Poisson lognormal models for count data
Variational inference, Optimization

J. Chiquet, M. Mariadassou, S. Robin, B. Batardièrè + others

MIA Paris-Saclay, AgroParisTech, INRAE, Sorbonne University
Last update 16 November, 2022

https://pln-team.github.io/PLNmodels
Outline

1. **Framework** of multivariate Poisson lognormal models
2. **Optimization** with Variational inference
3. **Properties** of the Variational estimators
4. **A recent extension:** Zero-Inflated PLN
Multivariate Poisson lognormal models

Motivations, Framework
Generic form of data sets

Routinely gathered in ecology/microbiology/genomics

Data tables

- **Abundances**: read counts of species/transcripts $j$ in sample $i$
- **Covariates**: value of environmental variable $k$ in sample $i$
- **Offsets**: sampling effort for species/transcripts $j$ in sample $i$

Need frameworks to model *dependencies between counts*

- understand **environmental effects**
  $\leadsto$ explanatory models (multivariate regression, classification)
- exhibit **patterns of diversity**
  $\leadsto$ summarize the information (clustering, dimension reduction)
- understand **between-species interactions**
  $\leadsto$ 'network' inference (variable/covariance selection)
- correct for technical and **confounding effects**
  $\leadsto$ account for covariables and sampling effort
Models for multivariate count data

If we were in a Gaussian world...

The **general linear model** [MKB79] would be appropriate! For each sample \( i = 1, \ldots, n \),

\[
Y_i = \mathbf{x}_i^\top \mathbf{B} + \mathbf{o}_i + \mathbf{\varepsilon}_i, \quad \mathbf{\varepsilon}_i \sim \mathcal{N}(\mathbf{0}_p, \mathbf{\Sigma})
\]

null covariance \( \Leftrightarrow \) independence \( \sim \) uncorrelated species/transcripts do not interact

- **This model gives birth to** Principal Component Analysis, Discriminant Analysis, Gaussian Graphical Models, Gaussian Mixture models and many others . . .

With count data...

There is no generic model for multivariate counts

- Data transformation \((\log, \sqrt{\cdot})\): quick and dirty
- Non-Gaussian multivariate distributions [Ino+17]: do not scale to data dimension yet
- Latent variable models: interaction occur in a latent (unobserved) layer
The Poisson Lognormal model (PLN)

The PLN model [AH89] is a multivariate generalized linear model, where

- the counts $\mathbf{Y}_i$ are the response variables
- the main effect is due to a linear combination of the covariates $\mathbf{x}_i$
- a vector of offsets $\mathbf{o}_i$ can be specified for each sample.

\[
\mathbf{Y}_i | \mathbf{Z}_i \sim \mathcal{P}(\exp \mathbf{Z}_i), \quad \mathbf{Z}_i \sim \mathcal{N}(\mathbf{o}_i + \mathbf{x}_i^\top \mathbf{B}, \Sigma),
\]

The unknown parameters are

- $\mathbf{B}$, the regression parameters
- $\Sigma$, the variance-covariance matrix

Stacking all individuals together,

- $\mathbf{Y}$ is the $n \times p$ matrix of counts
- $\mathbf{X}$ is the $n \times d$ matrix of design
- $\mathbf{O}$ is the $n \times p$ matrix of offsets

Properties: over-dispersion, arbitrary-signed covariances

- mean: $\mathbb{E}(Y_{ij}) = \exp(o_{ij} + \mathbf{x}_i^\top \mathbf{B}.j + \sigma_{jj}/2) > 0$
- variance: $\nabla(Y_{ij}) = \mathbb{E}(Y_{ij}) + \mathbb{E}(Y_{ij})^2 \left(e^{\sigma_{jj}} - 1\right) > \mathbb{E}(Y_{ij})$
- covariance: $\text{Cov}(Y_{ij}, Y_{ik}) = \mathbb{E}(Y_{ij})\mathbb{E}(Y_{ik}) \left(e^{\sigma_{jk}} - 1\right).$
Natural extensions

Various tasks of multivariate analysis

- **Dimension Reduction**: rank constraint matrix $\Sigma$.
  \[
  Z_i \sim \mathcal{N}(\mu, \Sigma = CC^\top), \quad C \in \mathcal{M}_{pk} \text{ with orthogonal columns}.
  \]

- **Classification**: maximize separation between groups with means
  \[
  Z_i \sim \mathcal{N}(\mu_k 1_{\{i \in k\}}, \Sigma), \quad \text{for known memberships}.
  \]

- **Clustering**: mixture model in the latent space
  \[
  Z_i \mid i \in k \sim \mathcal{N}(\mu_k, \Sigma_k), \quad \text{for unknown memberships}.
  \]

- **Network inference**: sparsity constraint on inverse covariance.
  \[
  Z_i \sim \mathcal{N}(\mu, \Sigma = \Omega^{-1}), \quad \|\Omega\|_1 < c.
  \]

- **Variable selection**: sparsity constraint on regression coefficients
  \[
  Z_i \sim \mathcal{N}(x_i^\top B, \Sigma), \quad \|B\|_1 < c.
  \]
Oaks powdery mildew data set

Jakuschkin, Fievet, Schwaller, Fort, Robin, and Vacher [Jak+16] Study effects of the pathogen *E.Aphiltoïdes* (mildew) wrt bacterial and microbial communities

Species Abundances

- Microbial communities sampled on the surface of *n = 116* oak leaves
- Communities sequenced and cleaned resulting in *p = 114* OTUs (66 bacteria, 48 fungi).

Covariates and offsets

Characterize the samples and the sampling, most important being

- **tree**: Tree status with respect to the pathogen (susceptible, intermediate or resistant)
- **distTOground**: Distance of the sampled leaf to the base of the ground
- **orientation**: Orientation of the branch (South-West SW or North-East NE)
- **readsTOTfun**: Total number of ITS1 reads for that leaf
- **readsTOTbac**: Total number of 16S reads for that leaf
## Abundance table

### Data table

<table>
<thead>
<tr>
<th>b_OT...</th>
<th>b_OT...</th>
<th>b_OT...</th>
<th>b_OT...</th>
<th>b_OT...</th>
<th>b_OT...</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;int&gt;</td>
<td>&lt;int&gt;</td>
<td>&lt;int&gt;</td>
<td>&lt;int&gt;</td>
<td>&lt;int&gt;</td>
<td>&lt;int&gt;</td>
</tr>
<tr>
<td>146</td>
<td>1</td>
<td>6</td>
<td>6</td>
<td>68</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>121</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>113</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>0</td>
<td>20</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>0</td>
<td>11</td>
<td>316</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>0</td>
<td>8</td>
<td>424</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>0</td>
<td>7</td>
<td>2</td>
<td>312</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>72</td>
<td></td>
</tr>
</tbody>
</table>

Matrix of count (log-scale)
**Offset:** modeling sampling effort

The predefined offset uses the total sum of reads, accounting for technologies specific to fungi and bacteria:

\[
M_{01\_oaks} \leftarrow \text{PLN}(\text{Abundance} \sim 1 + \text{offset}(\log(\text{Offset})), \text{oaks})
\]

**Covariates:** tree and orientation effects ('ANOVA'-like)

The \text{tree} status is a natural candidate for explaining a part of the variance.

- We chose to describe the tree effect in the regression coefficient (mean)
- A possibly spurious effect regarding the interactions between species (covariance).

\[
M_{11\_oaks} \leftarrow \text{PLN}(\text{Abundance} \sim 0 + \text{tree} + \text{offset}(\log(\text{Offset})), \text{oaks})
\]

What about adding more covariates in the model, e.g. the orientation?

\[
M_{21\_oaks} \leftarrow \text{PLN}(\text{Abundance} \sim 0 + \text{tree} + \text{orientation} + \text{offset}(\log(\text{Offset})), \text{oaks})
\]
There is a clear gain in introducing the tree covariate in the model:

```r
rbind(M01 = M01_oaks$criteria,
      M11 = M11_oaks$criteria, M21 = M21_oaks$criteria) %>%
knitr::kable(format = "html")
```

<table>
<thead>
<tr>
<th></th>
<th>nb_param</th>
<th>loglik</th>
<th>BIC</th>
<th>ICL</th>
</tr>
</thead>
<tbody>
<tr>
<td>M01</td>
<td>6669</td>
<td>-32276.98</td>
<td>-48127.83</td>
<td>-52148.35</td>
</tr>
<tr>
<td>M11</td>
<td>6897</td>
<td>-31510.75</td>
<td>-47903.50</td>
<td>-51631.08</td>
</tr>
<tr>
<td>M21</td>
<td>7011</td>
<td>-31422.85</td>
<td>-48086.56</td>
<td>-51703.18</td>
</tr>
</tbody>
</table>

Looking at the coefficients $B$ associated with `tree` bring additional insights:
Discriminant Analysis

Use the `tree` variable for grouping (grouping is a factor of group to be considered)

```r
myLDA_tree <- PLNLDA(Abundance ~ 1 + offset(log(Offset)), grouping = oaks$tree, data = oaks)
```
A PCA analysis of the oaks data set

PCA_offset ← PLNPCA(Abundance ~ 1 + offset(log(Offset)), data = oaks, ranks = 1:30)
PCA: removing covariate effects

To hopefully find some hidden effects in the data, we can try to remove confounding ones:

```r
PCA_tree ← PLNPCA(Abundance ~ 0 + tree + offset(log(Offset)), data = oaks, ranks = 1:30)
```
Clustering of the oaks samples

```r
PLN_mixtures <- PLNmixture(Abundance ~ 1 + offset(log(Offset)), data = oaks, clusters = 1:3)
myPLN_mix <- getModel(PLN_mixtures, 3)
```

myPLN_mix$plot_clustering_pca()

myPLN_mix$plot_clustering_data()
networks ← PLNnetwork(Abundance ~ 0 + tree + offset(log(Offset)), data = oaks)
Availability

Help and documentation

- github group [https://github.com/pln-team](https://github.com/pln-team)
- PLNmodels website [https://pln-team.github.io/PLNmodels](https://pln-team.github.io/PLNmodels)

R/C++ Package **PLNmodels**

Last stable release on CRAN, development version available on GitHub).

```r
install.packages("PLNmodels")
remotes::install_github("PLN-team/PLNmodels@dev")
```

```r
library(PLNmodels)
packageVersion("PLNmodels")
```

### [1] '0.11.7.9500'

Python module **pyPLNmodels**

A Python/PyTorch implementation is about to be published
Simple torch example in R

data("oaks")

system.time(myPLN_torch ←
        PLN(Abundance ~ 1 + offset(log(Offset)),
            data = oaks, control = list(backend = "torch", trace = 0)))

## user  system elapsed
## 2.183  0.016  0.765

system.time(myPLN_nlopt ←
        PLN(Abundance ~ 1 + offset(log(Offset)),
            data = oaks, control = list(backend = "nlopt", trace = 0)))

## user  system elapsed
## 0.584  0.038  0.510

myPLN_torch$loglik

## [1] -32195.9

myPLN_nlopt$loglik

## [1] -32276.98
Variational inference for standard PLN

Optimisation
Inference: general ingredients

Estimate $\theta = (\mathbf{B}, \Sigma)$, predict the $\mathbf{Z}_i$, while the model marginal likelihood is

$$p_\theta(\mathbf{Y}_i) = \int \prod_{j=1}^{p} p_\theta(\mathbf{Y}_{ij} | \mathbf{Z}_{ij}) p_\theta(\mathbf{Z}_i) d\mathbf{Z}_i$$

Expectation-Maximization

With $\mathcal{H}(\mathbf{p}) = -\mathbb{E}_\mathbf{p}(\log(\mathbf{p}))$ the entropy of $\mathbf{p}$,

$$\log p_\theta(\mathbf{Y}) = \mathbb{E}_{p_\theta(\mathbf{Z} | \mathbf{Y})}[\log p_\theta(\mathbf{Y}, \mathbf{Z})] + \mathcal{H}[p_\theta(\mathbf{Z} | \mathbf{Y})]$$

EM requires to evaluate (some moments of) $p_\theta(\mathbf{Z} | \mathbf{Y})$, but there is no close form!

Variational approximation [WJ08]

Use a proxy $q_\psi$ of $p_\theta(\mathbf{Z} | \mathbf{Y})$ minimizing a divergence in a class $\mathcal{Q}$ (e.g., Küllback-Leibler divergence)

$$q_\psi(\mathbf{Z})^* \arg\min_{q \in \mathcal{Q}} D(q(\mathbf{Z}), p(\mathbf{Z} | \mathbf{Y})) \ , \ e.g., \ D(\cdot, \cdot) = KL(\cdot, \cdot) = \mathbb{E}_{q_\psi}[\log \frac{q(z)}{p(z)}] .$$
Consider \( Q \) the class of diagonal multivariate Gaussian distributions:

\[
\left\{ q : q(Z) = \prod_i q_i(Z_i), \quad q_i(Z_i) = \mathcal{N}(Z_i; m_i, \text{diag}(s_i \circ s_i)), \quad \psi_i = (m_i, s_i) \in \mathbb{R}_p \times \mathbb{R}_p \right\}
\]

and maximize the ELBO (Evidence Lower BOund)

\[
J(\theta, \psi) = \log p_\theta(Y) - KL[q_\psi(Z) \| p_\theta(Z|Y)] \\
= \mathbb{E}_\psi[\log p_\theta(Y, Z)] + \mathcal{H}[q_\psi(Z)] \\
= \frac{1}{n} \sum_{i=1}^n J_i(\theta, \psi_i),
\]

where, letting \( A_i = \mathbb{E}_q_i[\exp(Z_i)] = \exp(o_i + m_i + \frac{1}{2}s_i^2) \), we have

\[
J_i(\theta, \psi_i) = Y_i^\top(o_i + m_i) - \left(A_i - \frac{1}{2}\log(s_i^2)\right)^\top 1_p + \frac{1}{2}|\log|\Omega| \\
- \frac{1}{2}(m_i - \Theta x_i)^\top\Omega(m_i - \Theta x_i) - \frac{1}{2}\text{diag}(\Omega)^\top s_i^2 + \text{cst}
\]
Resulting Variational EM

Alternate until convergence between

- VE step: optimize $\psi$ (can be written individually)

$$
\psi_i^{(h)} = \arg \max J_i(\theta^{(h)}, \psi_i) \left( = \arg \min_{q_i} KL[q_i(\mathbf{Z}_i) \| p_{\theta^{(h)}}(\mathbf{Z}_i \mid \mathbf{Y}_i)] \right)
$$

- M step: optimize $\theta$

$$
\theta^{(h)} = \arg \max \frac{1}{n} \sum_{i=1}^{n} J_{Y_i}(\theta, \psi_i^{(h)})
$$

We end up with a $M$-estimator:

$$
\hat{\theta}^{ve} = \arg \max_{\theta} \left( \frac{1}{n} \sum_{i=1}^{n} \sup_{\psi_i} J_i(\theta, \psi_i) \right) = \arg \max_{\theta} \left( \frac{1}{n} \sum_{i=1}^{n} \tilde{J}_i(\theta) \right)
$$

where $\tilde{J}_i(\theta) = \sup_{\psi_i} J_i(\theta, \psi_i)$ is the profiled objective function.
Optimization of simple PLN models

Property of the objective function

The ELBO $J(\theta, \psi)$ is bi-concave, i.e.

- concave wrt $\psi = (M, S)$ for given $\theta$
- convex wrt $\theta = (\Sigma, B)$ for given $\psi$

but not jointly concave in general.

M-step: analytical

\[
\hat{B} = \left( X^\top X \right)^{-1} XM, \quad \hat{\Sigma} = \frac{1}{n} (M - X\hat{B})^\top (M - X\hat{B}) + \frac{1}{n} \text{diag}(1^\top S^2)
\]

VE-step: gradient ascent

\[
\frac{\partial J(\psi)}{\partial M} = (Y - A - (M - XB)\Omega), \quad \frac{\partial J(\psi)}{\partial S} = \frac{1}{S} - S \circ A - SD\Omega.
\]

Same routine for other PLN variants.
Implementations

Medium scale problems (R/C++ package)

- **algorithm**: conservative convex separable approximations [Sva02]
- **implementation**: NLopt nonlinear-optimization library [Joh11]
  \( \mapsto \) Up to thousands of sites \( (n \approx 1000s) \), hundreds of species \( (p \approx 100s) \)

Large scale problems (Python/Pytorch module)

- **algorithm**: Rprop (gradient sign + adaptive variable-specific update) [RB93]
- **implementation**: torch with GPU auto-differentiation [FL22; Pas+17]
  \( \mapsto \) Up to \( n \approx 100,000 \) and \( p \approx 10,000s \)

\( n = 10,000, \ p = 2,000, \ d = 2 \) (running time: 1 min 40s)
Variational estimators of standard PLN

Properties
Estimator Bias and consistency

M-estimation framework [Van00]

Let \( \hat{\psi}_i = \hat{\psi}_i(\theta, Y_i) = \arg \max_\psi J_i(\theta, \psi) \) and consider the stochastic map \( \bar{J}_n \) defined by

\[
\bar{J}_n : \theta \mapsto \frac{1}{n} \sum_{i=1}^{n} J_i(\theta, \hat{\psi}_i) \equiv \frac{1}{n} \sum_{i=1}^{n} \bar{J}_i(\theta)
\]

M-estimation suggests that \( \hat{\theta}^{\text{ve}} = \arg \max_\theta \bar{J}_n(\theta) \) should converge to \( \bar{\theta} = \arg \max_\theta \bar{J}(\theta) \) where \( \bar{J}(\theta) = \mathbb{E}_{\theta^*}[\bar{J}_Y(\theta)] = \mathbb{E}_{\theta^*}[J_Y(\theta, \hat{\psi}(\theta, Y))] \).

Theorem [WM15]

In this line, Westling and McCormick [WM15] show that under regularity conditions ensuring that \( \bar{J}_n \) is smooth enough (e.g. when \( \theta \) and \( \psi_i \) are restricted to compact sets),

\[
\hat{\theta}^{\text{ve}} \xrightarrow{\text{a.e.}} \bar{\theta} \quad \text{as} \quad n \to +\infty
\]

Open question: \( \bar{\theta} = \theta^* \)? No formal results as \( \bar{J} \) is untractable but numerical evidence suggests so.
Study Bias of the estimator of $\hat{B}$

- number of variables $p = 50$
- number of covariates $d \in \{2, 4\}$
- number of samples $n \in \{30, 250, 500, 1000\}$
- sampling effort (TSS) $\approx 10^4$
- $\Sigma$ as $\sigma_{jk} = \sigma^2 \rho^{|j-k|}$, with $\rho = 0.2$
- $\mathbf{B}$ with entries sampled from $\mathcal{N}(0, 1/d)$
- noise level $\sigma^2 \in \{0.25, 1, 4\}$
- 100 replicates
Bias of $\hat{B}$

Bias vanishes with $\hat{n}$
Root mean square error of $\hat{B}$

RMSE vanishes with $n$
Variance: naïve approach

Do as if $\hat{\theta}^{ve}$ was a MLE and $\bar{J}_n$ the log-likelihood.

Variational Fisher Information

The Fisher information matrix is given by (from the Hessian of $J$) by

$$ I_n(\hat{\theta}^{ve}) = \left( \begin{array}{cc} \frac{1}{n} (I_p \otimes X^\top) \text{diag} \left( \text{vec}(A) \right) (I_p \otimes X) & 0 \\ 0 & \frac{1}{2} \Omega^{-1} \otimes \Omega^{-1} \end{array} \right) $$

and can be inverted blockwise to estimate $\hat{V}(\hat{\theta})$.

Confidence intervals and coverage

$$ \hat{V}(B_{kj}) = \left[ n (X^\top \text{diag} \left( \text{vec}(\hat{A}_{.j}) \right) X)^{-1} \right]_{kk}, \quad \hat{V}(\Omega_{kl}) = 2 \hat{\Omega}_{kk} \hat{\Omega}_{ll} $$

The confidence intervals at level $\alpha$ are given by

$$ B_{kj} = \hat{B}_{kj} \pm \frac{q_{1-\alpha/2}}{\sqrt{n}} \sqrt{\hat{V}(B_{kj})}, \quad \Omega_{kl} = \hat{\Omega}_{kl} \pm \frac{q_{1-\alpha/2}}{\sqrt{n}} \sqrt{\hat{V}(\Omega_{kl})}. $$
Variance: empirical vs variational

Variance underestimated...
No trusted confidence intervals can be derived out-of-the-box
**Theorem [WM15]**

Under additional regularity conditions (still satisfied for example when $\theta$ and $\psi_i$ are restricted to compact sets), we have

$$\sqrt{n}(\hat{\theta}^{\text{ve}} - \bar{\theta}) \xrightarrow{d} \mathcal{N}(0, V(\bar{\theta})), \quad \text{where } V(\theta) = C(\theta)^{-1} D(\theta) C(\theta)^{-1}$$

for $C(\theta) = \mathbb{E}[\nabla_{\theta\theta} J(\theta)]$ and $D(\theta) = \mathbb{E} \left[(\nabla_{\theta} J(\theta)) (\nabla_{\theta} J(\theta)^\top)\right]$.

**Practical computations** (chain rule)

$$\hat{C}_n(\theta) = \frac{1}{n} \sum_{i=1}^{n} \left[ \nabla_{\theta\theta} J_i - \nabla_{\theta\psi_i} J_i (\nabla_{\psi_i\psi_i} J_i)^{-1} \nabla_{\theta\psi_i} J_i \right] (\theta, \hat{\psi}_i)$$

$$\hat{D}_n(\theta) = \frac{1}{n} \sum_{i=1}^{n} \left[ \nabla_{\theta} J_i \nabla_{\theta} J_i^\top \right] (\theta, \hat{\psi}_i)$$

**Caveat**

For $\theta = (\mathbf{B}, \Omega)$, $\hat{C}_n$ requires the inversion of $n$ matrices with $(p^2 + pd)$ rows/columns...

We thus first consider the estimation of $\theta = \mathbf{B}$ only, with known variance $\Omega^{-1}$.
Reasonably ugly formula

Additional matrix algebra efforts and computational tricks give

\[ \hat{D}_n(\theta) = \frac{1}{n} \sum_{i=1}^{n} [(\mathbf{Y}_i - \mathbf{A}_i)(\mathbf{Y}_i - \mathbf{A}_i)^\top] \otimes \mathbf{x}_i \mathbf{x}_i^\top \in \mathbb{R}^{dp \times dp} \]

and

\[ \hat{C}_n(\theta) = -\frac{1}{n} \sum_{i=1}^{n} \left( \Sigma + \text{diag}(\mathbf{A}_i)^{-1} + \frac{1}{2} \text{diag}(\mathbf{s}_i^4) \right)^{-1} \otimes \mathbf{x}_i \mathbf{x}_i^\top \in \mathbb{R}^{dp \times dp} \]

\[ \rightsquigarrow \text{Practically not very useful since } \Sigma \text{ is unknown} \]

Ongoing work

Derive the formula with unknown \( \Sigma \)

- Plugin-in \( \hat{\Sigma} \) in the formula of \( \hat{C}_n \) leads very poor results
- Need to account for cross-terms in \( \nabla_{\theta, \psi_i} J_i(\theta, \hat{\psi}_i) \) between \( \Omega \) and \( \psi_i \), and inverse with large matrices: limited practical interest
- Idea: use Jackknife resampling to estimate the variance
95% CI - sandwich coverage

Coverage seems ok with fixed variance matrix
Zero-inflated PLN
A zero-inflated PLN

Motivations

- account for a large amount of zero, i.e. with single-cell data,
- try to separate "true" zeros from "technical"/dropouts

The Model

Use two latent vectors $\mathbf{W}_i$ and $\mathbf{Z}_i$ to model excess of zeroes and dependence structure

\[
\mathbf{Z}_i \sim \mathcal{N}(\mathbf{o}_i + \mathbf{x}_i^\top \mathbf{B}, \Sigma)
\]

\[
W_{ij} \sim \mathcal{B}(\logit^{-1}(\mathbf{x}_i^\top \mathbf{B}_j^0))
\]

\[
Y_{ij} \mid W_{ij}, Z_{ij} \sim W_{ij}\delta_0 + (1 - W_{ij})\mathcal{P}(\exp\{Z_{ij}\})
\]

The unknown parameters are

- $\mathbf{B}$, the regression parameters (from the PLN component)
- $\mathbf{B}^0$, the regression parameters (from the Bernoulli component)
- $\Sigma$, the variance-covariance matrix

$\sim$ ZI-PLN is a mixture of PLN and Bernoulli distribution with shared covariates.
Variational approximation

\[ p(Z_i, W_i Y_i) \approx q_\psi(Z_i, W_i) \approx q_\psi_1(Z_i) q_\psi_2(W_i) \]

with

\[ q_\psi_1(Z_i) = \mathcal{N}(Z_i; m_i, \text{diag}(s_i \circ s_i)), \quad q_\psi_2(W_i) = \bigotimes_{j=1}^p \mathcal{B}(W_{ij}, \pi_{ij}) \]

Variational lower bound

Let \( \theta = (B, B^0, \Sigma) \) and \( \psi = (M, S, \Pi) \), then

\[ J(\theta, \psi) = \log p_\theta(Y) - KL(p_\theta(\cdot | Y) \| q_\psi(\cdot)) \]

\[ = \mathbb{E}_{q_\psi} \log p_\theta(Z, W, Y) - \mathbb{E}_{q_\psi} \log q_\psi(Z, W) \]

\[ = \mathbb{E}_{q_\psi} \log p_\theta(Y|Z, W) + \mathbb{E}_{q_\psi_1} \log p_\theta(Z) + \mathbb{E}_{q_\psi_2} \log p_\theta(W) \]

\[ - \mathbb{E}_{q_\psi_1} \log q_\psi_1(Z) - \mathbb{E}_{q_\psi_2} \log q_\psi_2(W) \]

**Property:** \( J \) is separately concave in \( \theta, \psi_1 \) and \( \psi_2 \).
Sparse regularization

Criterion

Recall that $\theta = (B, B^0, \Omega = \Sigma^{-1})$. Sparsity allows to control the number of parameters:

$$\arg\min_{\theta, \psi} J(\theta, \psi) + \lambda_1 \|B\|_1 + \lambda_2 \|\Omega\|_1 + \lambda_1 \|B^0\|_1$$

Alternate optimization

- (Stochastic) Gradient-descent on $B^0, M, S$
- Closed-form for posterior probabilities $\Pi$
- Inverse covariance $\Omega$
  - if $\lambda_2 = 0$, $\hat{\Sigma} = n^{-1} \left[ (M - XB)^\top (M - XB) + \bar{S}^2 \right]$
  - if $\lambda_2 > 0$, $\ell_1$ penalized MLE ( $\leadsto$ Graphical-Lasso with $\hat{\Sigma}$ as input)
- PLN regression coefficient $B$
  - if $\lambda_1 = 0$, $\hat{B} = [X^\top X]^{-1} X^\top M$
  - if $\lambda_1 > 0$, vectorize and solve a $\ell_1$ penalized least-squared problem

Initialize $B^0$ with logistic regression on $\delta_0(Y)$, $B$ with Poisson regression
A quick example in genomics (1)

scRNA data set

The dataset scRNA contains the counts of the 500 most varying transcripts in the mixtures of 5 cell lines in human liver (obtained with standard 10x scRNAseq Chromium protocol).

We subsample 500 random cells and the keep the 200 most varying genes

```r
library(PLNmodels); library(ZIPLN)
data(scRNA); set.seed(1234)
scRNA <- scRNA[sample.int(nrow(scRNA), 500), ]
scRNA$counts <- scRNA$counts[, 1:200]
scRNA$counts %>% as_tibble() %>% rmarkdown::paged_table()
```

<table>
<thead>
<tr>
<th>KRT81 &lt;int&gt;</th>
<th>AKR1B10 &lt;int&gt;</th>
<th>LCN2 &lt;int&gt;</th>
<th>AKR1C2 &lt;int&gt;</th>
<th>ALDH1A1 &lt;int&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>117</td>
<td>82</td>
<td>0</td>
<td>41</td>
<td>21</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>
A quick example in genomics (2)

Model fits

We adjust the standard PLN model and the ZI-PLN model with some sparsity on the precision matrix:

```r
system.time(myPLN <-
  PLN(counts ~ 1 + offset(log(total_counts)),
       data = scRNA, control = list(trace = 0, xtol_rel = 1e-4)))
```

```
##    user  system elapsed
##   6.05   0.20    5.10
```

```r
system.time(myZIPLN <-
  ZIPLN(counts ~ 1 + offset(log(total_counts)), rho = .2,
        data = scRNA, control = list(trace = 0)))
```

```
##    user  system elapsed
##  14.89   0.21   10.72
```
A quick example in genomics (3)

ZI-PLN seems to be less variant for predicting small counts
prcomp(myZIPLN$latent) %>% factoextra::fviz_pca_ind(col.ind = scRNA$cell_line)
See Sophie Donnet's talk for more about Stochastic Block Models

```r
library(sbm)
A ← myZIPLN$model_par$Omega ≠ 0; diag(A) ← 0
mySBM ← estimateSimpleSBM(A, estimOptions=list(plot=FALSE))
```
Conclusion

Summary

- PLN = generic model for multivariate count data analysis
- Flexible modeling of the covariance structure, allows for covariates
- Efficient V-EM algorithm
- Variational estimator is asymptotically normal (and hopefully unbiased) with computable covariance matrix.
- ZI-PLN reduces (some) problems induced by high sparsity in the data

Work in progress

- Caracterisation of Variational Estimator
  - with J. Stoehr: Direct likelihood optim (SGD with Important Sampling)
  - with J. Kwon: optimisation guarantees coupling adaptive SGD + variance reduction
- Connection/Comparison with VAE with e.g Poisson neg log-likelihood as loss

Advertisement

https://computo.sfds.asso.fr, a journal promoting reproducible research in ML and stat.


