blockcluster, simerge and C++ with R

Serge Iovleff, Seydou Nourou Sylla

To cite this version:

Serge Iovleff, Seydou Nourou Sylla. blockcluster, simerge and C++ with R. Mixture Models: Theory and Applications, Jun 2018, Paris, France. hal-01884822

HAL Id: hal-01884822
https://hal.inria.fr/hal-01884822

Submitted on 1 Oct 2018

blockcluster, simerge and C++ with R

Serge Iovleff,
Equipe Projet Modal,
Université de Lille

Nourou Sylla
Équipe G4BBM, Institut Pasteur de Dakar
36, Avenue Pasteur B.P. 220 - DAKAR

Summary

Co-Clustering

"Aims to organize data-set into a set of homogeneous blocks by simultaneous clustering of individuals and variables."

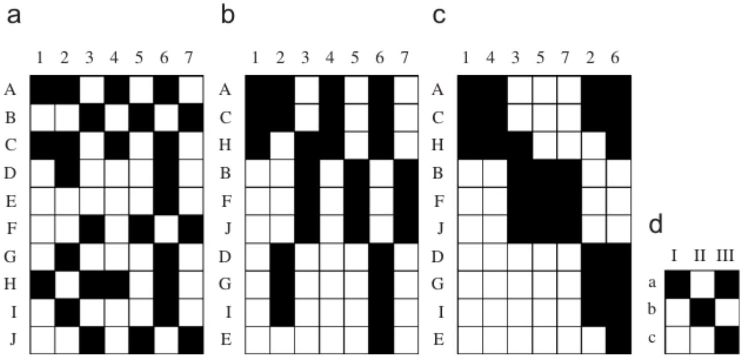


Figure: Binary data set (a), data reorganized by a partition on I (b), by partitions on I and J simultaneously (c) and summary matrix (d).

Model Based Approach

$\mathbf{x}$ is data set doubly indexed by a set I with n elements (individuals) and a set J with m elements (variables).
$\mathbf{z} = (z_{11}, \ldots, z_{ng})$ with $z_{ik} = 1$ if i belongs to cluster k and $z_{ik} = 0$ otherwise,
$\mathbf{w} = (w_{11}, \ldots, w_{md})$ with $w_{j\ell} = 1$ if j belongs to cluster ℓ and $w_{j\ell} = 0$ otherwise,

$$f(\mathbf{x}; \boldsymbol{\theta}) = \sum_{(\mathbf{z},\mathbf{w}) \in \mathcal{Z} \times \mathcal{W}} p(\mathbf{z}; \boldsymbol{\theta}) p(\mathbf{w}; \boldsymbol{\theta}) f(\mathbf{x} | \mathbf{z}, \mathbf{w}; \boldsymbol{\theta}) \qquad (1)$$

where $\mathcal{Z}$ and $\mathcal{W}$ denote the sets of all possible labelling $\mathbf{z}$ of I and $\mathbf{w}$ of J. There is $g^n \times d^m$ labelling possible.

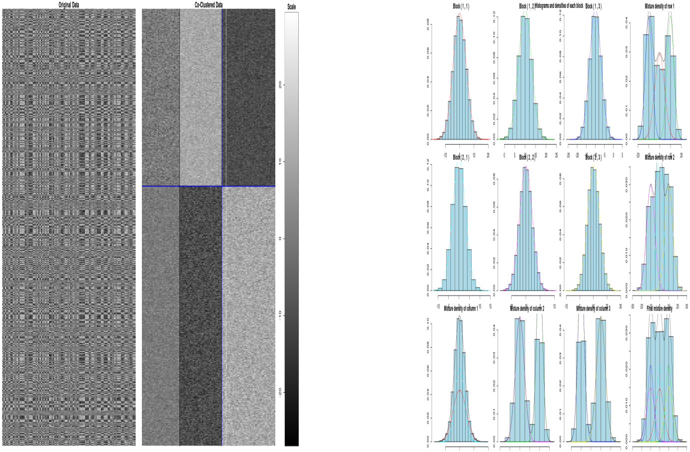
blockcluster: R Package For coclustering

- R interface to C++ library coclust (using STK++ in background),
- Simple and Robust API,
- Extend four basic functions "Plot","Summary","Show","Print",
- Implements “intelligent” estimation strategy.

Example

```
data(gaussiandata)
out<-coclusterGaussian(gaussiandata,model="
    pi_rho_sigma2kl",nbcocluster=c(2,3))
plot(out)
plot(out, type="distribution")
```

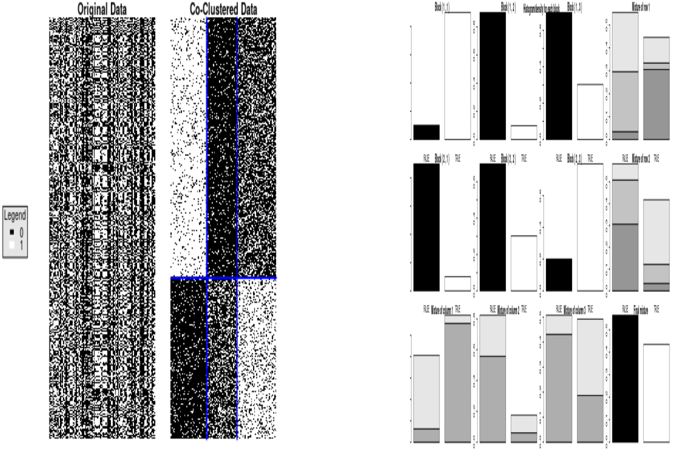
Example : Gaussian distribution



(a) (b)

Figure: Simulated and co-clustered data (a), Data block-distributions (b)

Example : Binary distribution

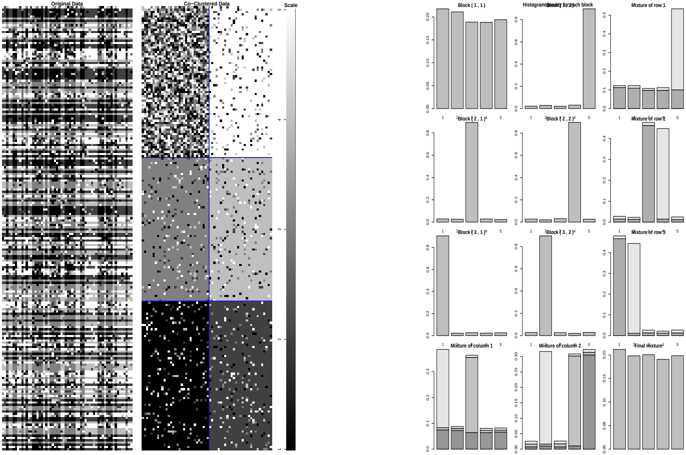


(a) (b)

Figure: Simulated and co-clustered data (a), Data block-distributions (b)

Example : Categorical distribution

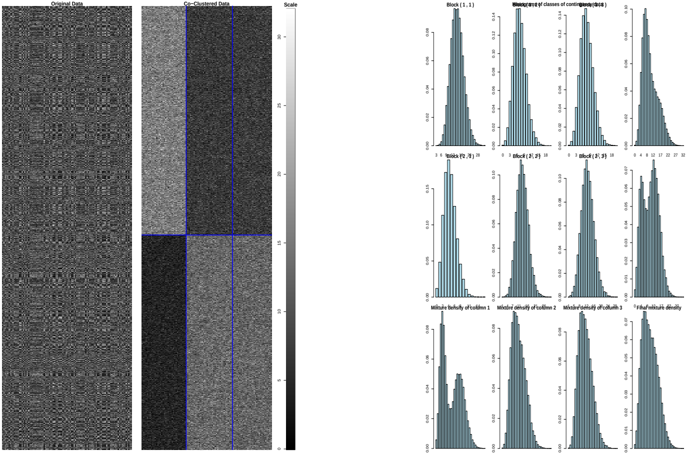
s



(a) (b)

Figure: Simulated and co-clustered data (a), Data block-distributions (b)

Example : Poisson distribution



(a)

(b)

Figure: Simulated and co-clustered data (a), Data block-distributions (b)

Development history

- First versions developed during ADT coclust (October 2011-October 2013). Implement binary, Poisson, Gaussian models; BEM and BCEM algorithms.
- Release 3.0 in 2014 add:
  1. Support for categorical data,
  2. Add Bayesian inference estimation algorithms,
  3. But stay unstable in certain situations (crashes..).
- Release 4.0 in November/December 2015 :
  1. Use STK++ as background library (code became cleaner and more compact).
  2. Fix (a lot of) crashes issues,
- Enhancement in release 4.2 in November/December 2016 (ADT Massicc)
  1. Adding selection criteria,
  2. Adding Gibbs estimation algorithms.

Summary

blockcluster package

simerge: Block clustering of binary data with Gaussian co-variables

C++ Programming with R: The `simerge` package

Preliminary Results

References

Simerge : Statistical Inference for the Management of Extreme Risks, Genetics and Global Epidemiology `http://mistis.inrialpes.fr/simerge/index.html`

SIMERGE is a LIRIMA project-team started in January 2015. It includes

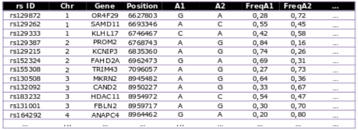
- Mistis (Inria Grenoble - Rhône-Alpes, France)
- LERSTAD (Laboratoire d'Etudes et de Recherches en Statistiques et Développement, Université Gaston Berger, Sénégal)
- IRD (Institut de Recherche pour le Développement, équipe G4BBM, Dakar, Sénégal)
- LEM (Lille Economie et Management, Université Lille 2)
- Modal (Inria Lille Nord-Europe)

The Associate team is built on two research themes:

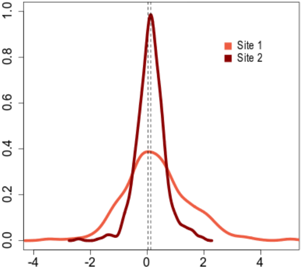
1. Spatial extremes, application to management of extreme risks
2. Classification, application to genetics and global epidemiology

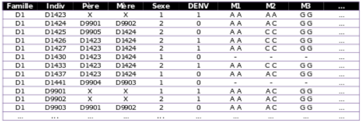
Challenge

Build statistical models in order to test association between diseases and human host genetics in a context of genome-wide screening.



rs ID	Chr	Gene	Position	A1	A2	FreqA1	FreqA2	...
rs129872	1	OR4F29	6627803	G	A	0,28	0,72	...
rs129262	1	SAMD11	6693346	A	C	0,55	0,45	...
rs129333	1	KLHL17	6746467	C	A	0,42	0,58	...
rs129387	2	PROM2	6768743	A	G	0,84	0,16	...
rs129215	2	KCNIP3	6835360	A	G	0,74	0,26	...
rs152324	2	FAHD2A	6962473	G	A	0,69	0,31	...
rs155308	2	TRIM43	7096057	A	G	0,27	0,73	...
rs130508	3	MKRN2	8945482	A	G	0,64	0,36	...
rs132092	3	CAND2	8950227	A	G	0,33	0,67	...
rs183232	3	HDAC11	8954972	A	C	0,54	0,47	...
rs131001	3	FBLN2	8959717	A	G	0,30	0,70	...
rs164292	4	ANAPC4	8964462	G	A	0,20	0,80	...
...	...	...	...	...	...	...	...	...





Famille	Indiv	Père	Mère	Sexe	DENV	M1	M2	M3	...
D1	D1423	X	X	1	1	A A	A A	G G	...
D1	D1424	D9901	D9902	2	0	A A	A C	G G	...
D1	D1425	D9905	D1424	2	0	A A	C C	G G	...
D1	D1426	D1423	D1424	2	1	A A	C C	G G	...
D1	D1427	D1423	D1424	2	1	A A	C C	G G	...
D1	D1430	D1423	D1424	1	0	-	-	-	...
D1	D1433	D1423	D1424	2	1	A A	C C	G G	...
D1	D1437	D1423	D1424	1	0	A A	A C	G G	...
D1	D1441	D9904	D9903	1	0	-	-	-	...
D1	D9901	X	X	1	1	A A	A C	G G	...
D1	D9902	X	X	2	1	A A	A C	G G	...
D1	D9903	D9901	D9902	2	0	A A	A C	G G	...
...	...	...	...	...	...	...	...	...	...

Figure: Genotypes on 719,656 SNPs (Single Nucleotide Polymorphism) typed on 481 individuals in Senegal, in rural area where malaria and arboviral diseases are endemic. 1 malaria quantitative phenotype on two sites: the individual effect on the risk of having malaria attack (iPFA).

Statistical Model

"Pour que `blockcluster` mette en évidence une cause génétique à l'iPFA, il faudrait que les populations aient été exposées à la maladie pendant plusieurs millénaires"(Cheick Loucoubar, head of G4BBM)

x is a binary data-set.
y is a data-set (co-variables) of $\mathbb{R}^p$ indexed by I.
Classical block model formulation for binary data is extended

$$f(\mathbf{x},\mathbf{y};\boldsymbol{\theta}) = \sum_{(\mathbf{z},\mathbf{w})\in\mathcal{Z}\times\mathcal{W}} p(\mathbf{z};\boldsymbol{\theta})p(\mathbf{w};\boldsymbol{\theta})f(\mathbf{x}|\mathbf{y},\mathbf{z},\mathbf{w};\boldsymbol{\theta})f(\mathbf{y}|\mathbf{z};\boldsymbol{\theta}). \qquad (2)$$

Dependency between x_{ij} and $\mathbf{y}_i$ modeled by canonical link for binary response data

$$f(x_{ij}|\mathbf{y}_i,\boldsymbol{\beta}_{z_i w_j}) = \text{logis}(\boldsymbol{\beta}_{z_i w_j}^T\mathbf{y}_i)^{x_{ij}} \left(1-\text{logis}(\boldsymbol{\beta}_{z_i w_j}^T\mathbf{y}_i)\right)^{1-x_{ij}} \qquad (3)$$

$$f(\mathbf{y}|\mathbf{z};\boldsymbol{\theta}) = \prod_i \phi(\mathbf{y}_i;\boldsymbol{\mu}_{z_i},\boldsymbol{\Sigma}_{z_i}) \text{ with } \phi \text{ multivariate Gaussian density.}$$

Estimation

EM algorithm not feasible as quantity $e_{ikj\ell} = P(z_{ik}w_{j\ell} = 1|\mathbf{x}, \mathbf{y}, \boldsymbol{\theta})$ is not computable.
Take $q(\mathbf{z}, \mathbf{w}) = t(\mathbf{z})r(\mathbf{w}) = \mathbf{t} \times \mathbf{r}$ with $\mathbf{t}$ and $\mathbf{r}$ matrices of sizes (n, g) and (m, d), then

$$l(\boldsymbol{\theta}) = \tilde{F}_C(\mathbf{t}, \mathbf{r}, \boldsymbol{\theta}) + KL(q(\mathbf{z}, \mathbf{w}) \parallel p(\mathbf{z}, \mathbf{w}|\mathbf{x}, \mathbf{y}, \boldsymbol{\theta})) \tag{4}$$

with $KL(q \parallel p)$ denoting the *Kullback-Liebler* divergence and $\tilde{F}_C$ denoting the *Free Energy* or *Fuzzy Criterion*

$$\begin{aligned} \tilde{F}_C(\mathbf{t}, \mathbf{r}, \boldsymbol{\theta}) = & \sum_k t_{.k} \log \pi_k + \sum_\ell r_{.\ell} \log \rho_l \\ & + \sum_{i,j,k,\ell} t_{ik} r_{j\ell} \log f(x_{ij}, \mathbf{y}_i; \boldsymbol{\theta}_{k\ell}) \\ & + H(\mathbf{t}) + H(\mathbf{r}) \end{aligned} \tag{5}$$

and $H(\mathbf{t})$, $H(\mathbf{r})$ denoting the entropy of $\mathbf{t}$ and $\mathbf{r}$.
Maximization of likelihood $l(\boldsymbol{\theta})$ is replaced by the following maximization

$$\operatorname*{argmax}_{\mathbf{t},\mathbf{r},\boldsymbol{\theta}} \tilde{F}_C(\mathbf{t}, \mathbf{r}, \boldsymbol{\theta}).$$

BEM algorithm

Initialization Set $\mathbf{t}^{(0)}, \mathbf{r}^{(0)}$ and $\boldsymbol{\theta}^{(0)} = (\boldsymbol{\pi}^{(0)}, \boldsymbol{\rho}^{(0)}, \boldsymbol{\beta}^{(0)}, \boldsymbol{\mu}^{(0)}, \boldsymbol{\Sigma}^{(0)})$.

(a) Row-EStep Compute $\mathbf{t}^{(c+1)}$ using formula

$$t_{ik}^{(c+1)} = \frac{\pi_k^{(c)} \prod_{jl} \left(f(x_{ij}|\mathbf{y}_i; \boldsymbol{\beta}_{kl}^{(c)}) \phi(\mathbf{y}_i; \boldsymbol{\mu}_k^{(c)}, \boldsymbol{\Sigma}_k^{(c)}) \right)^{r_{jl}^{(c)}}}{\sum_k \pi_k^{(c)} \prod_{jl} \left(f(x_{ij}|\mathbf{y}_i; \boldsymbol{\beta}_{kl}^{(c)}) \phi(\mathbf{y}_i; \boldsymbol{\mu}_k^{(c)}, \boldsymbol{\Sigma}_k^{(c)}) \right)^{r_{jl}^{(c)}}}.$$

(b) Row-MStep Compute $\boldsymbol{\pi}^{(c+1)}$, $\boldsymbol{\mu}^{(c+1)}$, $\boldsymbol{\Sigma}^{(c+1)}$ and estimate $\boldsymbol{\beta}^{(c+1/2)}$.

(c) Col-EStep Compute $\mathbf{r}^{(c+1)}$ using formula

$$r_{jl}^{(c+1)} = \frac{\rho_l^{(c)} \prod_{ik} f(x_{ij}|\mathbf{y}_i; \boldsymbol{\beta}_{kl}^{(c+1/2)})^{t_{ik}^{(c+1)}}}{\sum_l \rho_l^{(c)} \prod_{ik} f(x_{ij}|\mathbf{y}_i; \boldsymbol{\beta}_{kl}^{(c+1/2)})^{t_{ik}^{(c+1)}}}.$$

(d) Col-MStep Compute $\boldsymbol{\rho}^{(c+1)}$ and estimate $\boldsymbol{\beta}^{(c+1)}$.

Iterate Iterate **(a)-(b)-(c)-(d)** until convergence.

Measuring contribution of a variable

m_l denotes the number of columns with label l, i.e
$m_l = \#\{w_{jl} = 1,\ j = 1, \ldots m\}$ and for a row i fixed let m_{il} denotes the number of elements such that $w_{jl} = 1$ and $x_{ij} = 1$, i.e.
$m_{il} = \#\{w_{jl} x_{ij} = 1,\ j = 1, \ldots m\}$. The posterior probability of the co-variable **y** is

$$f(\mathbf{y}|\mathbf{x}, \mathbf{z}, \mathbf{w}, \boldsymbol{\theta}) \propto \prod_{i=1}^{n} \pi_{z_i} \phi(\mathbf{y}_i; \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}) \prod_{l=1}^{d} \rho_l^{m_l} \frac{e^{m_{il} \mathbf{y}_i^T \boldsymbol{\beta}_{z_i l}}}{\left(1 + e^{\mathbf{y}_i^T \boldsymbol{\beta}_{z_i l}}\right)^{m_l}} \qquad (6)$$

Taking log, contribution of the jth variable is computed as

$$I(j) = \log \rho_{w_j} + \sum_{i=1}^{n} \left(x_{ij} \mathbf{y}_i^T \boldsymbol{\beta}_{z_i w_j} - \log(1 + \exp(\mathbf{y}_i^T . \boldsymbol{\beta}_{z_i w_j})) \right). \qquad (7)$$

using MAP estimator for **z** and **w**.

Summary

blockcluster package

simerge: Block clustering of binary data with Gaussian co-variables

C++ Programming with R: The `simerge` package

Preliminary Results

References

Extreme Programming (XP)[1]

Extreme Programming is a discipline of software development based on values of simplicity, communication, feedback, courage, and respect.

- **Simple Design**: XP teams build software to a simple but always adequate design. They start simple, and through programmer testing and design improvement, they keep it that way.
- **Pair Programming**: All production software in XP is built by two programmers, sitting side by side, at the same machine.
- **Test-Driven Development**: XP is obsessed with feedback, and in software development, good feedback requires good testing.
- **Design Improvement (Refactoring)**: XP focuses on delivering business value in every iteration. To accomplish this over the course of the whole project, the software must be well-designed.
- **Coding Standard**: XP teams follow a common coding standard, so that all the code in the system looks as if it was written by a single – very competent – individual.

---

[1] `https://ronjeffries.com/xprog/what-is-extreme-programming/`

Extreme Programming (XP)[1]

Extreme Programming is a discipline of software development based on values of simplicity, communication, feedback, courage, and respect.

- **Simple Design**: XP teams build software to a simple but always adequate design. They start simple, and through programmer testing and design improvement, they keep it that way.
- **Pair Programming**: All production software in XP is built by two programmers, sitting side by side, at the same machine.
- **Test-Driven Development**: XP is obsessed with feedback, and in software development, good feedback requires good testing.
- **Design Improvement (Refactoring)**: XP focuses on delivering business value in every iteration. To accomplish this over the course of the whole project, the software must be well-designed.
- **Coding Standard**: XP teams follow a common coding standard, so that all the code in the system looks as if it was written by a single – very competent – individual.

---

[1] `https://ronjeffries.com/xprog/what-is-extreme-programming/`

Extreme Programming (XP)[1]

Extreme Programming is a discipline of software development based on values of simplicity, communication, feedback, courage, and respect.

- **Simple Design**: XP teams build software to a simple but always adequate design. They start simple, and through programmer testing and design improvement, they keep it that way.
- **Pair Programming**: All production software in XP is built by two programmers, sitting side by side, at the same machine.
- **Test-Driven Development**: XP is obsessed with feedback, and in software development, good feedback requires good testing.
- **Design Improvement (Refactoring)**: XP focuses on delivering business value in every iteration. To accomplish this over the course of the whole project, the software must be well-designed.
- **Coding Standard**: XP teams follow a common coding standard, so that all the code in the system looks as if it was written by a single – very competent – individual.

---

[1] `https://ronjeffries.com/xprog/what-is-extreme-programming/`

Design and Coding Standard

Use S4 class for R side and a mirror C++ class

```
setClass(
  Class = "CoClusterBinary",
  representation = representation(
    # y part
    yid       = "matrix",  # covariables
    mukd      = "matrix",  # means of yid
    sigmakd   = "matrix",  # standard deviations
    isCoMixture = "logical",# yid is a mixture ?
    # x part
    xij          = "matrix",
#....
# Constructor of the S4 class
setMethod(
  f="initialize",
  signature=c("CoClusterBinary"),
  definition=function(.Object, x, y, nbcocluster,
      isCoMixture)
```

Design and Coding Standard

Use S4 class for R side and a mirror C++ class

```
class CoClusterBinaryModel: public STK::IRunnerBase
{
  public:
    // constructor of the C++ class
    CoClusterBinaryModel(Rcpp::S4 s4Model);
    //....
    STK::RMatrix<double> yid_;
    STK::RMatrix<double> mukd_;
    STK::RMatrix<double> sdkd_;
    bool isCoMixture_;
    STK::RMatrix<double> xij_;
```

Design and Coding Standard

Use S4 class for R side and a mirror C++ class

C++ constructor get R structure and wrap them as STK++ arrays

```
# Constructor of the S4 class
setMethod(
  f="initialize",
  signature=c("CoClusterBinary"),
  definition=function(.Object, x, y, nbcocluster,
      isCoMixture)
```

```
CoClusterBinaryModel::CoClusterBinaryModel( Rcpp::S4
    s4Model):
  //.....
  , yid_((SEXP)s4Model.slot("yid"))
  , mukd_((SEXP)s4Model.slot("mukd"))
  , sdkd_((SEXP)s4Model.slot("sigmak     d"))
  , isCoMixture_(s4Model.slot("isCoMixture"))
  , xij_((SEXP)s4Model.slot("xij"))
  //.....
```

Exemple: Computation of the Fuzzy Criterion $\tilde{F}_C$

R side

```
setMethod(
  f="logLikelihood",
  signature = "CoClusterBinary",
  definition = function(object)
  {
    .Call("logLikelihood",object,package="simerge")
  }
)
```

Exemple: Computation of the Fuzzy Criterion $\tilde{F}_C$

C side

```
extern "C" SEXP logLikelihood( SEXP model)
{
  Rcpp::S4 s4model(model);
  CoClusterBinaryModel coclust(model);

  coclust.computeLogLikelihood();
  coclust.getValues(model);

  return model;
}
```

Exemple: Computation of the Fuzzy Criterion $\tilde{F}_C$.

$$\tilde{F}_C(\mathbf{t},\mathbf{r};\boldsymbol{\theta}) = \sum_k t_{.k} \log \pi_k + \sum_\ell r_{.\ell} \log \rho_l + H(\mathbf{t}) + H(\mathbf{r})$$
$$+ \sum_{i,j,k,\ell} t_{ik} r_{j\ell} (\log(1+\exp(\mathbf{y}_i^T \boldsymbol{\beta}_{kl})) + x_{ij}\mathbf{y}_i^T \boldsymbol{\beta}_{kl}) + \log(\phi(\mathbf{y}_i;\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k))$$

```
setMethod(
  f="entropy",
  signature = "CoClusterBinary",
  definition = function(object)
  {
    epsilon <- 1e-15
    tik <- object@tik
    rjl <- object@rjl
    object@rowEntropy <- -sum(tik*log(epsilon+tik))
    object@colEntropy <- -sum(rjl*log(epsilon+rjl))
    return(object)
  }
)
```

Exemple: Computation of the Fuzzy Criterion $\tilde{F}_C$.

$$\tilde{F}_C(\mathbf{t},\mathbf{r};\boldsymbol{\theta}) = \sum_k t_{.k} \log \pi_k + \sum_\ell r_{.\ell} \log \rho_l + H(\mathbf{t}) + H(\mathbf{r})$$
$$+ \sum_{i,j,k,\ell} t_{ik} r_{j\ell} (\log(1+\exp(\mathbf{y}_i^T \boldsymbol{\beta}_{kl})) + x_{ij}\mathbf{y}_i^T \boldsymbol{\beta}_{kl}) + \log(\phi(\mathbf{y}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k))$$

C++ side

```
rowEntropy_= -tik_.prod( (tik_+RealMin).log()).sum();
colEntropy_= -rjl_.prod( (rjl_+RealMin).log()).sum();
```

Exemple: Computation of the Fuzzy Criterion $\tilde{F}_C$.

$$\tilde{F}_C(\mathbf{t}, \mathbf{r}; \boldsymbol{\theta}) = \sum_k t_{.k} \log \pi_k + \sum_\ell r_{.\ell} \log \rho_l + H(\mathbf{t}) + H(\mathbf{r})$$
$$+ \sum_{i,j,k,\ell} t_{ik} r_{j\ell} (\log(1 + \exp(\mathbf{y}_i^T \boldsymbol{\beta}_{kl})) + x_{ij} \mathbf{y}_i^T \boldsymbol{\beta}_{kl}) + \log(\phi(\mathbf{y}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k))$$

```
for(k in 1:K)
{
  for(l in 1:L)
  {
    object@likelihoodkl[k,l] =
     ( (tik_[,k] * yid%*%betakld[k,l,]) %*% xij_
     + crossprod(tik[,k],plogis(yid_%*%betakld[k,l
          ,],0,1,F,T))
     ) %*% rjl[,l];
    }
  }
```

Exemple: Computation of the Fuzzy Criterion $\tilde{F}_C$.

$$\tilde{F}_C(\mathbf{t},\mathbf{r};\boldsymbol{\theta}) = \sum_k t_{.k}\log\pi_k + \sum_\ell r_{.\ell}\log\rho_l + H(\mathbf{t}) + H(\mathbf{r})$$

$$+ \sum_{i,j,k,\ell} t_{ik} r_{j\ell}(\log(1+\exp(\mathbf{y}_i^T\boldsymbol{\beta}_{kl})) + x_{ij}\mathbf{y}_i^T\boldsymbol{\beta}_{kl}) + \log(\phi(\mathbf{y}_i;\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k))$$

```
for(int k=0; k<K_; ++k)
{
  for(int l=0; l<L_; ++l)
  {
    likelihoodkl_(k,l)
     = ( tik_.col(k).prod( yid_*betakld_(k,l)).
         transpose() * xij_
       + tik_.col(k).dot( (yid_*betakld_(k,l)).lcdfc(
           logis_) )
       ) * rjl_.col(l);
    }
  }
```

Exemple: Computation of the Fuzzy Criterion $\tilde{F}_C$.

$$\tilde{F}_C(\mathbf{t},\mathbf{r};\boldsymbol{\theta}) = \sum_k t_{.k}\log\pi_k + \sum_\ell r_{.\ell}\log\rho_l + H(\mathbf{t}) + H(\mathbf{r}) + \sum_{i,j,k,\ell} t_{ik}r_{j\ell}(\log(1+\exp(\mathbf{y}_i^T\boldsymbol{\beta}_{kl})) + x_{ij}\mathbf{y}_i^T\boldsymbol{\beta}_{kl}) + \log(\phi(\mathbf{y}_i;\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k))$$

```
gaussianLogLikelihood_= computeGaussianLogLikelihood()
    ;
fuzzyLogLikelihood_= likelihoodkl_.sum()
                  + tk_.dot(pik_.log())
                  + rl_.dot(rhol_.log())
                  + gaussianLogLikelihood_;
fuzzyCriterion_= fuzzyLogLikelihood_
               + rowEntropy_ + colEntropy_;
```

Summary

Data set

$n = 444$ individuals and $m = 515721$ SNPs conserved.

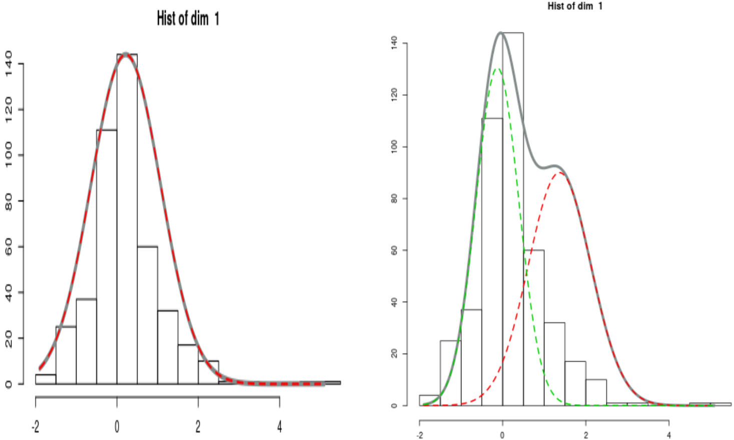


Figure: Histogram of the iPFA variable and fitted Gaussian mixture models obtained with MixAll package

Model selection

ICL BIC-like approximations leads to the following $\mathrm{BIC}(g, d)$

$$-2 \max_{\boldsymbol{\theta}} \log f(\mathbf{x}, \mathbf{y}; \boldsymbol{\theta}) + (g-1) \log n + \lambda \log n + (d-1) \log m + gd(p+1) \log(mn)$$

with λ the number of parameters of the **y** distribution.

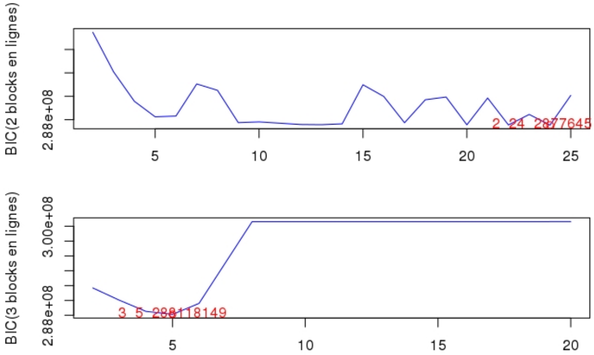
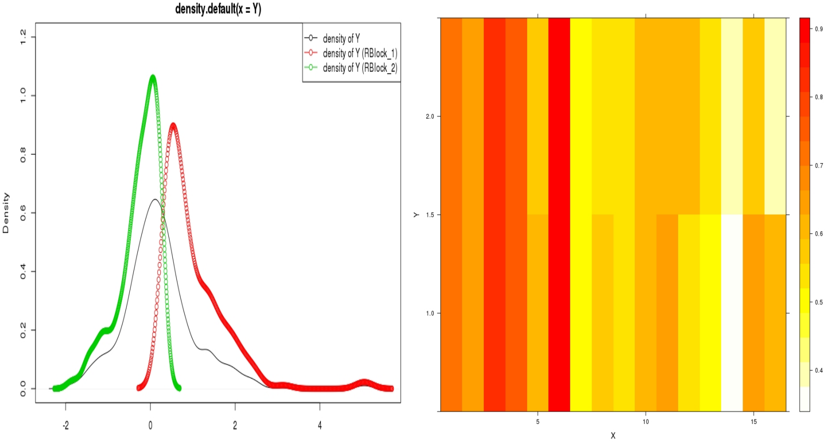


Figure: Choosing the number of blocks (Note: implemented criteria was **wrong**)

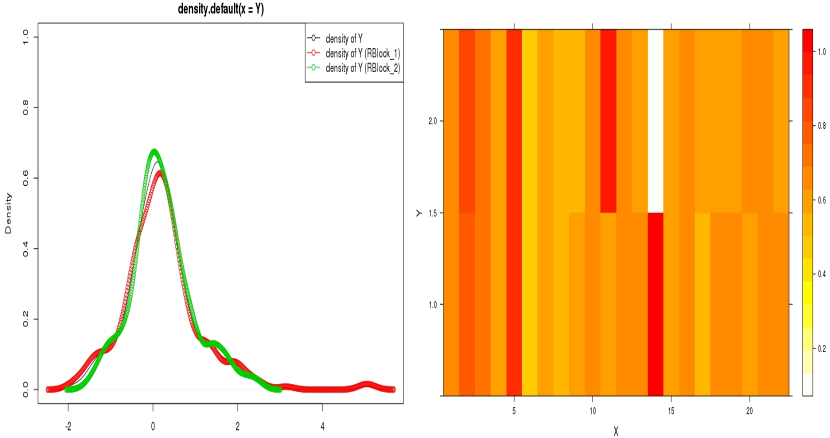
Results with $(g, d) = (2, 22)$ and $\mathbf{y}$ Gaussian mixture



(a) (b)

Figure: iPFA density (a), Proportion of mutation (b), BIC = 290551317

Results with $(g, d) = (2, 22)$ and $\mathbf{y}$ Gaussian rv



(a) (b)

Figure: iPFA density (a), Proportion of mutation (b), BIC = 287770996

Influence Measure

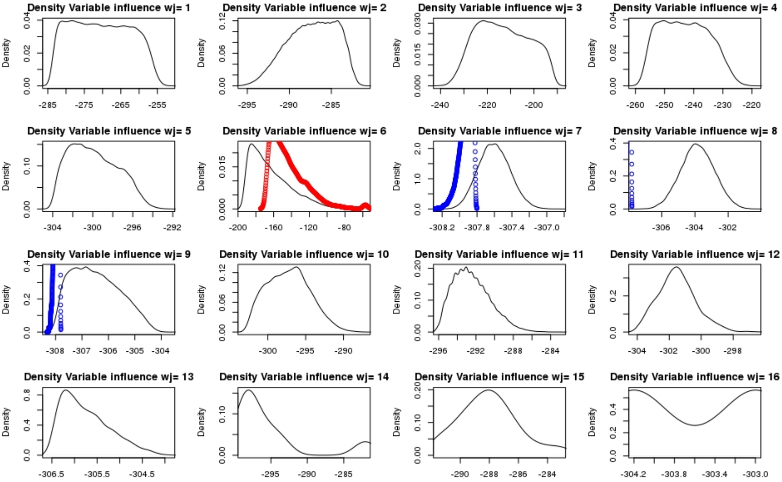


Figure: Repartition of the influence in clusters (by columns)

Summary

blockcluster package

simerge: Block clustering of binary data with Gaussian co-variables

C++ Programming with R: The `simerge` package

Preliminary Results

References

Merci à la G4BBM team

Cheikh LOUCOUBAR – **Biomathematician**

PhD in Statistical Genetics
Head of the Group
Dakar University / Paris 5

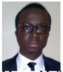
Maryam DIARRA – **Biomathematician**

PhD in Applied Mathematics
Saint Louis University (UGB)

Mamadou DIOP – **Computer Scientist**

Bioinformatician
Master in Computer Science
Saint Louis University (UGB)

Dame SY – **Data Manager**



DTS in Computer Science

Mame Malick DIENG – **Computer Scientist**

Master in Computer Science
Saint Louis University (UGB)

Seydou Nourou SYLLA – **Biomathematician**

PhD in Applied Mathematics
Saint Louis University (UGB

Amadou DIALLO – **Biomathematician**

Bachelor in Mathematics
Minot State University, USA

Mareme S. THIAM – **Master Fellow in Mathematics**

M2 Mathematics – Big Data
AIMS

Aboubacry GAYE – **Master Fellow in Mathematics**

M2 Mathematics
Saint Louis University (UGB)

Main Activities

- Research on human host genetic diversity and implication in malaria phenotypes
- New grant application

Other Activities

- Support IPD units in data management and analysis
- Teaching in collaborations with universities

Links

- http://www.pasteur.sn/recherche/biostatistique-bio-informatique-et-modelisation/
- https://cran.r-project.org/package=blockcluster
- https://cran.r-project.org/package=rtkore
- https://cran.r-project.org/package=MixAll
- http://www.stkpp.org
- https://modal.lille.inria.fr/wikimodal/doku.php