B}$ and $\boldsymbol{\Lambda}^{1/2}$ with $\boldsymbol{M}$ and $\boldsymbol{S}$ fixed. However, since the blockwise gradients can be easily derived, we may still use gradient-based local optimization algorithms for maximizing the ELBO.

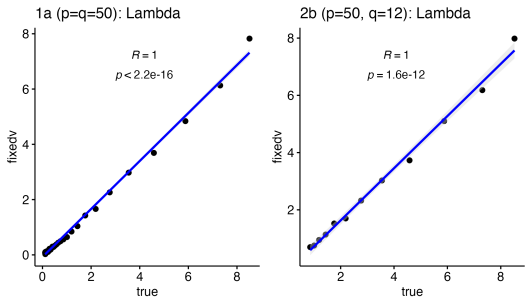
In $\boldsymbol{\Lambda}^{1/2}$, we only have q parameters instead of $q \times q$.

Current implementation: optimization using `NLOPT` library in `C++`, similar to `?`.

2.4 Simulation (Test code...)

Let $n = 5000$, $p = 50$. Consider two scenarios: true covariance has full rank and has low rank.

- **Scenario 1 (true covariance with full rank)**:
  1. Generate $\mathbf{\Sigma} = AR_1(0.8)$. Find its eigen-decomposition $\mathbf{\Sigma} = \boldsymbol{V}\mathbf{\Lambda}\boldsymbol{V}^\intercal$.
  2. Generate $\boldsymbol{Z} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ and $\boldsymbol{Y} \sim Poisson(e^{\boldsymbol{Z}})$.
  3. Fit model with $\boldsymbol{V}$,
     a. (correct) assuming $\mathbf{\Sigma}$ has full rank, $\mathbf{\Lambda} \in \mathbb{R}^{p\times p}$, we use the full $\boldsymbol{V}$ here.
     b. (low rank appriximation) assuming $\mathbf{\Sigma}$ has lower rank, $\mathbf{\Lambda} \in \mathbb{R}^{q\times q}$ with $q = 20, 10, 5$, we only use $\boldsymbol{V}_{[,1:q]}$ here.
- **Scenario 2 (true covariance with low rank, $q = 12$):**
  1. Generate $\mathbf{\Sigma} = AR_1(0.8)$. Find its eigen-decomposition $\mathbf{\Sigma} = \boldsymbol{V}\mathbf{\Lambda}\boldsymbol{V}^\intercal$.
  2. Let $\widetilde{\mathbf{\Sigma}} = \boldsymbol{V}_{[,1:q]}\mathbf{\Lambda}_{[1:q,1:q]}\boldsymbol{V}^T_{[,1:q]}$. Find its eigen-decomposition $\widetilde{\mathbf{\Sigma}} = \widetilde{\boldsymbol{V}}\widetilde{\mathbf{\Lambda}}\widetilde{\boldsymbol{V}}^\intercal$. This will be our true covariance with known $rank = q$. (In reality we observe $\widetilde{\mathbf{\Sigma}}$ from external sources)
  3. Generate $\boldsymbol{Z} \sim \mathcal{N}(\mathbf{0}, \widetilde{\mathbf{\Sigma}})$ and $\boldsymbol{Y} \sim Poisson(e^{\boldsymbol{Z}})$.
  4. Fit model with $\widetilde{\boldsymbol{V}}$,
     a. (neglecting low-rankness) assuming $\mathbf{\Sigma}$ has full rank, $\mathbf{\Lambda} \in \mathbb{R}^{p\times p}$, we use the full $\widetilde{\boldsymbol{V}}$ here.
     b. (correct) assuming $\mathbf{\Sigma}$ has lower rank, $\mathbf{\Lambda} \in \mathbb{R}^{q\times q}$ with $q = 20, 10, 5$, we only use $\widetilde{\boldsymbol{V}}_{[,1:q]}$ here. (In reality we don't know the true rank, so would also need to implement a procedure to select for the best q.
- Compare $\widehat{\mathbf{\Lambda}}$ with the $\mathbf{\Lambda}$ (for scenario 1) or $\widetilde{\mathbf{\Lambda}}$ (for scenario 2). Here I'm only showing 1a and 2b (with correct $q = 12$), since the dimension of $\widehat{\mathbf{\Lambda}}$ doesn't match with the truth in 1b and 2a.



- Compare model-generated embeddings v.s. the true $\boldsymbol{Z}$. *fixedV* is our implementation and *full* is the original implementation (estimating the full $\mathbf{\Sigma}$ based on data). Results on next page.

  Since *fixedV* has much less parameters even if $p = q$, the variance should be smaller compared with *full*?
- Look at the estimated mean and covariance: problem: estimated mean is biased? looking into this

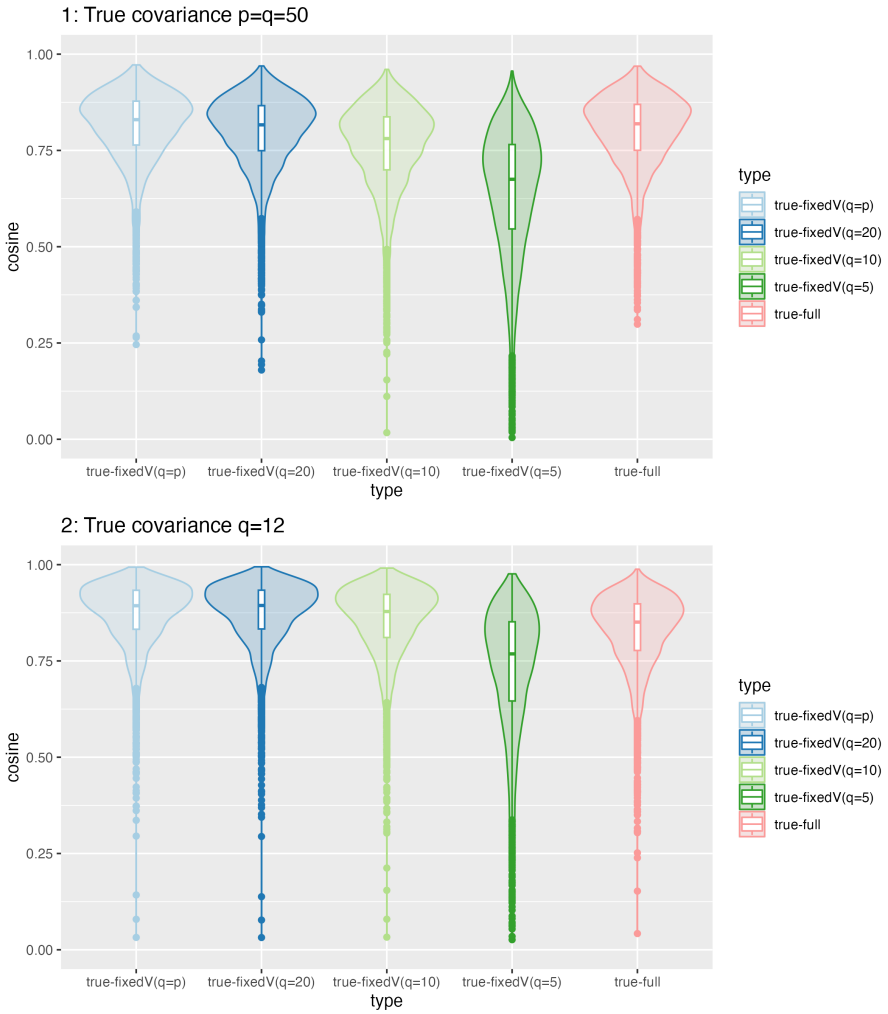


Figure 1: Cosine similarities between the latent variable $\boldsymbol{Z}$ obtained with free $\boldsymbol{\Sigma}$ and fixed eigenvectors $\boldsymbol{V}$.

Notes 4/5/2023

So far...

- Issue found in simulation: Parameter estimates are biased with `PLNModel` package even with small p and very large n.
- What causes the bias?
  - Based on **?**, Variational Gaussian Approximation (VGA) for our model (iid and full rank) is inconsistent.
- Potential remedies?
  - *(known $\mathbf{\Sigma}$, iid, full rank)* If we know the covariance parameters, try to re-parameterize the model and leverage independence in the VGA alg
  - *(unknown $\mathbf{\Sigma}$, iid, low rank)* If we use the current VGA alg, need to allow p to grow with n while constraining rank q to be relatively small, so that parameter estimation is consistent in iid setting.
  - *(unknown $\mathbf{\Sigma}$, iid, full rank)* Explore other methods that might help reduce the bias, e.g. black box variatoinal inference (BBVI) (**?**).

Why biased?

Theory of Gaussian Variational Approximation for a Poisson Mixed Model. **?**

- They considered repeated measures data:
  - Data: (X_{ij}, Y_{ij}), $1 \leq i \leq m$ corresponds to m patients, $1 \leq j \leq n$ corresponds to n repeated measures on those patients.
  - Model: $Y_{ij} \mid (X_{ij}, U_i)$ indep Poisson with mean $\exp(\beta_0 + \beta_1 X_{ij} + U_i)$, U_i independent $N(0, \sigma^2)$
  - Results: Estimators of the model parameters based on Gaussian variational approximation is consistent at rate $m^{-1/2} + n^{-1}$.
    - Number of model parameters: $d + 1$
    - Number of variational parameters: $2m$
    - Essentially, need to use the n repeated measures from each patient i to estimate the variational parameters m_i and s_i, so n cannot be too small.
- In our notations, their model is equivalent to

$$\begin{aligned} &\boldsymbol{Y}_i \mid \boldsymbol{Z}_i \sim Poisson(e^{\boldsymbol{Z}_i}) \quad , \quad Y_{ij} \mid Z_{ij} \text{ indep.} \\ &\boldsymbol{Z}_i = \boldsymbol{X}_i^\top \boldsymbol{B} + \boldsymbol{\epsilon}_i \quad , \quad \boldsymbol{\epsilon}_i = \epsilon_i \mathbf{1}_p \quad , \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2) \quad iid \end{aligned}$$

  v.s. our model

$$\begin{aligned} &\boldsymbol{Y}_i \mid \boldsymbol{Z}_i \sim Poisson(e^{\boldsymbol{Z}_i}) \quad , \quad Y_{ij} \mid Z_{ij} \text{ indep.} \\ &\boldsymbol{Z}_i = \boldsymbol{X}_i^\top \boldsymbol{B} + \boldsymbol{\epsilon}_i \quad , \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}_p, \mathbf{\Sigma}_{p \times p}) \quad iid \end{aligned}$$

  - Number of model parameters: $dp + p(p+1)/2$
  - Number of variational parameters: $2np \rightarrow$ too large, won't be able to estimate properly

- Their model is similar (though not exactly the same) to our model if we assume $\mathbf{\Sigma}$ has rank $q = 1$. In that case number of variational parameters can be reduce to $2nq = 2n$. Run a simulation to check: fix $rank = 1$, generate data with varying p and n, check bias. $\rightarrow$ verified that with large enough n, bias goes to 0 as p grows.

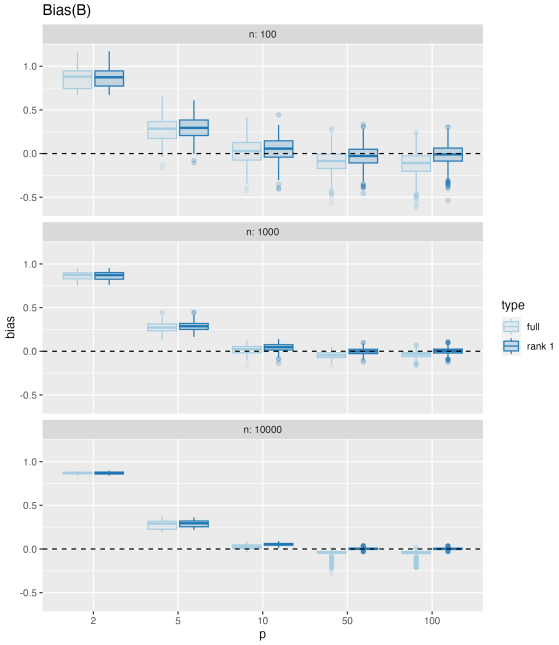


Figure 2: *full:* VGA with $2np$ variational parameters; *rank 1:* VGA with $2n$ variational parameters. Here I'm only showing bias of B but the bias of $\mathbf{\Sigma}$ has a similiar pattern.

Potential remedies?

1. With Known $\mathbf{\Sigma}$, modify VGA

Our model:

$$\begin{aligned} &\boldsymbol{Y}_i \mid \boldsymbol{Z}_i \sim Poisson(e^{\boldsymbol{Z}_i}) \quad , \quad Y_{ij} \mid Z_{ij} \ indep. \\ &\boldsymbol{Z}_i = \boldsymbol{X}_i^\intercal \boldsymbol{B} + \boldsymbol{V}\boldsymbol{\Lambda}^{1/2}\boldsymbol{W}_i \quad , \\ &\boldsymbol{W}_i \sim \mathcal{N}(0_q, \boldsymbol{I}_{q\times q}) \quad iid \end{aligned}$$

- Re-parameterize: $\boldsymbol{W}_i = \boldsymbol{\Lambda}^{-1/2}\boldsymbol{V}^\intercal(\boldsymbol{Z}_i - \boldsymbol{X}_i^\intercal \boldsymbol{B}) \sim \mathcal{N}(0_q, \boldsymbol{I}_{q\times q}) \quad iid$
- Question: can we leverage the iid nature of $\boldsymbol{W}_i$ to use less variational parameters to estimate $f(\boldsymbol{W}_i \mid \boldsymbol{Y}_i; \boldsymbol{\Lambda}, \boldsymbol{V})$? For now, consider $q = p$, $\mathbf{\Sigma} = diag(\sigma_1^2, \cdots, \sigma_p^2)$ known, $\mu_1, \cdots, \mu_p$ unknown,
  - For each $j = 1, \cdots, p$, let p_θ be pdf of $poisson(\theta)$ and ϕ be pdf of $Normal(0, 1)$,

$$\begin{aligned} f(W_{ij} \mid \boldsymbol{Y}_i; \boldsymbol{\Lambda}, \boldsymbol{V}) &= \frac{\left\{ f(Y_{ij} \mid W_{ij}) \prod_{j' \neq j} f(Y_{ij'}) \right\} f(W_{ij})}{f(\boldsymbol{Y}_i)} \\ &= \frac{f(Y_{ij} \mid W_{ij}) f(W_{ij})}{f(Y_{ij})} \\ &= \frac{p_{\exp(\mu_j + \sigma_j^2 W_{ij})}(Y_{ij}) \phi(W_{ij})}{\int_{\mathbb{R}} p_{\exp(\mu_j + \sigma_j^2 w)}(Y_{ij}) \phi(w) \ dw} \end{aligned}$$

– In this case, the integral can be computed numerically, e,g, through quadrature or sampling, so no VGA is needed. If so, our estimators should be unbiased, and knowing σ_j is expected to help reduce the variance in our estimation.

– However in general, VGA is still needed for approximation of $p(\boldsymbol{W}_i \mid \boldsymbol{Y}_i; \boldsymbol{\Lambda}, \boldsymbol{V})$. In that case:

  * For $j \neq k$, $f(W_{ij} \mid \boldsymbol{Y}_i; \boldsymbol{\Lambda}, \boldsymbol{V}) \neq f(W_{ik} \mid \boldsymbol{Y}_i; \boldsymbol{\Lambda}, \boldsymbol{V})$ if $\mu_j \neq \mu_k$ or $\sigma_j \neq \sigma_j$, so we still need $2p$ parameters (p for mean and p for variance) for each $i = 1, \cdots, n$.
  * Knowing σ_j will not help reduce the bias of the estimation since the number of variational parameters to be estimated stays the same [pending:]. It might still help reduce the variance.

2. VGA with growing p in low rank

Based on the intuition from **?**, VGA for our model might be consistent if we allow p to grow along with n while controlling q to be small relative to p.

- Run simulation to check intuition: For fixed $n = 1000$, $p = 50$ or 100, increase q from 1 to p.
- *full* is using VGA with $2np$ variational parameters, and *true rank* is using VGA with $2nq$ variational parameters.

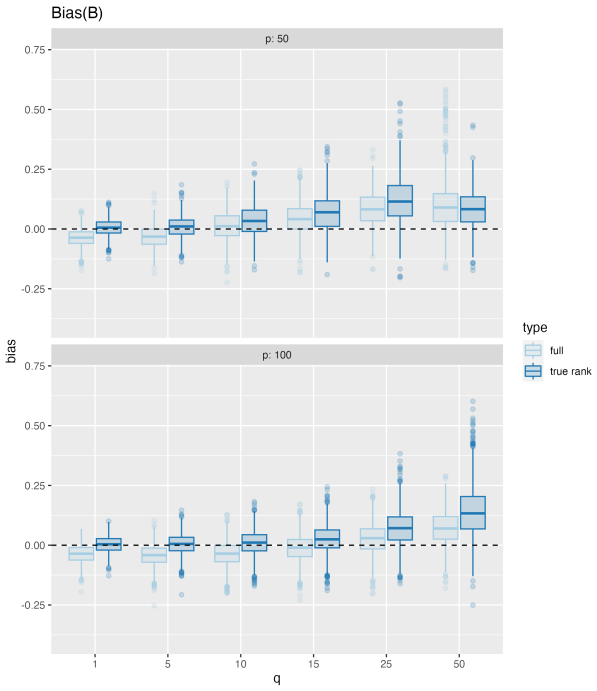


Figure 3: *full:* VGA with $2np$ variational parameters; *true rank:* VGA with $2nq$ variational parameters. Here I'm only showing bias of B but the bias of $\boldsymbol{\Sigma}$ has a similiar pattern.

- As expected, bias for *true rank* with $2nq$ variational parameters is small with small q, but increases as q grows. We do not know how to control the bias for *full* with $2np$ parameters.
- No theoretical results on consistency rate currently exists under this setting.

3. Other methods

BBVI: *Black Box Variational Inference*, **?**

- Algorithm:

Algorithm 2 Black Box Variational Inference (II)

Input: data x, joint distribution p, mean field variational family q.
Initialize $\lambda_{1:n}$ randomly, $t = 1$.
repeat
  // **Draw** S **samples from the variational approximation**
  for $s = 1$ **to** S **do**
    $z[s] \sim q$
  end for
  for $i = 1$ **to** n **do**
    for $s = 1$ **to** S **do**
      $f_i[s] = \nabla_{\lambda_i} \log q_i(z[s]|\lambda_i)(\log p_i(x, z[s]) - \log q_i(z[s]|\lambda_i))$
      $h_i[s] = \nabla_{\lambda_i} \log q_i(z[s]|\lambda_i)$
    end for
    $\hat{a}_i^* = \frac{\sum_{d=1}^{n_i} \widehat{\text{Cov}}(f_i^d, h_i^d)}{\sum_{d=1}^{n_i} \widehat{\text{Var}}(h_i^d)}$
    $\hat{\nabla}_{\lambda_i} \mathcal{L} \triangleq \frac{1}{S} \sum_{s=1}^{S} f_i[s] - \hat{a}_i^* h_i[s]$
  end for
  $\rho = t$tth value of a Robbins Monro sequence
  $\lambda = \lambda + \rho \hat{\nabla}_\lambda \mathcal{L}$
  $t = t + 1$
until change of λ is less than 0.01.

- Advantage:
  - Flexible variational family, not limited to Gaussian distribution, as long as
    - we know the score function of the variational distribution, and
    - we can sample from that distribution
  - It should be computationally fast.
- It is currently unclear whether it can help resolve the consistency issue:
  - No matter what distribution we choose, we still need to estimate $2np$ variational parameters.
  - The sampling scheme is used for estimating the gradients of the ELBO. Increasing the number of samples S might make this estimation better, which will likely make convergence faster. However, it does not necessarily help estimate the variational distribution better as essentially we are not gaining any information from this step. Therefore it might not be able to fix our consistency issue.
  - Currently no theoretical guarantee found.
- Still looking into this and trying to understand whether/how it can help.

MCMC-EM: A multivariate Poisson-log normal mixture model for clustering transcriptome sequencing data **?**

- Algorithm: sample from $f(\boldsymbol{W} \mid \boldsymbol{Y}; \boldsymbol{\Lambda}, \boldsymbol{V})$ using MCMC, then calculate expected values based on samples
  - Using `RStan` package

- Advantage:
  - If p not too large when the MCMC can work well, parameter estimates should have consistency rate of mle.
- Concerns:
  - Will not work well for large p. In the paper they only tested the alg on $n = 200, p = 6$. Highly doubt that it will work well in our setting where p can be in the hundreds.
  - In the paper they didn't show any result on the performance of parameter estimation.
  - Probably too slow if directly applied to our datasets with much larger n and p.
- Also looking into other methods based on sampling or approximation that might work...

4. Two-step smoothing

Idea: approximate $p(W \mid X)$ by $q_x(W) = \mathcal{N}(g(x), h(x))$, where $g(x)$ and $h(x)$ are modeled by NN.

4.1 Simulation

With $n = 10000, 50000$, $p = 50, 100, 200$, $q = 5, 10, 20$,

Data Generation

1. Generate $\boldsymbol{\Sigma} = AR_1(0.5)$. Find its eigen-decomposition $\boldsymbol{\Sigma} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^\top$.
2. Let $\widetilde{\boldsymbol{\Sigma}} = \boldsymbol{V}_{[,1:q]}\boldsymbol{\Lambda}_{[1:q,1:q]}\boldsymbol{V}^T_{[,1:q]}$.
3. Generate $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{0}, \widetilde{\boldsymbol{\Sigma}})$ and $\boldsymbol{Y} \sim Poisson(e^{\boldsymbol{Z}})$.

Parameter estimation

1. Find optimal $\boldsymbol{B}, \boldsymbol{\Sigma}, \boldsymbol{M}, \boldsymbol{S}$ that maximaize the ELBO (**??**) using blockwise gradient descent (**?**).
2. Fit $\boldsymbol{M} \sim \boldsymbol{X}$ and $\boldsymbol{S} \sim \boldsymbol{X}$ using two-layer FCNN to obtain $\widehat{\boldsymbol{M}}$, $\widehat{\boldsymbol{S}}$
   - Hidden layer: 30 nodes each, ReLU activation
   - Output layer: q nodes, linear activation for $\boldsymbol{M}$ and softplus activation for $\boldsymbol{S}$ (to ensure positiveness)
3. Plug in $\widehat{\boldsymbol{M}}$ and $\widehat{\boldsymbol{S}}$ and use blockwise gradient descent to find $\widehat{\boldsymbol{B}}$ and $\widehat{\boldsymbol{\Sigma}}$

Results

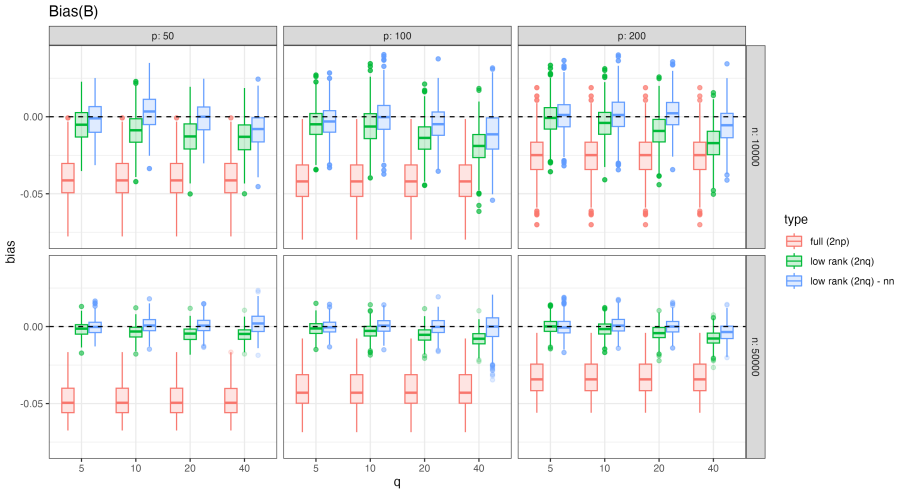


Figure 4: *full:* VGA with $2np$ variational parameters; *true rank:* VGA with $2nq$ variational parameters; *true rank:* with additional NN smoothing step. Here I'm only showing bias of B but the bias of $\mathbf{\Sigma}$ has a similiar pattern.

3 Notes 4/23/2023

Model:

$$\begin{aligned}
&\boldsymbol{X}_i \mid \boldsymbol{Z}_i \sim Poisson(e^{\boldsymbol{Z}_i}) \quad , \quad Y_{ij} \mid Z_{ij} \ indep. \\
&\boldsymbol{Z}_i = \boldsymbol{D}_i^\intercal \boldsymbol{B} + \boldsymbol{V}\boldsymbol{\Lambda}^{1/2}\boldsymbol{W}_i \quad , \\
&\boldsymbol{W}_i \sim \mathcal{N}(0_q, \boldsymbol{I}_{q\times q}) \quad iid
\end{aligned}$$

Idea: approximate $p(W \mid X)$ by $q_x(W) = \mathcal{N}(g(x), h(x))$, where $g(x)$ and $h(x)$ are modeled by NN.

3.1 Simulation

With $n = 10000, 50000$, $p = 50, 100, 200$, $q = 5, 10, 20$,

3.1.1 Data Generation

1. Generate $\boldsymbol{\Sigma} = AR_1(0.5)$. Find its eigen-decomposition $\boldsymbol{\Sigma} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^\intercal$.
2. Let $\widetilde{\boldsymbol{\Sigma}} = \boldsymbol{V}_{[,1:q]}\boldsymbol{\Lambda}_{[1:q,1:q]}\boldsymbol{V}_{[,1:q]}^T$.
3. Generate $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{0}, \widetilde{\boldsymbol{\Sigma}})$ and $\boldsymbol{Y} \sim Poisson(e^{\boldsymbol{Z}})$.

3.1.2 Parameter estimation

1. Find optimal $\boldsymbol{B}, \boldsymbol{\Sigma}, \boldsymbol{M}, \boldsymbol{S}$ that maximaize the ELBO (**??**) using blockwise gradient descent (**?**).
2. Fit $\boldsymbol{M} \sim \boldsymbol{X}$ and $\boldsymbol{S} \sim \boldsymbol{X}$ using two-layer FCNN to obtain $\widehat{\boldsymbol{M}}$, $\widehat{\boldsymbol{S}}$
   - Hidden layer: 30 nodes each, ReLU activation
   - Output layer: q nodes, linear activation for $\boldsymbol{M}$ and softplus activation for $\boldsymbol{S}$ (to ensure positiveness)
3. Plug in $\widehat{\boldsymbol{M}}$ and $\widehat{\boldsymbol{S}}$ and use blockwise gradient descent to find $\widehat{\boldsymbol{B}}$ and $\widehat{\boldsymbol{\Sigma}}$

3.1.3 Results

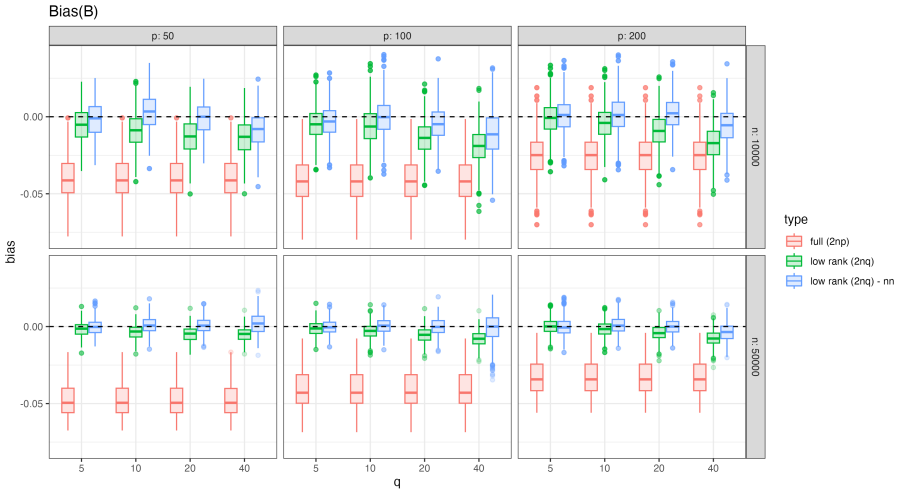
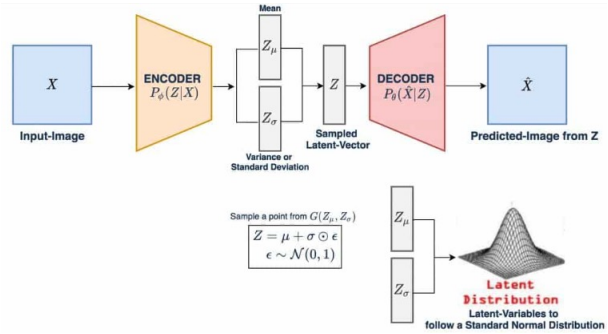


Figure 5: *full:* VGA with $2np$ variational parameters; *true rank:* VGA with $2nq$ variational parameters; *true rank:* with additional NN smoothing step. Here I'm only showing bias of B but the bias of $\mathbf{\Sigma}$ has a similiar pattern.

4 Notes on Variational Autoencoder (VAE)

4.1 VAE Framework



$$p(z) = \mathcal{N}(0, I)$$
$$p(x \mid z) = \mathcal{N}(f(z), cI) \quad , \quad f \in F$$

Approximate $p(z \mid x)$ by a Gaussian distribution

$$q_x(z) = \mathcal{N}(g(x), h(x)) \quad , \quad g \in G \quad h \in H.$$

We are looking for the optimal g^*, h^*, f^* such that

$$\begin{aligned}
(g^*, h^*) &= \underset{(g,h)\in G\times H}{\arg\min}\ KL(q_x(z), p(z \mid x)) \\
&= \underset{(g,h)\in G\times H}{\arg\max} \left\{ \mathbb{E}_{z\sim q_x}\left(-\frac{||x - f(x)||^2}{2c} \right) - KL(q_x(z), p(z)) \right\} \\
f^* &= \underset{f\in F}{\arg\max}\, \mathbb{E}_{z\sim q_x^*}(\log p(x \mid z)) \\
&= \underset{f\in F}{\arg\max}\, E_{z\sim q_x^*}\left(-\frac{||x - f(x)||^2}{2c} \right)
\end{aligned}$$

i.e.

$$(f^*, g^*, h^*) = \underset{(f,g,h)\in F\times G\times H}{\arg\max} \left\{ \underbrace{\mathbb{E}_{z\sim q_x}\left(-\frac{||x - f(x)||^2}{2c} \right)}_{\text{"reconstruction term"}} - \underbrace{KL(q_x(z), p(z))}_{\text{"approximation/regularization term"}} \right\}$$

- the constant c rules the balance between the two terms: with higher c we assume high variance around $f(z)$ for the decoder in the model so we favor the second term over the first term.
- the second term is often refered to as the "regularization term" because it forces the distributions returned by the encoder to be close to a standard normal distribution
  - We expect the latent space to be "regular"
    - Continuity: two close points in the latent space should not give two completely different contents once decoded

    - Completeness: for a chosen distribution, a point sampled from the latent space should give meaningful content once decoded
  - To achieve this we want to regularise both the covariance matrix and the mean of the distributions returned by the encoder
  - By forcing it to be standard normal, we require covariance to be close to identity, preventing distributions with tiny variances; and mean close to 0, preventing encoded distributions to be too far apart from each others.

4.2 Our Idea

Model:

$$\begin{aligned} p(w) &= \mathcal{N}(0, I) \\ z &= x^\intercal B + V\Lambda^{1/2}w \\ p(x \mid z) &= Poisson(e^z) \end{aligned}$$

Let $\theta = (B, \Sigma)$ denotes the model parameters. If we were to put the model under the VAE framework:
Approximate $p(w \mid x)$ by a Gaussian distribution

$$q_x(w) = \mathcal{N}(g(x), h(x)) \quad , \quad g \in G \quad h \in H.$$

We can find optimal (f^*, g^*, θ^*) by maximizing the ELBO, similar as VAE but with different forms since the model assumptions are different. The encoder is the same but for decoder we no longer need NN - simply update by gradient descent.