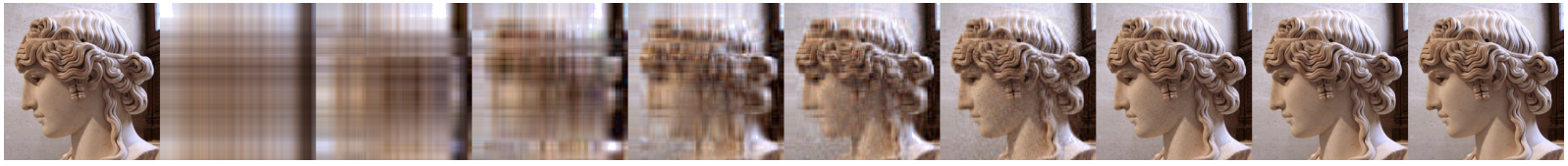


PLN-PCA à la sauce bayésienne du faible rang. All you need is a SVD prior! (Battacharya & Dunson's prior)

Eric Parent, Pierre Gloaguen, Achille Thin

Rochebrune, Stats aux sommets, Mars 2024

Outline 1/2

- Réduire la dimension: intérêt des "features latents," grande dimension, parcimonie et "explicabilité." Bensecri vs Deheuvels.
  – En version classique, on procéde souvent en deux temps.
  - $(X, Y) -> F$ (analyse descriptive par extraction des PCA) puis
  - $Y_{new} = g(F, X_{new})$ (étude fonctionnelle) La géométrie projective casse l'incertitude, qui revient dans l'art de l'interprétation ($\pm$ poétique) des composantes principales.

  – En bayésien hiérarchique notamment, on veut simplement un "module" qui prenant X, Y en entrée donne $[F|Y, X]$ avec certaines propriétés probabilistes. D'où le rôle d'un a priori structurant $[F]$.
- Hypothèses et Modèle
  – La géométrie SVD
  – Le modèle *probabiliste* d'Analyse en Composantes Principales
  – Bhattacharya & Dunson's Gamma Process as a *shrinking* prior et vive la conjugaison Gamma Normale!

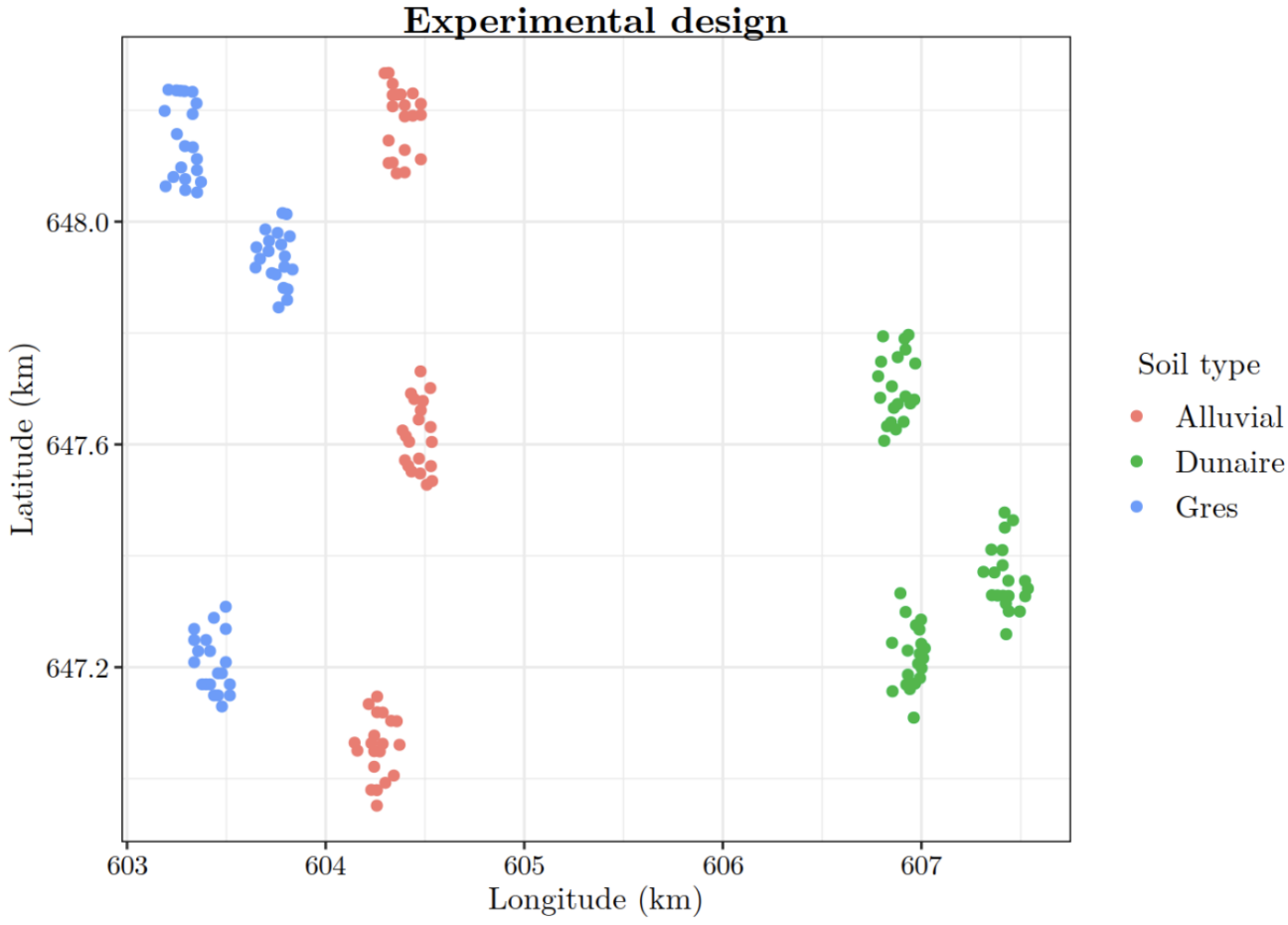
{Le slide précédent montrait les approximations de rang 1, 2, 4, 8, 16, 32, 64, 128 puis full rank de l'image du buste d'Antinoüs Mondragone, villa Mondragone à Frascati (Wikipedia, Sharayanan)}

Outline 2/2 qu'on ne fera pas

- Extensions théoriques immédiates
  – LEGO bricks : Gibbs, Inference Variationnelle, GLM
  – Hoffman & Blei:Coordinate Ascent VI pour l'optimisation du gradient
  – Réseaux de neurones
- Applications
  – Ovaskainen et al : MCMC for Joint Species Distribution Modelling
  – Codes publicitly available on
  `https://github.com/papayoun/VI_fo_Poisson_PCA`
- Conclusions : Et Bayes ?

Historique : Mon stage auprès du Pr. Gloaguen

Pierre va à Kourou rencontre Giacommo Sellan. Données d'abondance forestière de Bornéo.



Historique : Mon stage auprès du Pr. Gloaguen

Un jeu de données "R for data science" pour tous les élèves de l'agro . . .

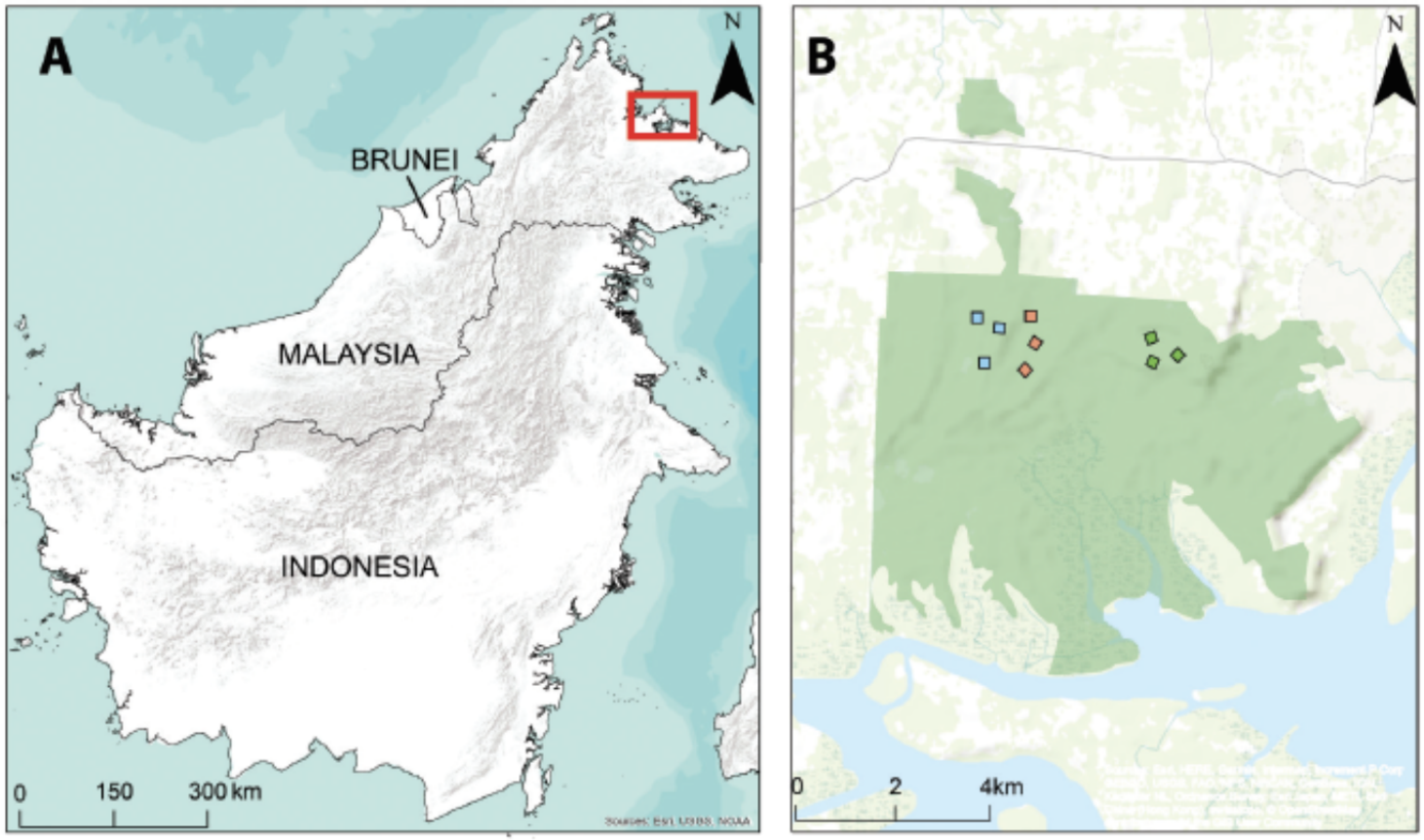
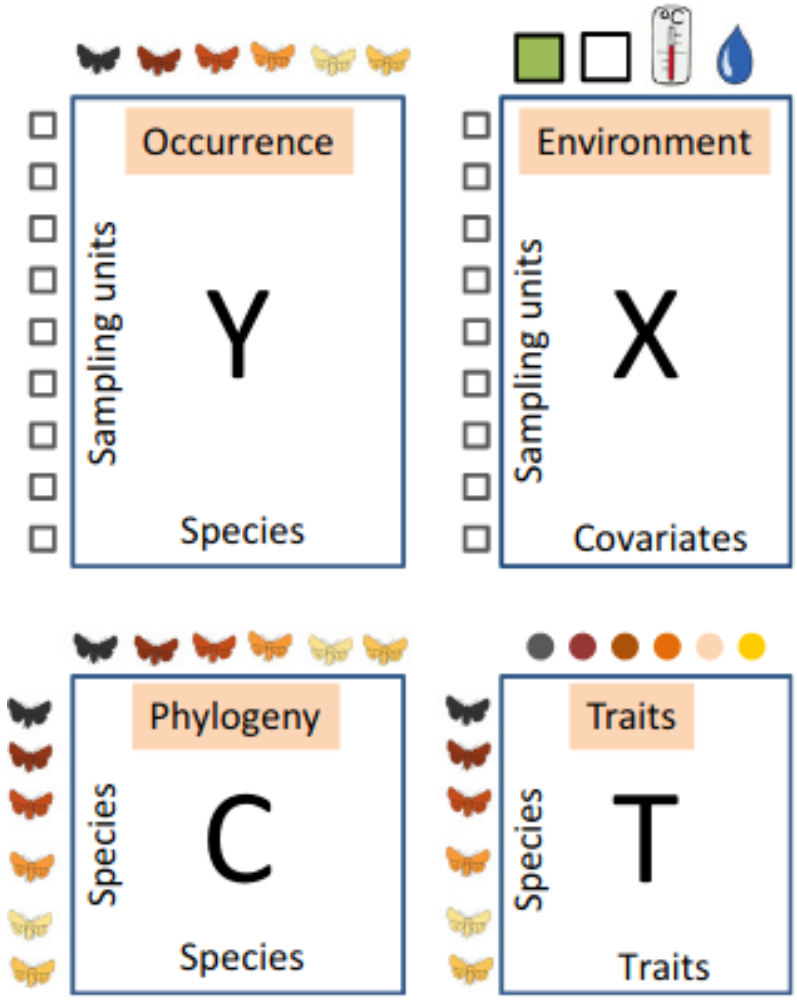


Figure 1 Study sites: (A) the island of Borneo with the approximate position of the Kabili–Sepilok Forest Reserve in Sabah, Malaysia highlighted in red; (B) the nine study plots within the KSFR (dark green); blue squares represent the sandstone plots, red squares represent the alluvial plots and green squares represent the heath forest plots

Figure 1: Lieu de récolte des données, d'après (Sellan 2021)

Historique : Bornéo

Historique : Ovaskainen et le package Hmsc.



Ovaskainen, Otso and Abrego, Nerea.(2020), Joint species distribution modelling: with applications in R, Cambridge University Press.

Historique : Joint Species Modelling (Ovaskainen) grâce à Hmsc

$$\underset{n\times p}{Y} \sim \mathcal{P}(e^{Z})$$

n sites et p espèces.

$$Z_i \sim \mathcal{N}(\underset{1\times m}{X_i}\ \underset{m\times p}{\beta}\ ,\ \underset{p\times p}{\Sigma})$$

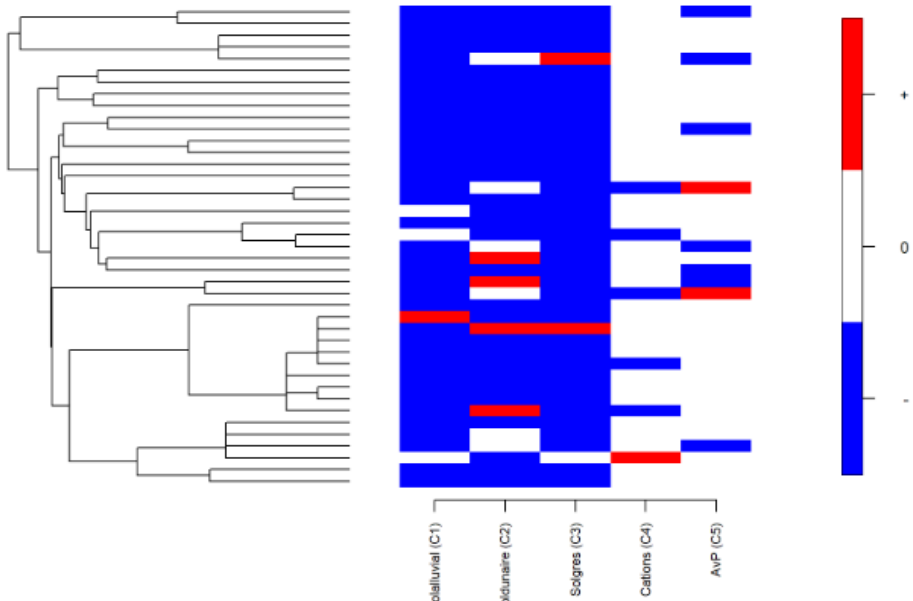
m variables environnementales

$$\underset{m\times p}{\beta} \sim \mathcal{N}(\underset{(m\times t)(t\times p)}{\Gamma\ T'}, \rho\ \underset{p\times p}{C} + (1-\rho)I$$

t traits généraux de l'espèce

- Priors
  - sur Γ, ρ
  - sur Σ

Moments d'ordre un (Hmsc)



Décomposition en valeurs singulières (SVD)

- On veut étudier la structure d'une base de données massives, un tableau **M** (n *individus* $\times p$ *variables*). Par exemple, en écologie, les individus seraient des unités de recueil de données $i = 1 : n$ et les variables les espèces d'intérêt $j = 1 : p$ dont on mesure une caractéristique $M_{i,j}$.
- Théorème:

$$\underset{n\times p}{M} = \underset{(n\times n)}{U}\ \underset{(n\times p)}{D}\ \underset{(p\times p)}{V}{}'$$

– U matrice unitaire (rotation) $n \times n$,
– D une matrice n$\times$p dont les coefficients diagonaux sont des réels positifs ou nuls
– V' transposée de V, matrice unitaire $p \times p$.

Décomposition en valeurs singulières (SVD)

- Principe de la démo:
  – Trouver un premier jeu de deux vecteurs $\underset{1\times n}{u}$ et v tels que

$$\max u'Mv - \frac{\sigma_u}{2}u'u - \frac{\sigma_v}{2}v'v$$

La *maximisation* donne $M'u = \sigma_v v$ et $Mv = \sigma_u u$ soit

$$MM'u = \sigma_u\sigma_v u$$

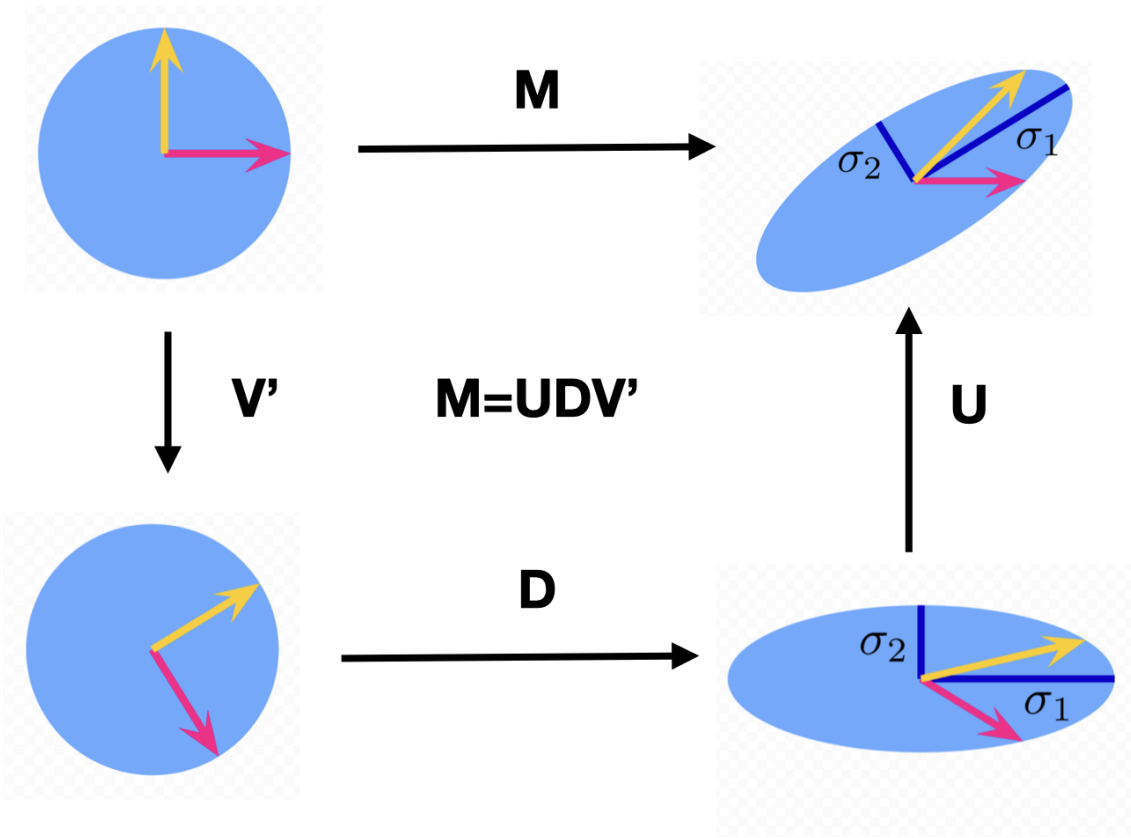
$$M'Mv = \sigma_u\sigma_v v$$

u est vecteur propre de MM', v est vecteur propre de MM', on montre aussi $\sigma_u = \sigma_v = \sigma$.

- Comme pour l'ACP, on cherche ensuite deux vecteurs unitaires orthogonaux aux premiers optenus maximisant $u'Mv$, etc., et on range les valeurs propres par ordre de magnitude décroissante. On construit ainsi les "rotations" U et V d'où

$$\underset{n\times p}{M} = \underset{(n\times n)}{U}\ \underset{(n\times p)}{D}\ \underset{(p\times p)}{V}'$$

Interprétation géométrique de la SVD



Pour toute application linéaire de $\mathbb{R}^p \to \mathbb{R}^n$, on peut trouver une base orthonormale pour $\mathbb{R}^p$ et une base orthonormale pour $\mathbb{R}^n$ telles que l'on associe au i-ème vecteur de base de $\mathbb{R}^p$ un multiple positif du i-ème vecteur de base de $\mathbb{R}^n$, les vecteurs restants ayant pour image 0. Dans ces bases, l'application est donc ainsi représentée par une matrice diagonale dont les coefficients sont des réels positifs.

SVD : un exemple par simulation

```
p <- 20; k <- 2; n <- 15
Etroit <- matrix(runif(n*k), nrow = n, ncol = k)
Long <- matrix(runif(p*k), nrow = p, ncol = k)
residus= matrix(rnorm(n*p,0,0.01), nrow = n, ncol = p)
M <- Etroit%*%t(Long)+residus; svdM= svd(M); str(svdM)
```

```
List of 3
 $ d: num [1:15] 9.0142 1.4724 0.0735 0.0668 0.0637 ...
 $ u: num [1:15, 1:15] -0.103 -0.261 -0.118 -0.142 -0.193 ...
 $ v: num [1:20, 1:15] -0.285 -0.267 -0.224 -0.372 -0.154 ...
```

```
print(sum(abs(M-svdM$u%*%diag(svdM$d)%*%t(svdM$v))^2))
```

```
[1] 1.409471e-28
```

```
diag(t(svdM$u)%*%svdM$u)
```

```
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
diag(t(svdM$v)%*%svdM$v)
```

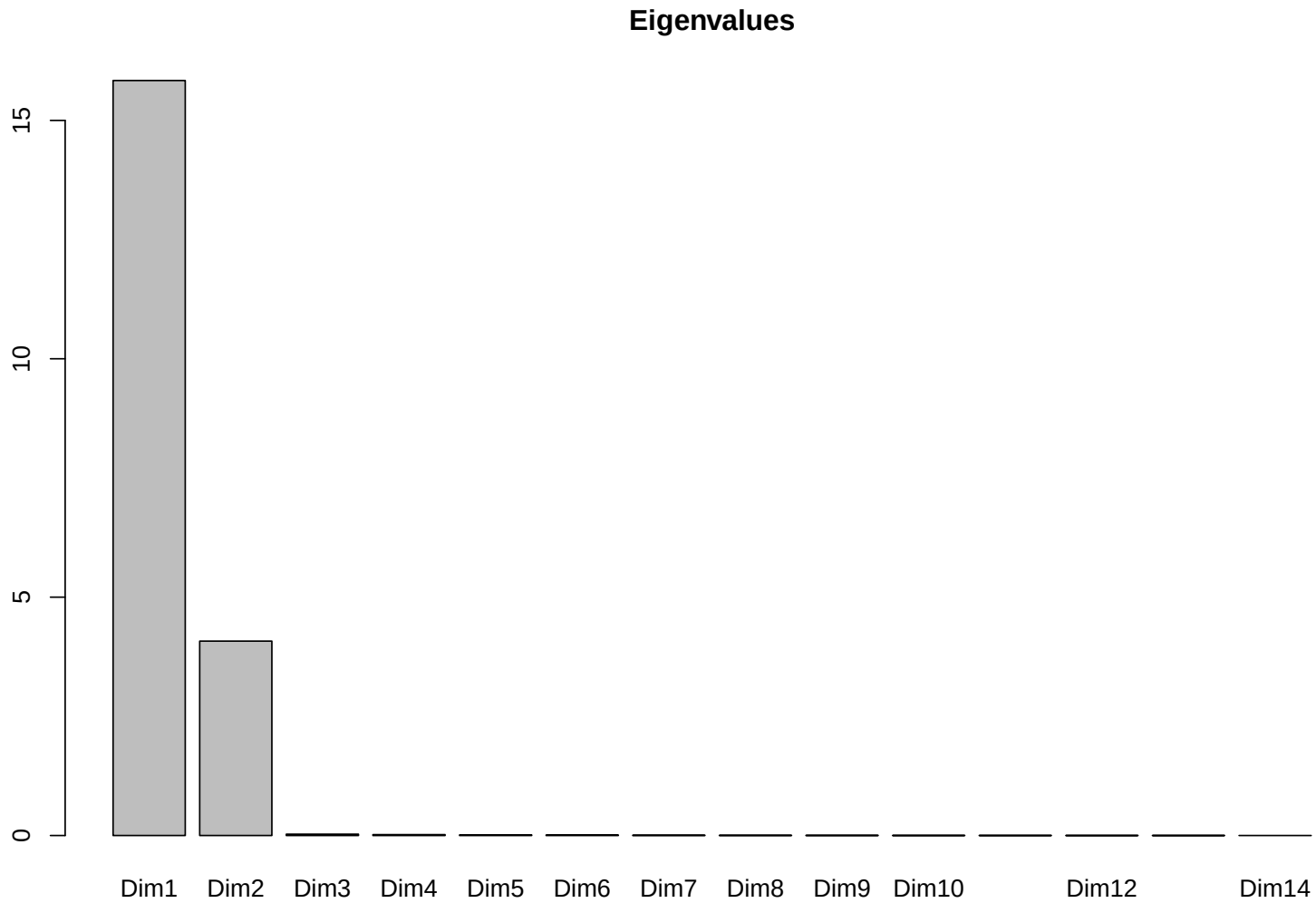
```
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
print((eigen(svdM$v%*%t(svdM$v), only.values = TRUE))$values[1:min(30,m
```

```
 [1]  1.0e+00  1.0e+00  1.0e+00  1.0e+00  1.0e+00  1.0e+00  1.0e+00  1.0e+00  1.0e+00  1.0e+00  1.0e+00  1.0e+00
[13]  1.0e+00  1.0e+00  1.0e+00  3.7e-16  8.5e-18 -1.1e-16 -1.4e-16 -4.7e-16
```

SVD : un exemple par simulation

```
library(FactoMineR)
res.pca <- PCA(M, graph=FALSE)
barplot(res.pca$eig[,1], main = "Eigenvalues",
  names.arg = paste("Dim", 1:nrow(res.pca$eig), sep = ""))
```



Vers une ACP probabiliste

- Imaginons la structure d'une base de données massives $\mathbf{Y}$, *n individus* $\times p$ *variables*). D'abord si variables explicatives d'environnement X, l'effet fixe β.
- Création d'un Modèle Normal Multivarié
  Inspirons nous de la SVD du tableau $Y - X\beta$ pour écrire, avec k *quelconque* (même si bien sûr, dans un objectif de réduction de dimension on choisira $k \leq min(p, n)$):

$$\underset{(n\times p)}{Y - X\beta} \approx \underset{(n\times k)}{U}\ \underset{(k\times k)}{diag(d_k^2)}\ \underset{(k\times p)}{V} + \underset{n\times p}{\epsilon}$$

soit, pour la ligne i de Y:

$$Y_i \approx X_i\beta + \begin{bmatrix} u_i^1 & u_i^2 & \dots & u_i^k \end{bmatrix} \begin{bmatrix} d_1^2 & 0 & \dots & 0 \\ 0 & \ddots & \vdots & \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & \dots & d_k^2 \end{bmatrix} \begin{bmatrix} v_1^1 & v_1^2 & \dots & v_1^p \\ \vdots & \vdots & \vdots & \vdots \\ v_k^1 & v_k^2 & \dots & v_k^p \end{bmatrix} + \epsilon_i$$

$$Y_i \approx X_i\beta + \begin{bmatrix} u_i^1 & u_i^2 & \dots & u_i^k \end{bmatrix} \begin{bmatrix} d_1^2 \times (v_1^1 & v_1^2 & \dots & v_1^p) \\ \vdots & \vdots & \vdots & \vdots \\ d_k^2 \times (v_k^1 & v_k^2 & \dots & v_k^p) \end{bmatrix} + \epsilon_i$$

Construction du modèle PPCA

$$Y_i \approx X_i\beta + \begin{bmatrix} u_i^1 & u_i^2 & \dots & u_i^k \end{bmatrix} \begin{bmatrix} d_1^2 \times (v_1^1 & v_1^2 & \dots & v_1^p) \\ \vdots & \vdots & \vdots & \vdots \\ d_k^2 \times (v_k^1 & v_k^2 & \dots & v_k^p) \end{bmatrix} + \epsilon_i$$

Quittons la géométrie pour introduire des variables aléatoires, celà sera plus joliment statistique. Pour une expérience de mesure i, il nous faut:

- D'abord un effet fixe $X_i\beta$, avec prior classique pour le paramètre β.
- Ensuite $U_i \approx \eta_i$: un vecteur ligne de longueur k qui repère sur k *features* résumantes les caractéristiques essentielles de l'expérience i. Compte tenu du caractère unitaire de U, on va spécifier η_i comme une grandeur latente gaussienne indépendante de dimension k, $\eta_i \sim N(O, I_k)$.
- Enfin $diag(d_k^2)V \approx \Lambda^T$: Λ est une matrice $k \times p$. Compte tenu du caractère unitaire de V, on sait que ses k colonnes sont de moins *lourdes* à cause du rangement des valeurs propres $d_1 \geq d_2 \dots \geq d_k$. Cette matrice Λ intervient de la même façon pour toutes les mesures $i = 1 : n$.
  Contrairement à η_i, ce n'est pas une variable latente, mais un *paramètre* (dont le prior reste à définir)!
- sans oublier le terme d'erreur classique $\epsilon_i \sim N(0, \sigma^2)$.

Finalement $\mathbf{Y}$ est un modèle normal multivarié avec indépendance entre les lignes : La ligne i est multivariée normale

Q'avons nous construit? un *simple* LMM(M)?

$$\underset{n\times p}{\mathbf{Y}} = \mathbf{X}\beta + \underset{n\times k}{\eta} \overset{(p\times k)^T}{\Lambda^T} + \underset{n\times p}{\epsilon}$$

$$\epsilon_{i,j} \sim N(0,1),\ \eta_{i,l} \sim N(0,1),\ \epsilon_{i,j} \perp \eta_{i,l}$$

pour $1 \leq i \leq n, 1 \leq j \leq p, 1 \leq l \leq k$

Soit en marginalisant:

$$\mathbf{Y_i} \sim \mathcal{N}_p\left(\mathbf{X_i}\beta, \Sigma\right)$$

avec $\underset{p\times p}{\Sigma} = \Lambda\mathbb{E}(\eta'\eta)\Lambda^T + diag(\sigma_j^2) = \underset{(p\times k)(k\times p)}{\Lambda\Lambda^T} + diag(\sigma_j^2)$

Reste à spécifier Λ, voir la big astuce de Batthacharya et Dunson

$$\Lambda^T \approx \begin{bmatrix} d_1^2 \times (v_1^1 & v_1^2 & \dots & v_1^p) \\ \vdots & \vdots & \vdots & \vdots \\ d_k^2 \times (v_k^1 & v_k^2 & \dots & v_k^p) \end{bmatrix}$$

Quelles lois a priori pour une ACP Bayesienne?

$$\mathbf{Y}_i = \mathbf{X}_i\beta + \eta_i\Lambda^T + \epsilon_\mathbf{i}$$

- **Variance** sur $\text{diag}(\sigma_j^2), 1 \leq j \leq p$ Inverse Gamma: Standard
- **Composantes Latentes** sur $\eta_i, 1 \leq i \leq n$: Standard$\mathcal{N}_p(0, I_p)$
- **Loading priors** sur la matrice $\underset{p\times k}{\Lambda}$ pour traduire en probabilité :
  $\lambda_{j,k} \approx d_l \times v_{j,l}$ avec $d_1 \geq d_2 \geq \ldots \geq d_k$
  – Imaginons $k \to \infty$: le prior sur Λ doit aider à forcer les colonnes de Λ à devenir de plus en plus *légères* avec leur rang, jusqu'à 0 pour la dimension inconnue théorique k.

The *multiplicative gamma process shrinkage prior* of Bhattacharya and Dunson (2011) allows for conjugate scheme and penalize high rank columns of Λ.

Bhattacharya, A. and Dunson, D. B.(2011), Sparse Bayesian Infinite Factor Models, Biometrika.

Le processus multiplicatif gamma comme prior

- Idée: Pénaliser de plus en plus les colonnes de rangs élevés de la matrice $p \times k$ Λ ;
  – Fixons pour $1 \leq j \leq p$ et $1 \leq h \leq q$, $\phi_{j,h} \overset{\text{ind}}{\sim} \mathcal{G}\text{amma}\left(\frac{3}{2}, \frac{3}{2}\right)$.
  – Pour $1 \leq h \leq q$, $\delta_h \overset{\text{ind}}{\sim} \mathcal{G}\text{amma}(\alpha, 1)$ de telle sorte que $\alpha > 1$ (donc en espérance une progression $\mathbb{E}[\delta_h] > 1$);
  – Le prior suivant répond aux exigences:

$$\Lambda_{j,h} | \phi_{j,h}, \delta_{1:h} \overset{\text{ind}}{\sim} \mathcal{N}\left(0, \phi_{j,h}^{-1} \prod_{\ell=1}^{h} \delta_\ell^{-1}\right).$$

- Quand le rang h augmente, les dernières colonnes de la matrice Λ tendent à se contracter vers 0 (leur moyenne a priori) car la precision de la colonne h augmente (en probabilité) comme $\prod_{\ell=1}^{h} \delta_\ell$.
- Reste la loi a priori pour α: Non informative, plus grande que 1.
- **Implementation** de l'inference bayésienne. MCMC: voir `Hmsc` le package `R` associé au livre d' Ovaskainen and Abrego (2020). Algorithme glouton. Facile à reprogrammer en Jags ou en Stan.

Ovaskainen, O. and Abrego, N. (2020), Joint Species Distribution Modelling: With Applications in R.

Ecriture du modèle en Jags : juste quelques lignes

```
modelString = "
model{
a[1] <- 3; a[2] <- 3; deminu <- 3/2; a_sigma <- 3; b_sigma <- 2;

#shrinkage prior
                delta[1] ~ dgamma(a[1],1)
for( h in 2:q){delta[h] ~ dgamma(a[2],1)}
for( h in 1:q){ tau[h] <- prod(delta[1:h])}
for( j in 1:p){ for( h in 1:q){ phi[j,h] ~ dgamma(deminu,deminu)
                        precilambda[j,h] <- phi[j,h]*tau[h]
                        lambda[j,h]~ dnorm(0,precilambda[j,h])}}
for(i in 1:n){for (h in 1:q){eta[i,h] ~ dnorm(0,1)}}
muY <- eta %*% t(lambda)
for(j in 1:p){ preciE[j] ~ dgamma(a_sigma,b_sigma)
              for (i in 1:n){Y[i,j] ~ dnorm(muY[i,j],preciE[j])}}
}"
```

Lancer le code Jags

```
library(rjags); library(ggmcmc); Y=M
data_for_JAGS <- list( Y=Y, n = dim(Y)[1],p = dim(Y)[2], q = 5)
n.adapt = 300; burnin = 300; n.iter = burnin * 3; thin = 1
jm <- jags.model(file = textConnection(modelString),
  data = data_for_JAGS,
  n.chains = 3 ,
  n.adapt = n.adapt)
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 300
   Unobserved stochastic nodes: 300
   Total graph size: 1021

Initializing model
```

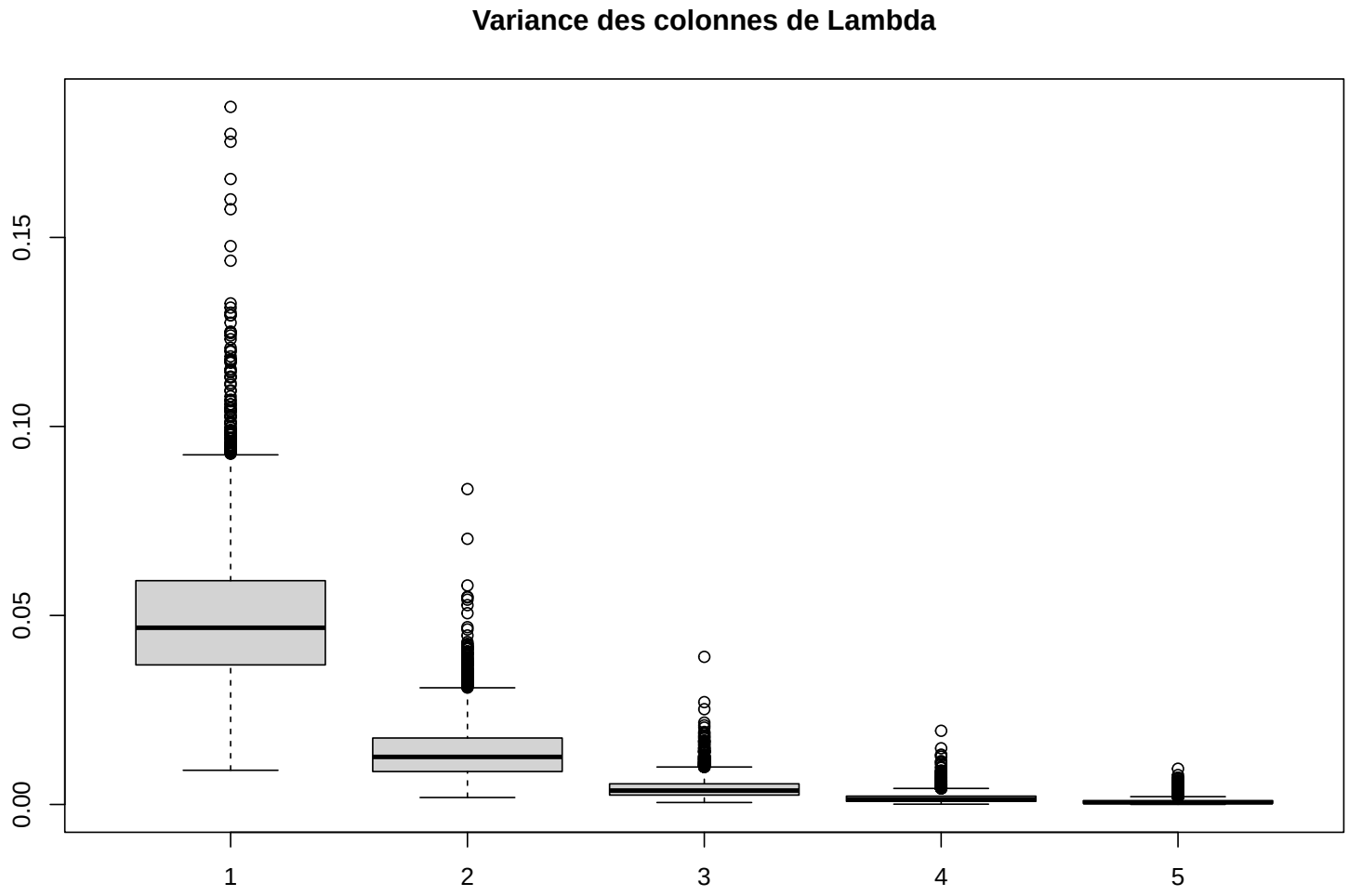
```
update(jm, burnin)
jsamples <- coda.samples(model = jm,
  variable.names = c("lambda","preciE", "eta","delta","phi","tau"),
  n.iter = n.iter,
  thin = thin)

result_jags <- ggs(jsamples)
```

Extraire les variables d'intérêt

```
library(tidyverse)
n_samples <- n.iter*3
Lambdas_jags <- filter(result_jags, str_detect(Parameter, "lambda")) %>
  group_by(Chain, Iteration) %>%
  group_map(function(x, g){
    pull(x, value) %>%
      matrix(nrow = ncol(Y), byrow = TRUE)
  }) %>%
  {.[1:n_samples]}
Lambdas_jags_var <- Lambdas_jags %>%
  sapply(function(x) apply(x, 2, var)) %>%
  t()
```

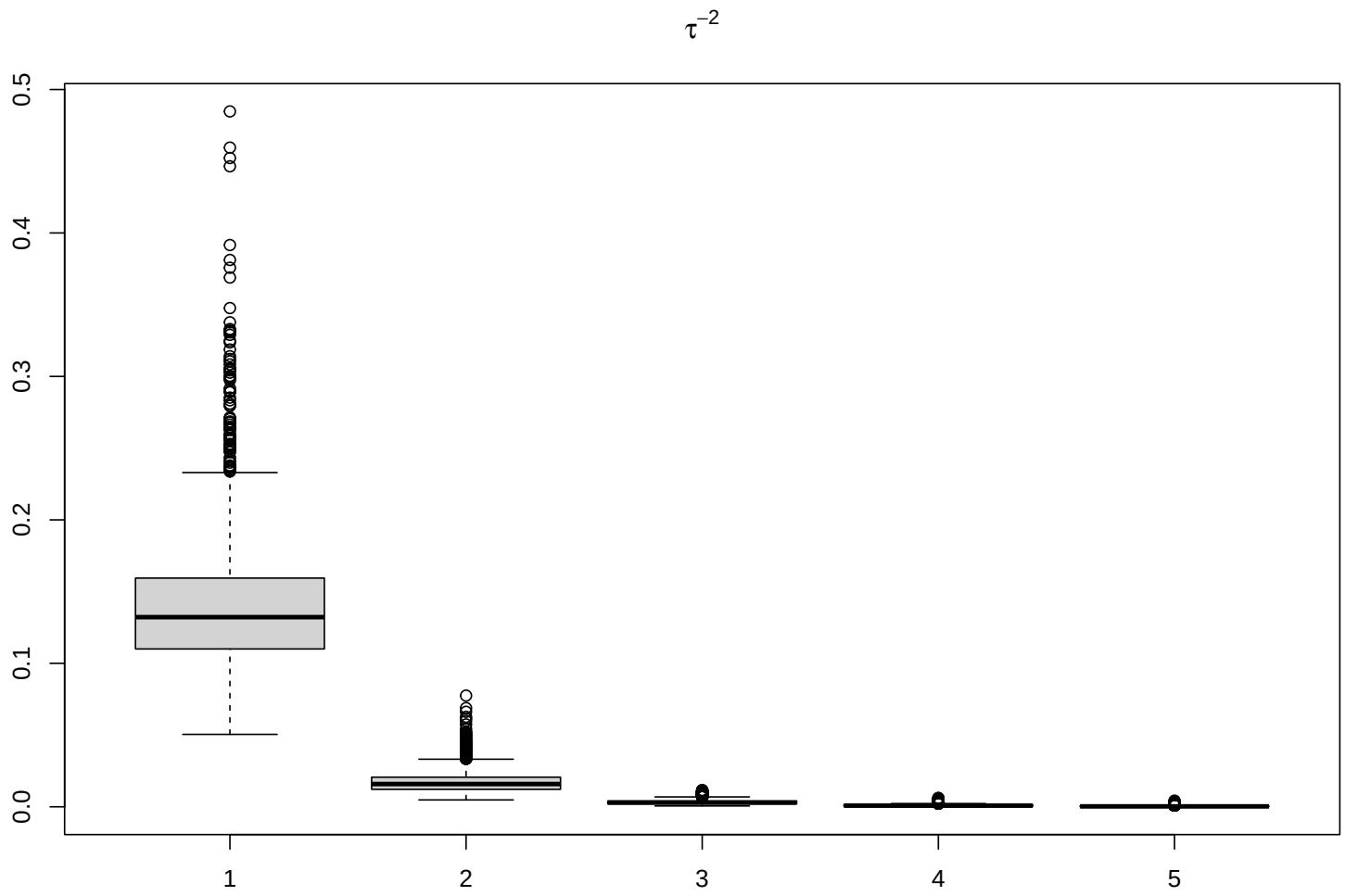
Représenter les sorties

```
Lambdas_jags_var %>% boxplot()
title(main = paste ("Variance des colonnes de", expression(Lambda)))
```



Représenter les sorties

```
filter(result_jags, str_detect(Parameter, "tau")) %>%
  group_by(Chain, Iteration) %>%
  group_map(function(x, g){pull(x, value) }) %>%
  {.[1:n_samples]} %>% sapply(function(x) 1/x) %>% t() %>% boxplot()
title(main =  expression(tau^-2))
```



Conclusions

Pratico-philosophique

- Quid de l'orthodoxie Bayesienne?
  – Est-ce vraiment une approche bayésienne? Pas d'élicitation, Pas de fonction de coût explicite, pas de décision formalisée
  – Oui mais utiliser le prior de Bhattacharya & Dunson's s'emboîte à la perfection dans le Lego land de l'inférence bayésienne de la modélisation hiérarchique et de l'inférence de type Gibbs.
  – A employer comme un module qui révèle automatiquement la dimension du modèle?
  – Chiquet, Mariadassou, and Robin (2018) proposent une autre approche pour construire Λ...

Les promesses pratiques de l'approche (Voir exposé de Pierre G.)

- Passage en variable latente pour des GLMs type $Z = \mathcal{P}(e^Y)$;
- Passage à l'échelle n ou p grands: recours au *Variational* Bayes pour une inférence a posteriori type champ moyen. (cf Hoffman et al. (2013)) On peut continuer à tirer parti du caractère modulaire et conjugué.
- Améliorer le cout computationnel grâce à l'optimisation fonctionnelle (cf Kingma and Welling (2014)).
- Codes (à vos risques et périls!) see `https://github.com/papayoun/VI_fo_Poisson_PCA`

Ouf!

Merci pour votre intérêt!

Shrinking prior + Variational Bayes = a game of LEGO bricks

Due to conditionnal independence the Evidence Lower Bound criterion can decomposed as

$$\begin{aligned}
\text{ELBO} =& \mathbb{E}_q \left[\log \left([Y|Z]\right)\right] \\
&+ \mathbb{E}_q \left[\log \left([Z|\eta, \Lambda, \Sigma, \beta]\right)\right] \\
&+ \mathbb{E}_q \left[\log \left([\beta]\right)\right] \\
&+ \mathbb{E}_q \left[\log \left([\Sigma]\right)\right] \\
&+ \mathbb{E}_q \left[\log \left([\eta]\right)\right] \\
&+ \mathbb{E}_q \left[\log \left([\Lambda|\delta, \phi]\right)\right] \\
&+ \mathbb{E}_q \left[\log \left([\phi]\right)\right] \\
&+ \mathbb{E}_q \left[\log \left([\delta]\right)\right] \\
&- \mathbb{E}_q \left[\log q(Z, \Lambda, \Sigma, \eta, \phi, \delta, \beta)\right]
\end{aligned}$$

$\Rightarrow$ Loop over local coordinate optimizations with latent $\theta \in Z, \Lambda, \Sigma, \eta, \phi, \delta, \beta$:

$$\mathrm{J}(\theta) = \left(\mathbb{E}_q \left[\log \left([\theta|\ \not{\theta}]\right)\right] - \mathbb{E}_q \left[\log q(\theta)\right]\right)$$

$\Rightarrow$ Coordinate Ascent Variational Inference algorithm

Explicit conjugate results for most components

As an example consider updating $\phi_{j,h}$

The terms implying $\phi_{j,h}$ are the following:

$$\left(\frac{\nu}{2}+\frac{1}{2}-1\right)\log\phi_{j,h}-\left(\frac{\nu}{2}+0.5\times\Lambda_{j,h}^{2}\prod_{\ell=1}^{h}\delta_{\ell}\right)\phi_{j,h}.$$

Therefore, the updates of the Gamma distribution parameters are given by:

$$\begin{aligned}A^{\phi_{j,h}} &= \frac{\nu}{2}+\frac{1}{2}\\ B^{\phi_{j,h}} &= \frac{\nu}{2}+\frac{1}{2}\left(\left(M^{\Lambda_{j,h}}\right)^{2}+V_{h,h}^{\Lambda_{j}}\right)\prod_{\ell=1}^{h}\frac{A^{\delta_{\ell}}}{B^{\delta_{\ell}}}.\end{aligned}$$

With the notable exception of $q(Z)$

No closed form expression for updating $Z \Rightarrow$ numerically maximising:

$$\mathbb{E}_{q_Z}\left[\log[Y|Z]\right] + \mathbb{E}_{q_Z}\left[\log[Z|\eta, \Lambda, \Sigma, \beta]\right] - \mathbb{E}_q\left[\log q_Z(Z)\right]$$

For CAVI algorithm, we take $q_Z(Z) = \prod_{i,j} q_Z(Z_{i,j})$ in the normal family.

Up to constant terms (with regards to $q(Z_i j)$), for each (i, j) maximise the partial ELBO function :

Denoting $\mathbb{E}_q(Z_{i,j}) = M$ and $\mathbb{V}ar_q(Z_{i,j}) = V$:

$$Y_{i,j}M - e^{M+\frac{V}{2}} - 0.5\frac{A^{\sigma_j}}{B^{\sigma_j}}M^2 - 0.5\frac{A^{\sigma_j}}{B^{\sigma_j}}V + M \times \frac{A^{\sigma_j}}{B^{\sigma_j}}\left(M^{\eta_i}M^{\Lambda_j} + X_i M^{\beta_j}\right) + \frac{\log|V|}{2}$$

Straightforward **Implementation** through $n \times p$ calls to the R *optim* subroutine

References

Bhattacharya, Anirban, and David B Dunson. 2011. "Sparse Bayesian Infinite Factor Models." *Biometrika*, 291–306.

Chiquet, Julien, Mahendra Mariadassou, and Stéphane Robin. 2018. "Variational Inference for Probabilistic Poisson PCA." *The Annals of Applied Statistics* 12 (4): 2674–98.

Hoffman, Matthew D, David M Blei, Chong Wang, and John Paisley. 2013. "Stochastic Variational Inference." *Journal of Machine Learning Research*.

Kingma, Diederik P, and Max Welling. 2014. "Stochastic Gradient VB and the Variational Auto-Encoder." In *Second International Conference on Learning Representations, ICLR*, 19:121.

Ovaskainen, Otso, and Nerea Abrego. 2020. *Joint Species Distribution Modelling: With Applications in R*. Cambridge University Press.