Poisson lognormal models for count data

Variational inference, Optimization

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

MIA Paris-Saclay, AgroParisTech, INRAE, Sorbonne Universyty

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**
  $\rightsquigarrow$ explanatory models (multivariate regression, classification)
- exhibit **patterns of diversity**
  $\rightsquigarrow$ summarize the information (clustering, dimension reduction)
- understand **between-species interactions**
  $\rightsquigarrow$ 'network' inference (variable/covariance selection)
- correct for technical and **confounding effects**
  $\rightsquigarrow$ 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 $i = 1, \ldots, n$,

$$\underbrace{\mathbf{Y}_i}_{\text{abundances}} = \underbrace{\mathbf{x}_i^\top \mathbf{B}}_{\text{covariates}} + \underbrace{\mathbf{o}_i}_{\text{sampling effort}} + \boldsymbol{\varepsilon}_i, \quad \boldsymbol{\varepsilon}_i \sim \mathcal{N}(\mathbf{0}_p, \underbrace{\boldsymbol{\Sigma}}_{\text{between-species dependencies}})$$

null covariance $\Leftrightarrow$ independence $\rightsquigarrow$ 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, $\sqrt{}$) : 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.

$$\mathbf{Y}_i|\mathbf{Z}_i \sim \mathcal{P}\left(\exp \mathbf{Z}_i\right), \qquad \mathbf{Z}_i \sim \mathcal{N}(\mathbf{o}_i + \mathbf{x}_i^\top \mathbf{B}, \mathbf{\Sigma}),$$

The unkwown parameters are

- $\mathbf{B}$, the regression parameters
- $\mathbf{\Sigma}$, 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: $\mathbb{E}(Y_{ij}) = \exp\left(o_{ij} + \mathbf{x}_i^\top \mathbf{B}_{\cdot j} + \sigma_{jj}/2\right) > 0$
- variance: $\mathbb{V}(Y_{ij}) = \mathbb{E}(Y_{ij}) + \mathbb{E}(Y_{ij})^2 \left(e^{\sigma_{jj}} - 1\right) > \mathbb{E}(Y_{ij})$
- covariance: $\mathrm{Cov}(Y_{ij}, Y_{ik}) = \mathbb{E}(Y_{ij})\mathbb{E}(Y_{ik}) \left(e^{\sigma_{jk}} - 1\right).$

Natural extensions

Various tasks of multivariate analysis

- Dimension Reduction: rank constraint matrix $\boldsymbol{\Sigma}$.

$$\mathbf{Z}_i \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma} = \mathbf{C}\mathbf{C}^\top), \quad \mathbf{C} \in \mathcal{M}_{pk} \text{ with orthogonal columns.}$$

- Classification: maximize separation between groups with means

$$\mathbf{Z}_i \sim \mathcal{N}(\boldsymbol{\mu}_k \mathbf{1}_{\{i \in k\}}, \boldsymbol{\Sigma}), \quad \text{for known memberships.}$$

- Clustering: mixture model in the latent space

$$\mathbf{Z}_i \mid i \in k \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \quad \text{for unknown memberships.}$$

- Network inference: sparsity constraint on inverse covariance.

$$\mathbf{Z}_i \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma} = \boldsymbol{\Omega}^{-1}), \quad \|\boldsymbol{\Omega}\|_1 < c.$$

- Variable selection: sparsity constraint on regression coefficients

$$\mathbf{Z}_i \sim \mathcal{N}(\mathbf{x}_i^\top \mathbf{B}, \boldsymbol{\Sigma}), \quad \|\mathbf{B}\|_1 < c.$$

Illustration on ecological data (eDNA)

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

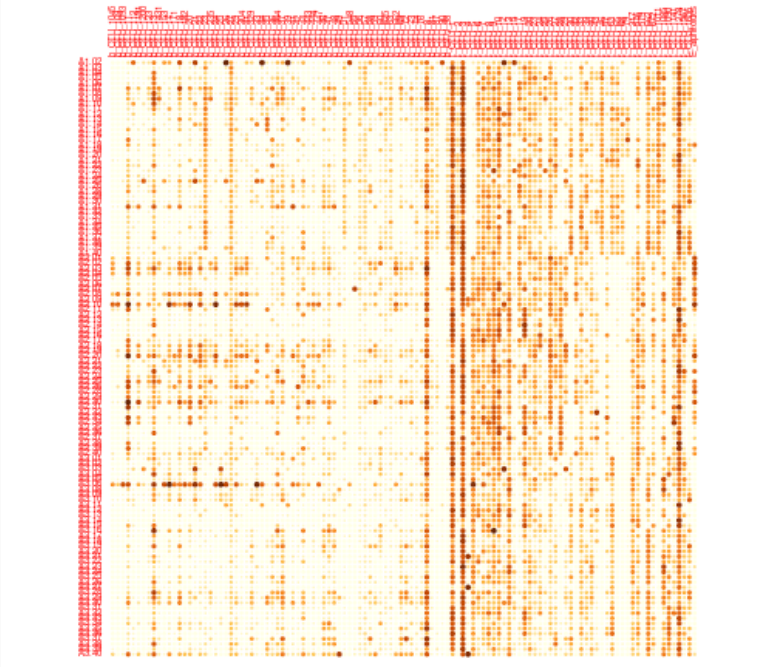
Abundance table

Data table

b_OT...	b_OT...	b_OT...	b_OT...	b_OT...
<int>	<int>	<int>	<int>	<int>
146	1	6	6	68
0	1	0	0	4
0	0	0	0	128
1	1	0	4	121
1	1	1	0	113
2	20	0	20	90
2	3	0	11	316
4	3	0	8	424
42	0	7	2	312
2	0	0	4	72

1-10 of 116... Previous **1** 2 3 ... 12 Next

Matrix of count (log-scale)



PLN with offsets and covariates (1)

Offset: modeling sampling effort

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

```
M01_oaks <- PLN(Abundance ~ 1 + offset(log(Offset)) , oaks)
```

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

The `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).

```
M11_oaks <- PLN(Abundance ~ 0 + tree + offset(log(Offset)), oaks)
```

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

```
M21_oaks <- PLN(Abundance ~  0 + tree + orientation + offset(log(Offset)), oaks)
```

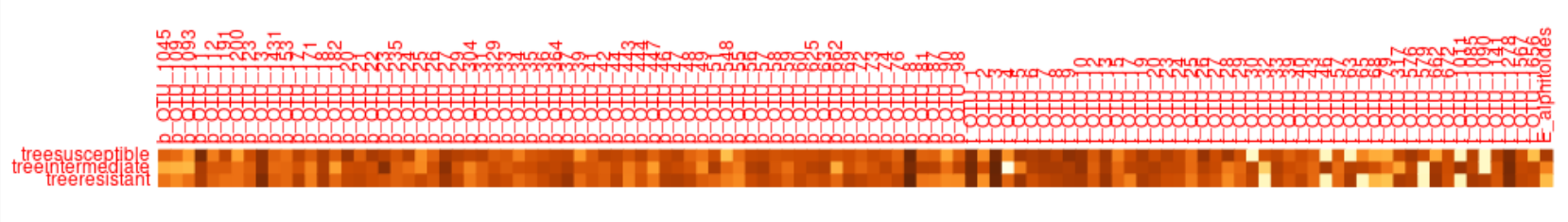
PLN with offsets and covariates (2)

There is a clear gain in introducing the tree covariate in the model:

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

	nb_param	loglik	BIC	ICL
M01	6669	-32276.98	-48127.83	-52148.35
M11	6897	-31510.75	-47903.50	-51631.08
M21	7011	-31422.85	-48086.56	-51703.18

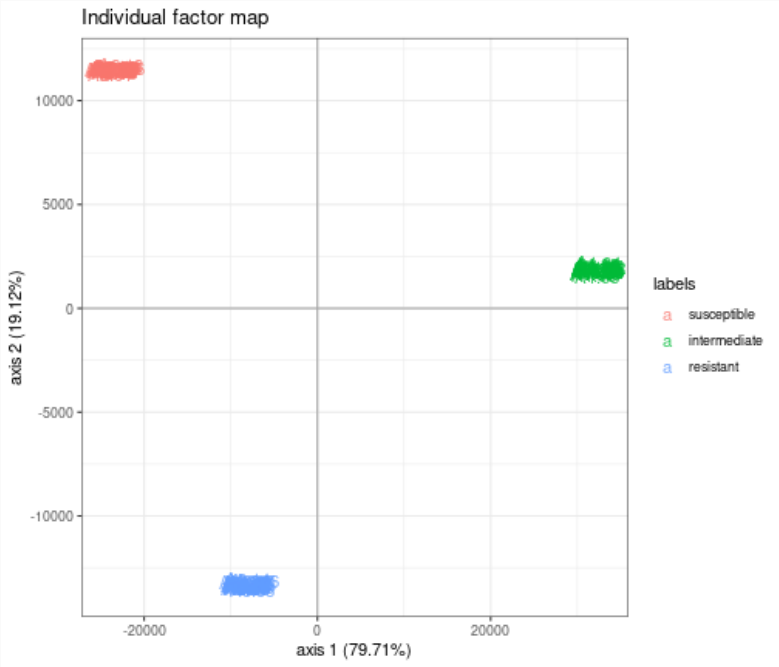
Looking at the coefficients $\mathbf{B}$ associated with `tree` bring additional insights:

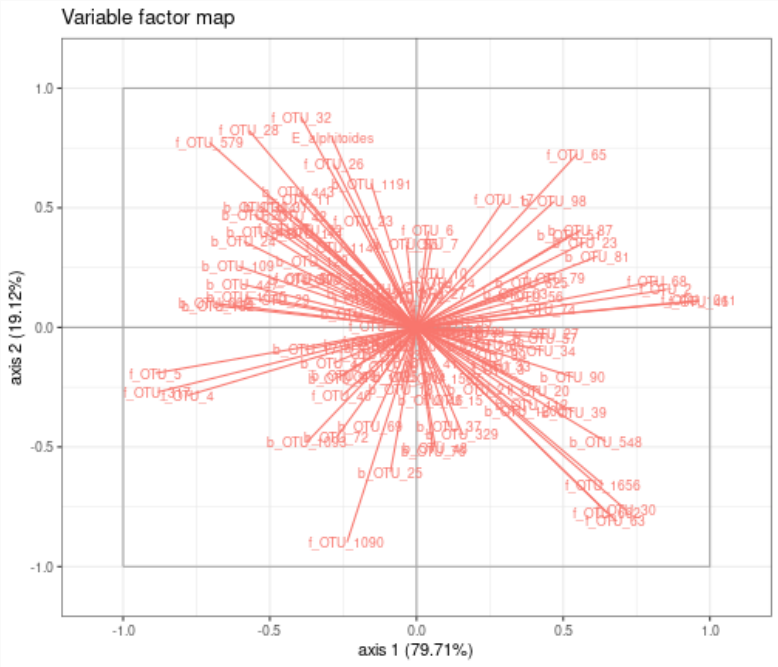


Discriminant Analysis

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

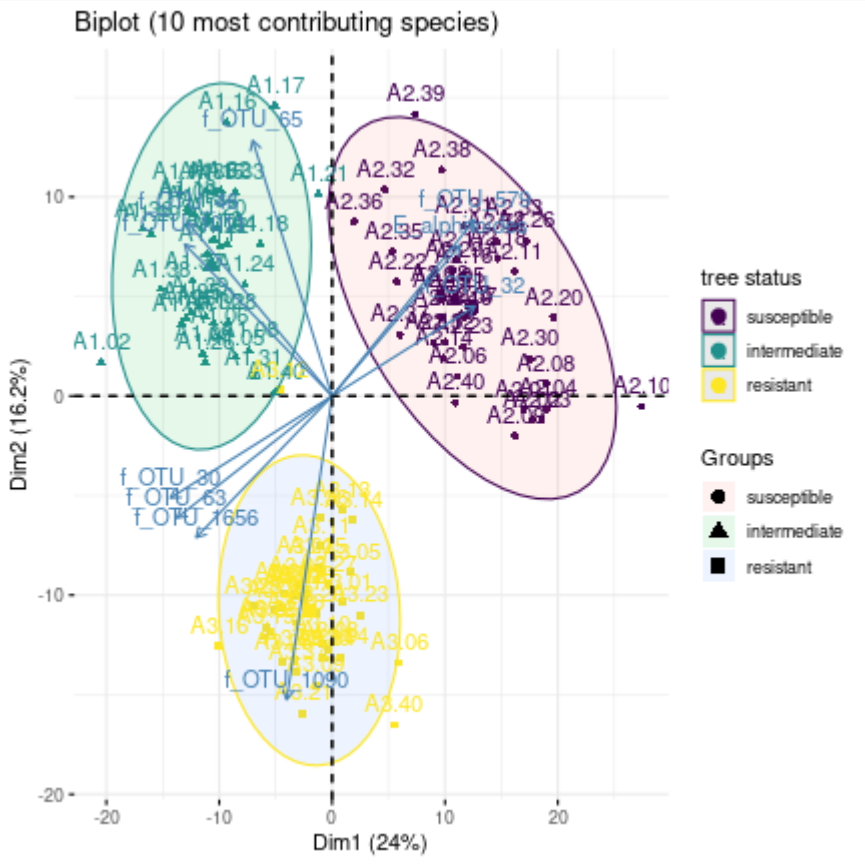
```
myLDA_tree ←
  PLNLDA(Abundance ~ 1 + offset(log(Offset)), grouping = oaks$tree, data = oaks)
```





A PCA analysis of the oaks data set

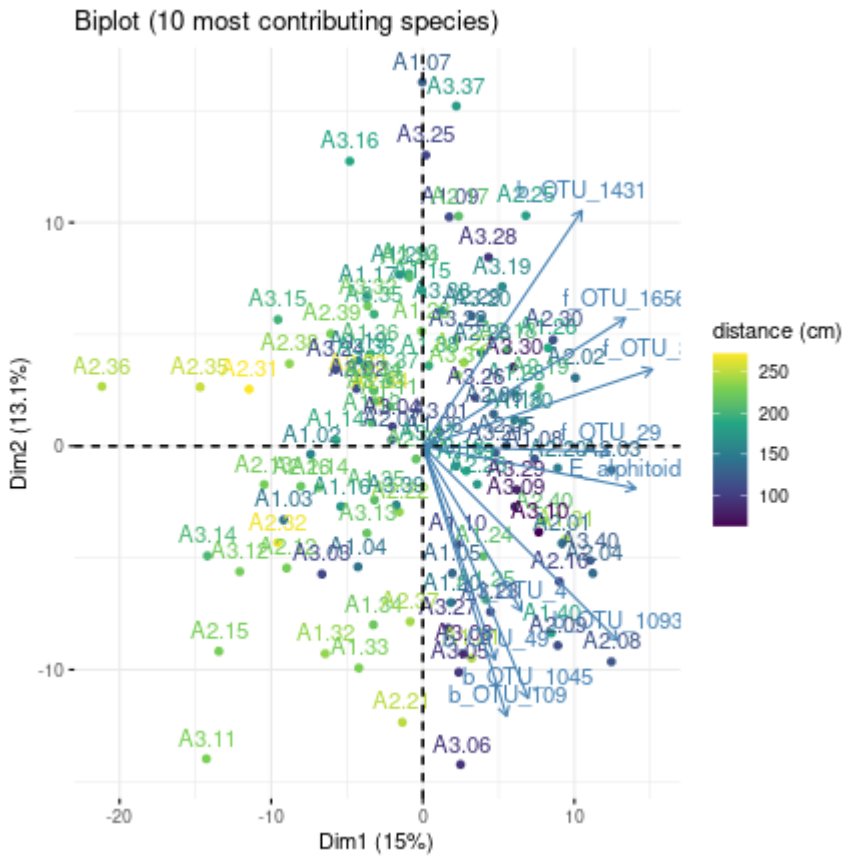

```
PCA_offset ← PLNPCA(Abundance ~ 1 + offset(log(Offset)), data = oaks, ranks = 1:30)
```



PCA: removing covariate effects

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

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

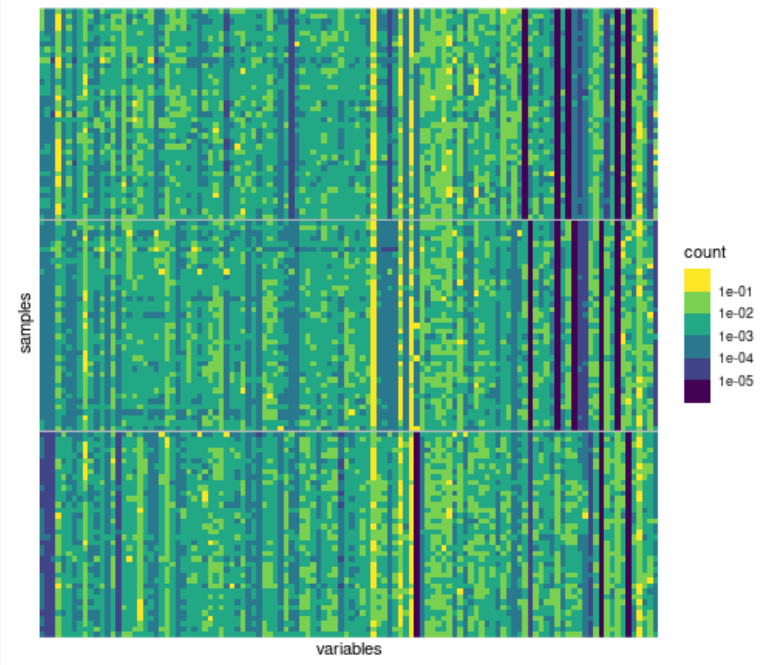


Clustering of the oaks samples

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

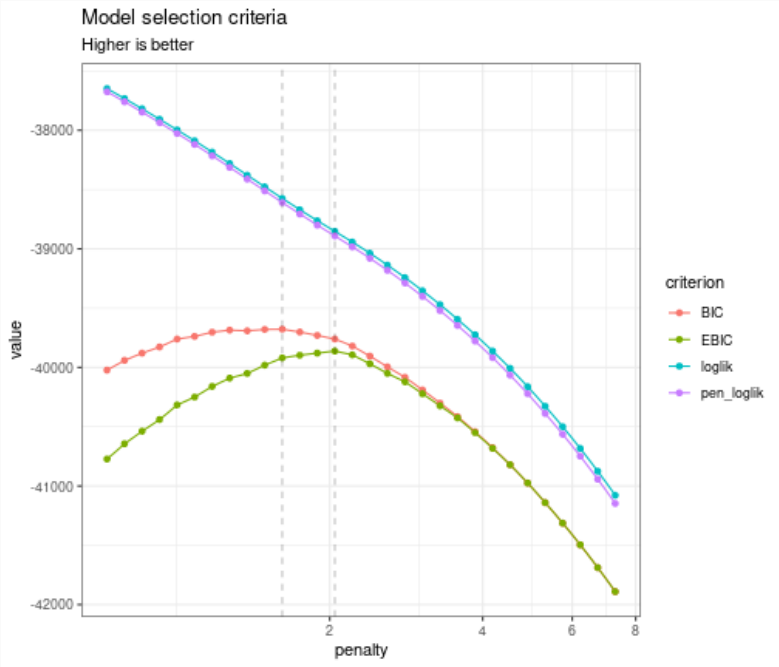
```
myPLN_mix$plot_clustering_pca()
```

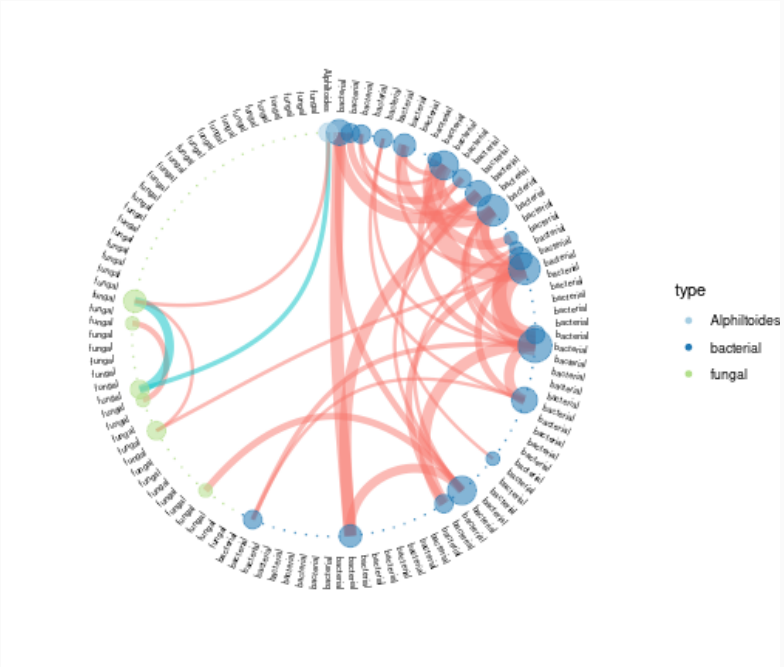

```
myPLN_mix$plot_clustering_data()
```



Network inference


```
networks ← PLNnetwork(Abundance ~ 0 + tree + offset(log(Offset)), data = oaks)
```





Availability

Help and documentation

- github group https://github.com/pln-team
- PLNmodels website https://pln-team.github.io/PLNmodels

R/C++ Package `PLNmodels`

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

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

```
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


```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
```

```r
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
```

```r
myPLN_torch$loglik
```

```
## [1] -32195.9
```

```r
myPLN_nlopt$loglik
```

```
## [1] -32276.98
```

Variational inference for standard PLN

Optimisation

Inference: general ingredients

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

$$p_\theta(\mathbf{Y}_i) = \int_{\mathbb{R}_p} \prod_{j=1}^{p} p_\theta(Y_{ij}|Z_{ij})\, p_\theta(\mathbf{Z}_i) \mathrm{d}\mathbf{Z}_i$$

Expectation-Maximization

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

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

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

Variational approximation [WJ08]

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

$$q_\psi(\mathbf{Z})^\star \arg\min_{q \in \mathcal{Q}} D\left(q(\mathbf{Z}), p(\mathbf{Z}|\mathbf{Y})\right), \text{ e.g., } D(.,.) = KL(.,.) = \mathbb{E}_{q_\psi}\left[\log \frac{q(z)}{p(z)}\right].$$

Inference: specific ingredients

Consider $\mathcal{Q}$ the class of diagonal multivariate Gaussian distributions:

$$\left\{q:\ q(\mathbf{Z}) = \prod_i q_i(\mathbf{Z}_i),\ q_i(\mathbf{Z}_i) = \mathcal{N}\left(\mathbf{Z}_i; \mathbf{m}_i, \mathrm{diag}(\mathbf{s}_i \circ \mathbf{s}_i)\right), \boldsymbol{\psi}_i = (\mathbf{m}_i, \mathbf{s}_i) \in \mathbb{R}_p \times \mathbb{R}_p\right\}$$

and maximize the ELBO (Evidence Lower BOund)

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

where, letting $\mathbf{A}_i = \mathbb{E}_{q_i}[\exp(Z_i)] = \exp\left(\mathbf{o}_i + \mathbf{m}_i + \frac{1}{2}\mathbf{s}_i^2\right)$, we have

$$\begin{aligned} J_i(\theta, \psi_i) =& \mathbf{Y}_i^\intercal(\mathbf{o}_i + \mathbf{m}_i) - \left(\mathbf{A}_i - \frac{1}{2}\log(\mathbf{s}_i^2)\right)^\intercal \mathbf{1}_p + \frac{1}{2}|\log|\boldsymbol{\Omega}| \\ & - \frac{1}{2}(\mathbf{m}_i - \boldsymbol{\Theta}\mathbf{x}_i)^\intercal \boldsymbol{\Omega}(\mathbf{m}_i - \boldsymbol{\Theta}\mathbf{x}_i) - \frac{1}{2}\mathrm{diag}(\boldsymbol{\Omega})^\intercal \mathbf{s}_i^2 + \mathrm{cst} \end{aligned}$$

Resulting Variational EM

Alternate until convergence between

- VE step: optimize $\boldsymbol{\psi}$ (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 θ

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

We end up with a M-estimator:

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

where $\bar{J}_i(\theta) = \sup_{\psi_i} J_i(\theta, \psi_i)$ 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 $\psi = (\mathbf{M}, \mathbf{S})$ for given θ
- convace wrt $\theta = (\mathbf{\Sigma}, \mathbf{B})$ for given ψ

but not jointly concave in general.

M-step: analytical

$$\hat{\mathbf{B}} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}\mathbf{M}, \quad \hat{\mathbf{\Sigma}} = \frac{1}{n}\left(\mathbf{M} - \mathbf{X}\hat{\mathbf{B}}\right)^\top \left(\mathbf{M} - \mathbf{X}\hat{\mathbf{B}}\right) + \frac{1}{n}\mathrm{diag}(\mathbf{1}^\intercal \mathbf{S}^2)$$

VE-step: gradient ascent

$$\frac{\partial J(\psi)}{\partial \mathbf{M}} = \left(\mathbf{Y} - \mathbf{A} - (\mathbf{M} - \mathbf{X}\mathbf{B})\mathbf{\Omega}\right), \qquad \frac{\partial J(\psi)}{\partial \mathbf{S}} = \frac{1}{\mathbf{S}} - \mathbf{S} \circ \mathbf{A} - \mathbf{S}\mathrm{D}_{\mathbf{\Omega}}.$$

$\rightsquigarrow$ Same routine for other PLN variants.

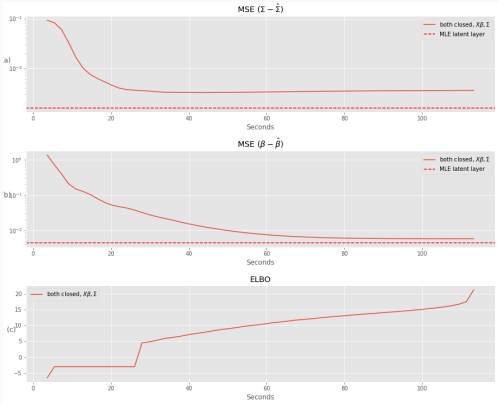
Implementations

Medium scale problems (R/C++ package)

- **algorithm**: conservative convex separable approximations [Sva02]
- **implementation**: `NLopt` nonlinear-optimization library [Joh11]
  $\rightsquigarrow$ 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]
  $\rightsquigarrow$ 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 $\hat{\psi}_i = \hat{\psi}_i(\theta, \mathbf{Y}_i) = \arg\max_\psi J_i(\theta, \psi)$ and consider the stochastic map $\bar{J}_n$ defined by

$$\bar{J}_n \ : \quad \theta \mapsto \frac{1}{n}\sum_{i=1}^n J_i(\theta, \hat{\psi}_i) \triangleq \frac{1}{n}\sum_{i=1}^n \bar{J}_i(\theta)$$

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

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 θ and ψ_i are restricted to compact sets),

$$\hat{\theta}^{\mathrm{ve}} \xrightarrow[n\to+\infty]{a.e.} \bar{\theta}$$

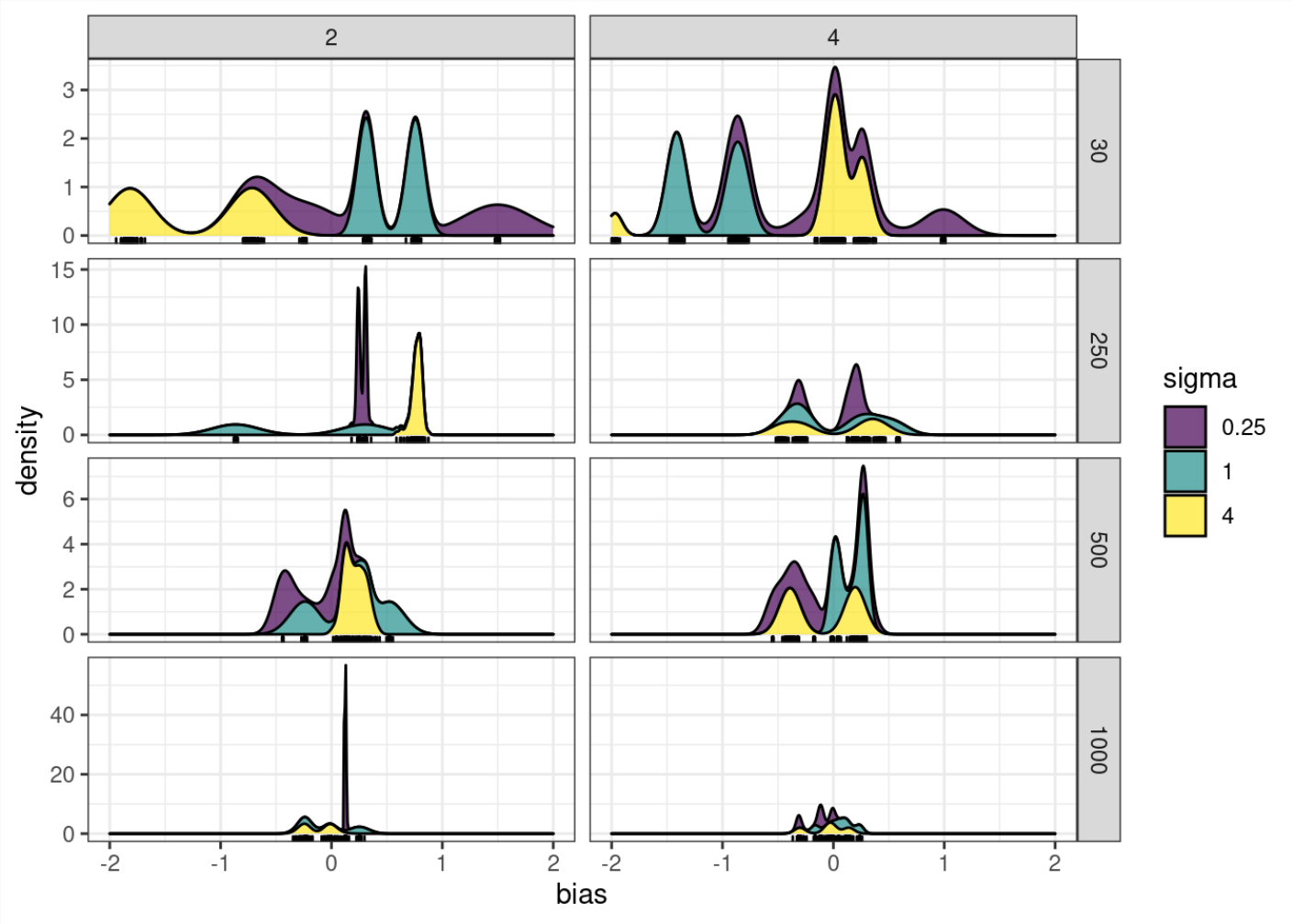
Open question: $\bar{\theta} = \theta^\star$? No formal results as $\bar{J}$ is untractable but numerical evidence suggests so.

Numerical study

Study Bias of the estimator of $\hat{\boldsymbol{B}}$

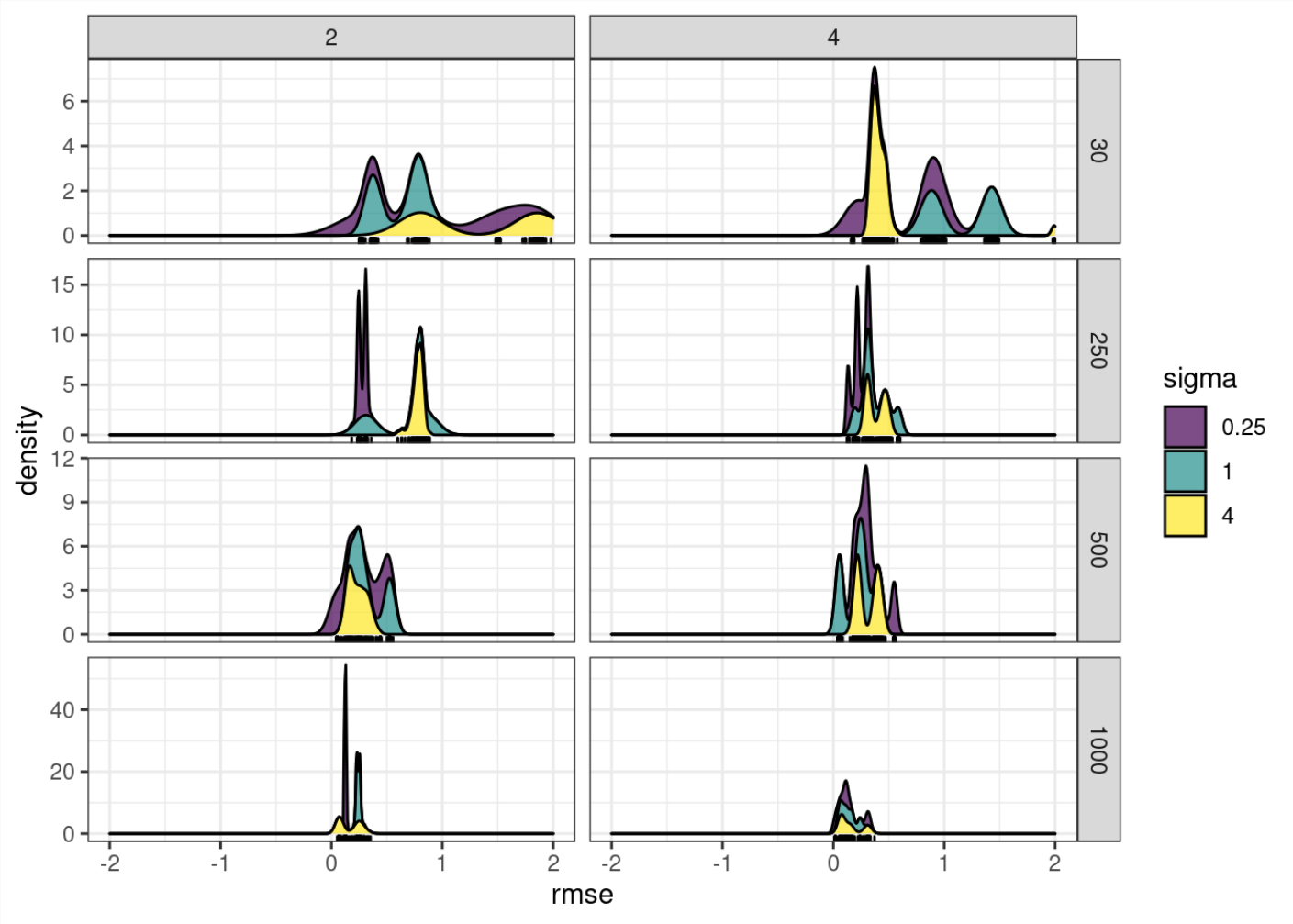
- number of variables $p = 50$
- number of covariates $d \in \{2, 4\}$
- number of samples $n \in \{30, 250, 500, 1000\}$
- sampling effort (TSS) $\approx 10^4$
- $\boldsymbol{\Sigma}$ as $\sigma_{jk} = \sigma^2 \rho^{|j-k|}$, with $\rho = 0.2$
- $\mathbf{B}$ with entries sampled from $\mathcal{N}(0,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

Bias of $\hat{B}$



Bias vanishes with `$n$`

Root mean square error of $\hat{B}$



RMSE vanishes with `$n$`

Variance: naïve approach

Do as if $\hat{\theta}^{\mathrm{ve}}$ 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

$$I_n(\hat{\theta}^{\mathrm{ve}}) = \begin{pmatrix} \frac{1}{n}(\mathbf{I}_p \otimes \mathbf{X}^\top)\mathrm{diag}(\mathrm{vec}(\mathbf{A}))(\mathbf{I}_p \otimes \mathbf{X}) & \mathbf{0} \\ \mathbf{0} & \frac{1}{2}\boldsymbol{\Omega}^{-1} \otimes \boldsymbol{\Omega}^{-1} \end{pmatrix}$$

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

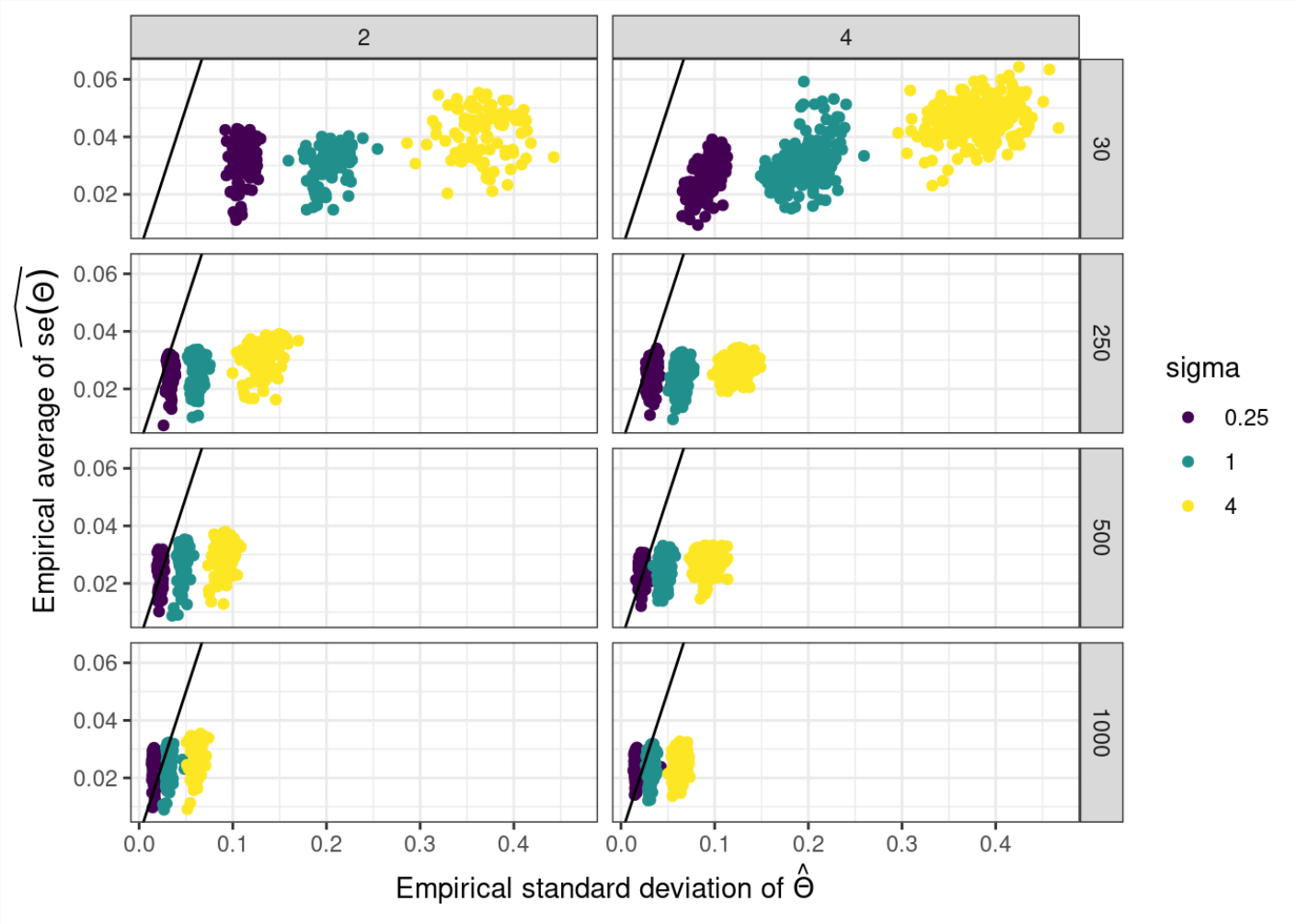
Confidence intervals and coverage

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

The confidence intervals at level α are given by

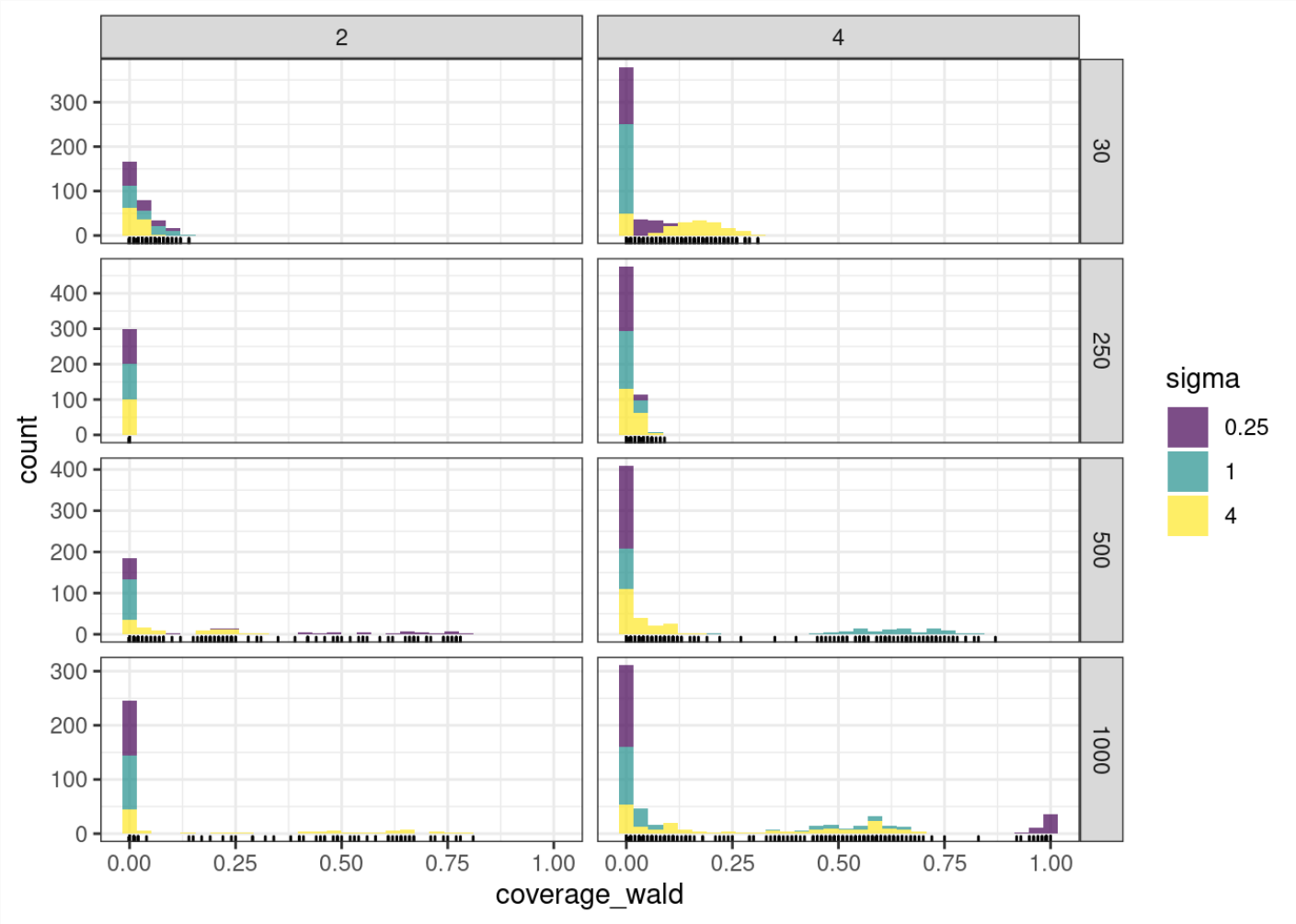
$$B_{kj} = \hat{B}_{kj} \pm \frac{q_{1-\alpha/2}}{\sqrt{n}}\sqrt{\hat{\mathbb{V}}(B_{kj})}, \qquad \Omega_{kl} = \hat{\Omega}_{kl} \pm \frac{q_{1-\alpha/2}}{\sqrt{n}}\sqrt{\hat{\mathbb{V}}(\Omega_{kl})}.$$

Variance: empirical vs variational



Variance underestimated...

95% confident interval - coverage



No trusted confidence intervals can be derived out-of-the box

Variance : sandwich estimator

Theorem [WM15]

Under additional regularity conditions (still satisfied for example when $\boldsymbol{\theta}$ and $\boldsymbol{\psi}_i$ are restricted to compact sets), we have

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

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

Practical computations chain rule

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

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

Caveat

For $\theta = (\mathbf{B}, \boldsymbol{\Omega})$, $\hat{C}_n$ 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 $\boldsymbol{\Omega}^{-1}$

Reasonably ugly formula

Additional matrix algebra efforts and computational tricks give

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

and

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

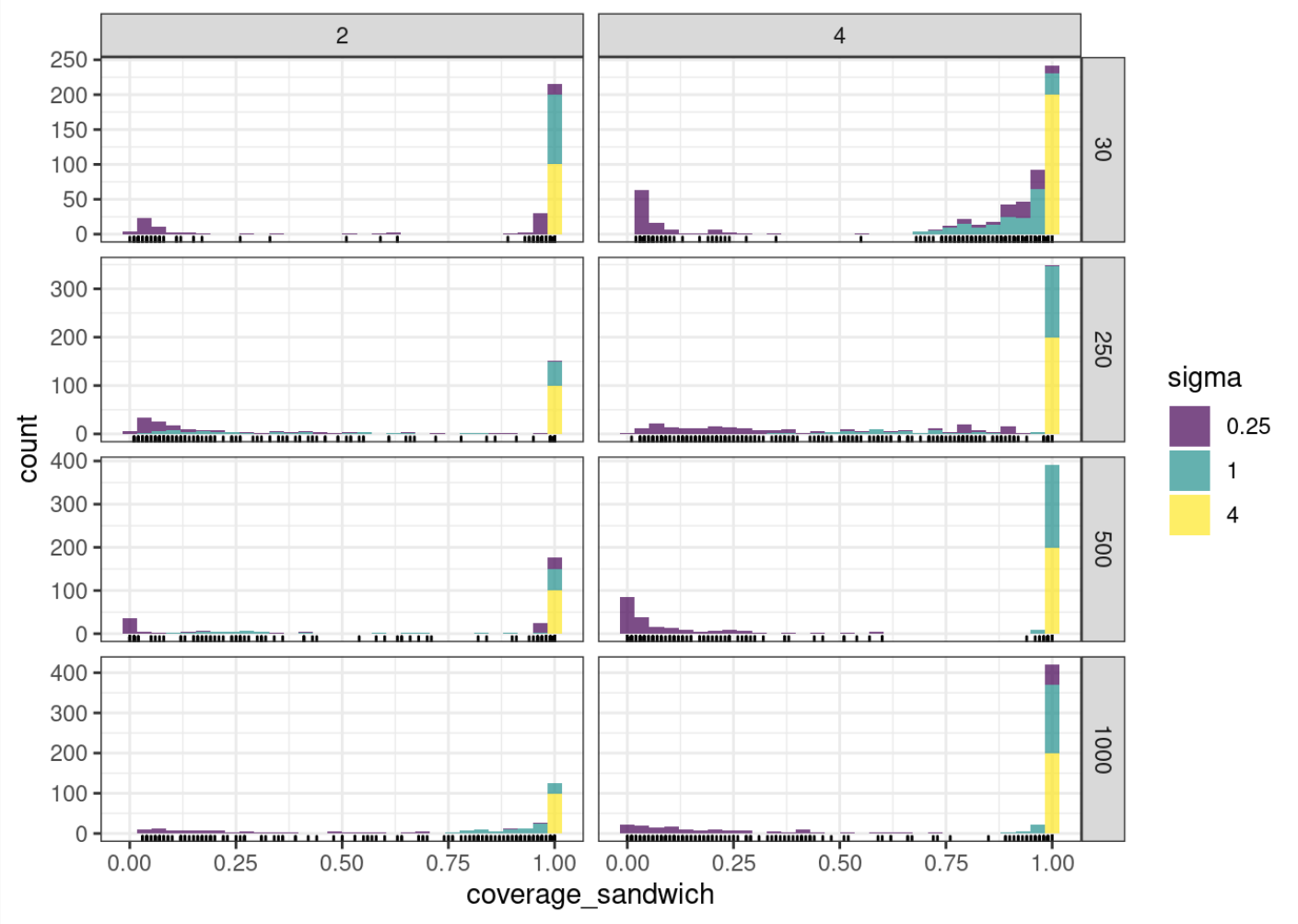
$\rightsquigarrow$ Practically not very useful since Σ is unknown

Ongoing work

Derive the formula with unknown $\mathbf{\Sigma}$

- 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 Ω and ψ_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

$$\begin{aligned}
\mathbf{Z}_i &\sim \mathcal{N}(\mathbf{o}_i + \mathbf{x}_i^\top \mathbf{B}, \mathbf{\Sigma}) \\
W_{ij} &\sim \mathcal{B}(\text{logit}^{-1}(\mathbf{x}_i^\top \mathbf{B}_j^0)) \\
Y_{ij} \,|\, W_{ij}, Z_{ij} &\sim W_{ij}\delta_0 + (1 - W_{ij})\mathcal{P}\left(\exp\{Z_{ij}\}\right),
\end{aligned}$$

The unkwown parameters are

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

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

ZI-PLN Inference

Same routine...

Variational approximation

$$p(\mathbf{Z}_i, \mathbf{W}_i \mathbf{Y}_i) \approx q_\psi(\mathbf{Z}_i, \mathbf{W}_i) \approx q_{\psi_1}(\mathbf{Z}_i) q_{\psi_2}(\mathbf{W}_i)$$

with

$$q_{\psi_1}(\mathbf{Z}_i) = \mathcal{N}(\mathbf{Z}_i; \mathbf{m}_i, \mathrm{diag}(\mathbf{s}_i \circ \mathbf{s}_i)), \qquad q_{\psi_2}(\mathbf{W}_i) = \otimes_{j=1}^p \mathcal{B}(W_{ij}, \pi_{ij})$$

Variational lower bound

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

$$\begin{aligned} J(\theta, \psi) &= \log p_\theta(\mathbf{Y}) - KL(p_\theta(. \,|\mathbf{Y}) \| q_\psi(.)) \\ &= \mathbb{E}_{q_\psi} \log p_\theta(\mathbf{Z}, \mathbf{W}, \mathbf{Y}) - \mathbb{E}_{q_\psi} \log q_\psi(\mathbf{Z}, \mathbf{W}) \\ &= \mathbb{E}_{q_\psi} \log p_\theta(\mathbf{Y} | \mathbf{Z}, \mathbf{W}) + \mathbb{E}_{q_{\psi_1}} \log p_\theta(\mathbf{Z}) + \mathbb{E}_{q_{\psi_2}} \log p_\theta(\mathbf{W}) \\ &\quad - \mathbb{E}_{q_{\psi_1}} \log q_{\psi_1}(\mathbf{Z}) - \mathbb{E}_{q_{\psi_2}} \log q_{\psi_2}(\mathbf{W}) \end{aligned}$$

Property: J is separately concave in θ, ψ_1 and ψ_2.

Sparse regularization

Criterion

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

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

Alternate optimization

- (Stochastic) Gradient-descent on $\mathbf{B}^0, \mathbf{M}, \mathbf{S}$
- Closed-form for posterior probabilities $\boldsymbol{\Pi}$
- Inverse covariance $\boldsymbol{\Omega}$
  - if $\lambda_2 = 0$, $\hat{\boldsymbol{\Sigma}} = n^{-1} \left[(\mathbf{M} - \mathbf{XB})^\top (\mathbf{M} - \mathbf{XB}) + \bar{\mathbf{S}}^2 \right]$
  - if $\lambda_2 > 0$, ℓ_1 penalized MLE ($\rightsquigarrow$ Graphical-Lasso with $\hat{\boldsymbol{\Sigma}}$ as input)
- PLN regression coefficient $\mathbf{B}$
  - if $\lambda_1 = 0$, $\hat{\mathbf{B}} = [\mathbf{X}^\top \mathbf{X}]^{-1} \mathbf{X}^\top \mathbf{M}$
  - if $\lambda_1 > 0$, vectorize and solve a ℓ_1 penalized least-squared problem

Initialize B^0 with logistic regression on $\delta_0(\mathbf{Y})$, $\mathbf{B}$ 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

```
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()
```

KRT81	AKR1B10	LCN2	AKR1C2	ALDH1A1
<int>	<int>	<int>	<int>	<int>
1	0	1	0	0
3	1	3	0	0
117	82	0	41	21
1	2	2	0	0
2	1	0	0	2

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:

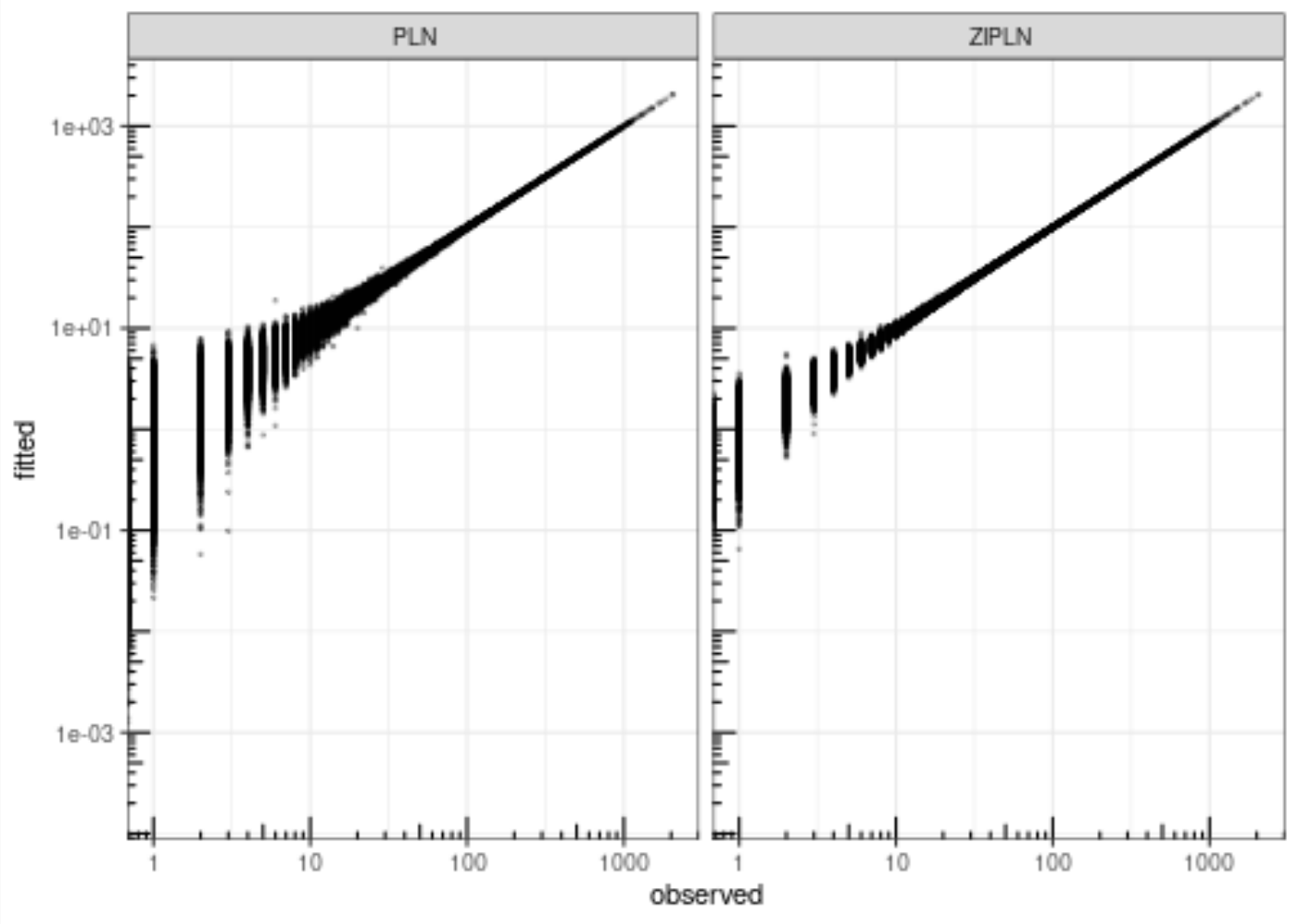
```
system.time(myPLN <-
    PLN(counts ~ 1 + offset(log(total_counts)),
        data = scRNA, control = list(trace = 0, xtol_rel = 1e-4)))
```

```
##    user  system elapsed
##   6.055   0.209   5.100
```

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

```
##    user  system elapsed
##  14.899   0.211  10.723
```

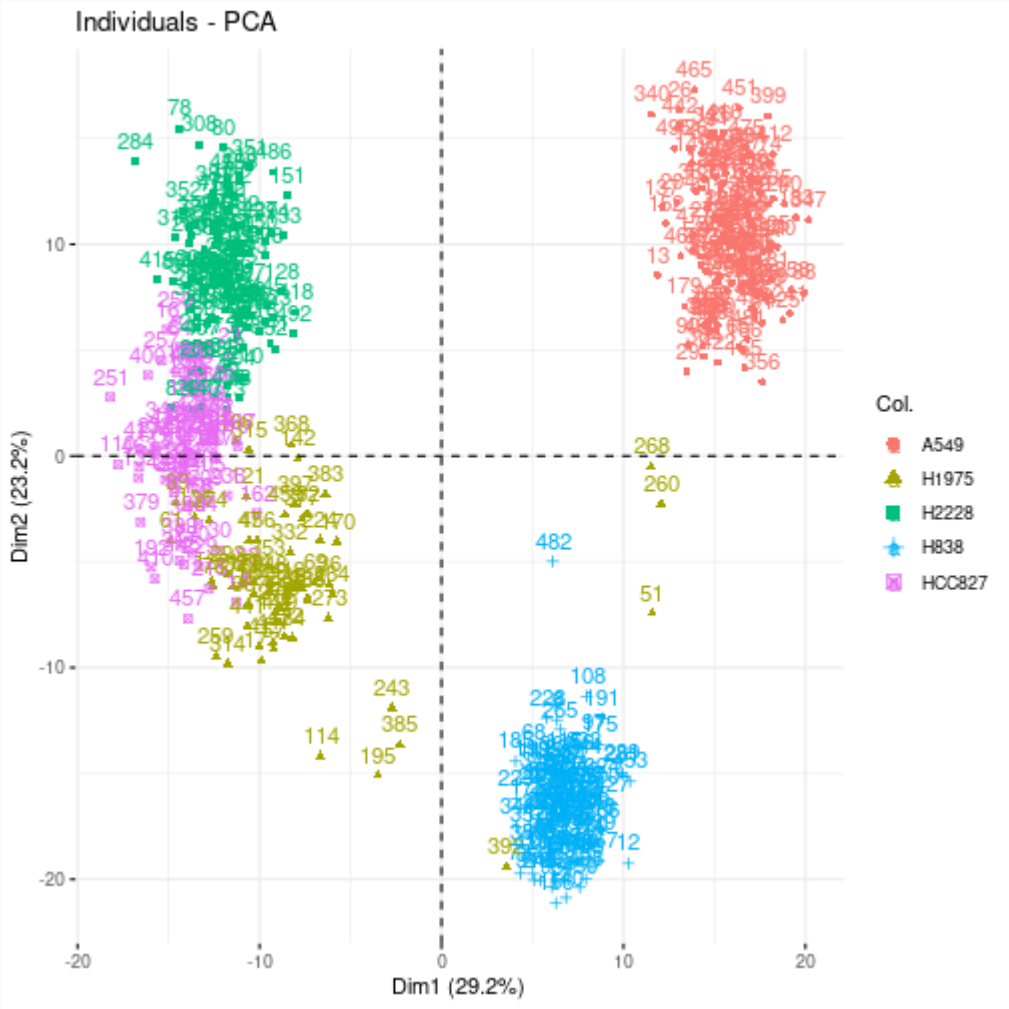
A quick example in genomics (3)



ZI-PLN seems to be less variant for predicting small counts

A quick example in genomics (4)


```
prcomp(myZIPLN$latent) %>% factoextra::fviz_pca_ind(col.ind = scRNA$cell_line)
```



A quick example in genomics (5)

See Sophie Donnet's talk for more about Stochastic Block Models

```
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.

References

Aitchison, J. and C. Ho (1989). "The multivariate Poisson-log normal distribution". In: *Biometrika* 76.4, pp. 643-653.

Chiquet, J., M. Mariadassou, and S. Robin (2018). "Variational inference for probabilistic Poisson PCA". In: *The Annals of Applied Statistics* 12, pp. 2674-2698. URL: http://dx.doi.org/10.1214/18-AOAS1177.

Chiquet, J., M. Mariadassou, and S. Robin (2019). "Variational inference for sparse network reconstruction from count data". In: *Proceedings of the 19th International Conference on Machine Learning (ICML 2019)*.

Chiquet, J., M. Mariadassou, and S. Robin (2021). "The Poisson-Lognormal Model as a Versatile Framework for the Joint Analysis of Species Abundances". In: *Frontiers in Ecology and Evolution* 9. DOI: 10.3389/fevo.2021.588292.

Facon, B., A. Hafsi, M. C. de la Masselière, et al. (2021). "Joint species distributions reveal the combined effects of host plants, abiotic factors and species competition as drivers of species abundances in fruit flies". In: *Ecological Letters*. DOI: 10.1111/ele.13825.

Falbel, D. and J. Luraschi (2022). *torch: Tensors and Neural Networks with 'GPU' Acceleration*. https://torch.mlverse.org/docs, https://github.com/mlverse/torch.

Inouye, D. I., E. Yang, G. I. Allen, et al. (2017). "A review of multivariate distributions for count data derived from the Poisson distribution". In: *Wiley Interdisciplinary Reviews: Computational Statistics* 9.3.

Jakuschkin, B., V. Fievet, L. Schwaller, et al. (2016). "Deciphering the pathobiome: intra-and interkingdom interactions involving the pathogen Erysiphe alphitoides". In: *Microbial ecology* 72.4, pp. 870-880.

Johnson, S. G. (2011). *The NLopt nonlinear-optimization package*. URL: http://ab-initio.mit.edu/nlopt.