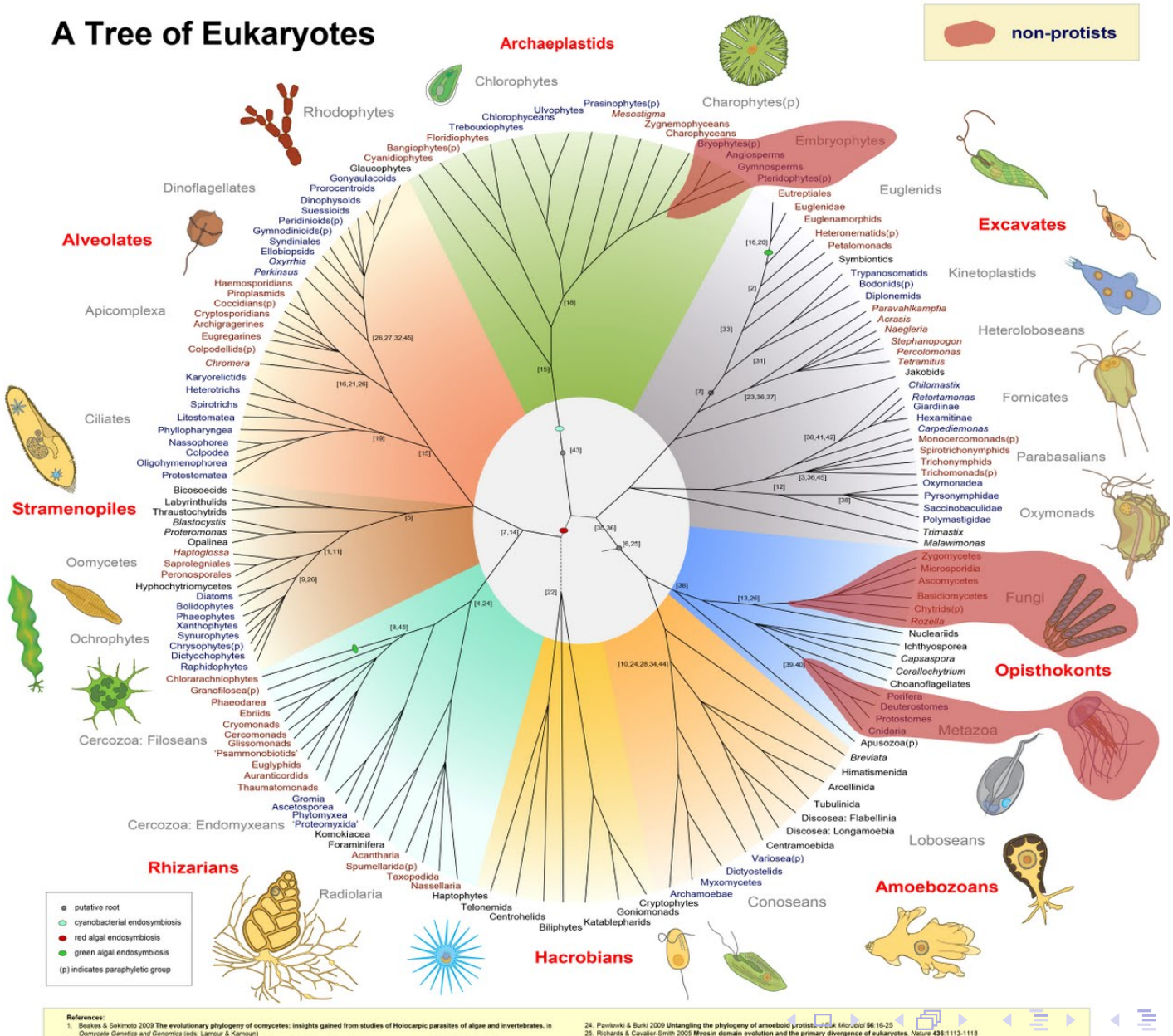
L'exploration de la diversité des protistes: l'apport du calcul intensif

J.-M. Frigerio, P. Chaumeil, F. Rué, S. Thérond, V. Louvet, O. Coulaud & A. Franc

INRA BioGeCo; INRIA Pleiade, Hiepacs, SED; CNRS IDRIS & GRICAD

JCAD, Lyon
26 octobre 2018

L'immense diversité des Eucaryotes !



References:
1. Beakes & Sekimoto 2009 The evolutionary phylogeny of oomycetes: insights gained from studies of Holocarpic parasites of algae and invertebrates. in Oomycete Genetics and Genomics (eds: Lamour & Kamoun)
24. Pawlowki & Burki 2009 Untangling the phylogeny of amoeboid protists. J Euk Microbiol 56:16-25
25. Richards & Cavalier-Smith 2005 Myosin domain evolution and the primary divergence of eukaryotes. Nature 436:1113-1118

1. Les données
2. Calcul des distances
3. Format et transfert des fichiers
4. Image euclidienne et réduction de la dimension
5. Projection aléatoire et librairie *Diodon*
6. Construction d'OTUs = classification non supervisée

Les données

Pour chaque échantillon

- un fichier de format dit `fasta`
- `ascii`
- de texte (`strings`)
- avec pour chaque séquence
  - un identifiant `>taratata`
  - une séquence `acgtgcatgcatgc...`

Tailles

- un fichier: entre 20 k et 150 k séquences
- taille $\leq$ 54 Mo
- typiquement 100 échantillons donc fichiers par projets

Distance entre séquences

Distance d'édition (Levenstein, 1965)

Etant données deux séquences s et s', le plus petit nombre d'opérations élémentaires parmi

- substitution
- insertion
- délétion

nécessaires pour transformer s en s'.

Algorithme de Needleman-Wunsch (NW, 1970)

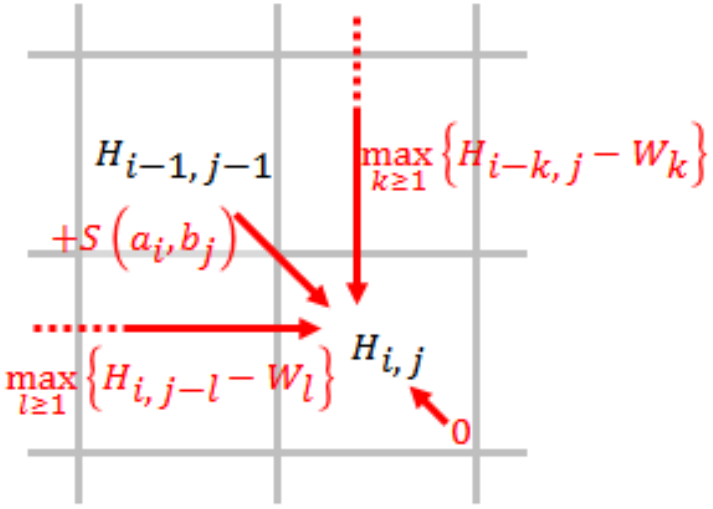
- algorithme de programmation dynamique

Algorithme de Smith-Waterman (SW, 1983)

- alignement local: une chaine peut être incluse dans une autre avec $d = 0$
- de complexité $\mathcal{O}(\ell\ell')$

Algorithme de SW

-	-	A	T	C	G	A	A
-	0	0	0	0	0	0	0
C	0	0	0	5	1	0	0
A	0	5	1	1	2	5	5
T	0	1	10	6	2	1	2
A	0	5	6	7	3	7	6
C	0	1	2	11	7	3	4



Parallélisation

Algorithm 1 pseudocode pour construire la matrice de distances

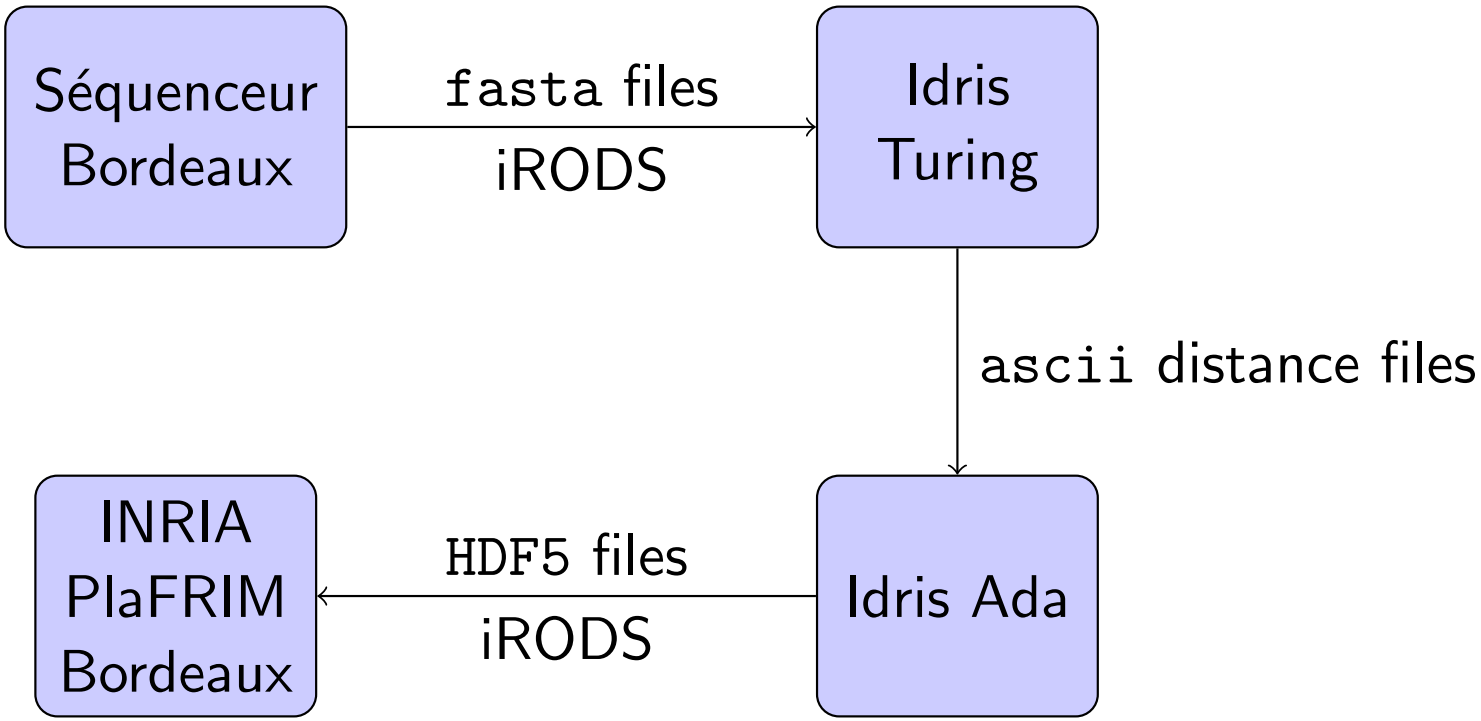
1: **for** $s \in [\![1, n-1]\!]$ **do**
2: **for** $s' \in [\![s+1, n]\!]$ **do**
3: `D[s,s']` $= \mathrm{SW}$(s,s')
4: **end for**
5: **end for**

Construit par blocs

$$\begin{pmatrix} B_{11} & \dots & B_{1n} \\ \vdots & & \vdots \\ B_{m1} & \dots & B_{mn} \end{pmatrix} \quad \& \quad \forall\, i,j, \quad B_{ij} \longrightarrow \mathrm{SW}(B_{ij})$$

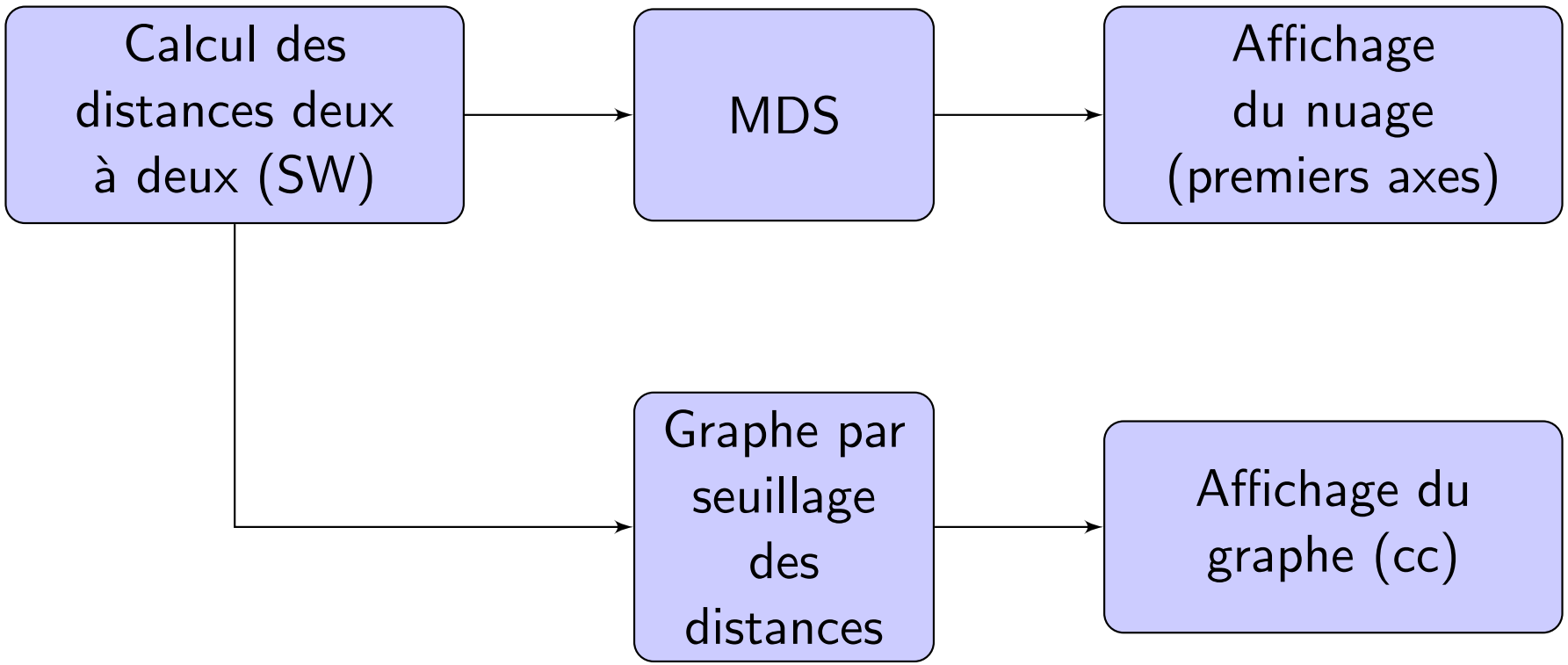
- programme en C, parallélisation en MPI
- passe parfaitement à l'échelle sur une machine hyperparallèle (Turing, Blue Gene Q, $16\,384 = 2^{14}$ cœurs)

Transfert pour post-traitement



Multidimensional Scaling

Tout commence par une distance ...



Concrètement ...

Multidimensional Scaling

Etant donné	$D_{ij} = d(i,j)$
	$r < n$
trouver	$\mathcal{X} = (x_1, \ldots, x_n)$
tel que	$x_i \in \mathbb{R}^r$
	$\|x_i - x_j\| \simeq d(i,j)$

Graphe

Etant donné	$D_{ij} = d(i,j)$
	$\alpha > 0$
Construire	$G = (V, E)$
tel que	$V = \{1, n\}$
	$i \sim j \Leftrightarrow d(i,j) \leq \alpha$

Algorithme de la MDS: cubique

Algorithm 2 pseudocode pour la MDS

1: **input:** D, r, `D[i,j]` $= d_{ij}$
2: build Gram matrix G from D: $(g_{ij} = \langle x_i, x_j \rangle)$
3: SVD of G: $G = U\Sigma V^{\mathrm{T}}$
4: coordinates: $X = U\Sigma^{1/2}$
5: **return** X

Projection aléatoire

Lemme de Johnson-Lindenstrauss (1984)

- Donnés: $n \in \mathbb{N},\ \epsilon \in [0,1], \quad k \geq \frac{8 \operatorname{Log} n}{\epsilon^2}$
- Alors, $\forall\, X = [x_1, \ldots, x_n], \quad x_i \in \mathbb{R}^n$
- $\exists \quad \mathbb{R}^n \xrightarrow{\quad f \quad} \mathbb{R}^k$
- tel que $\forall\, x, y \in X$

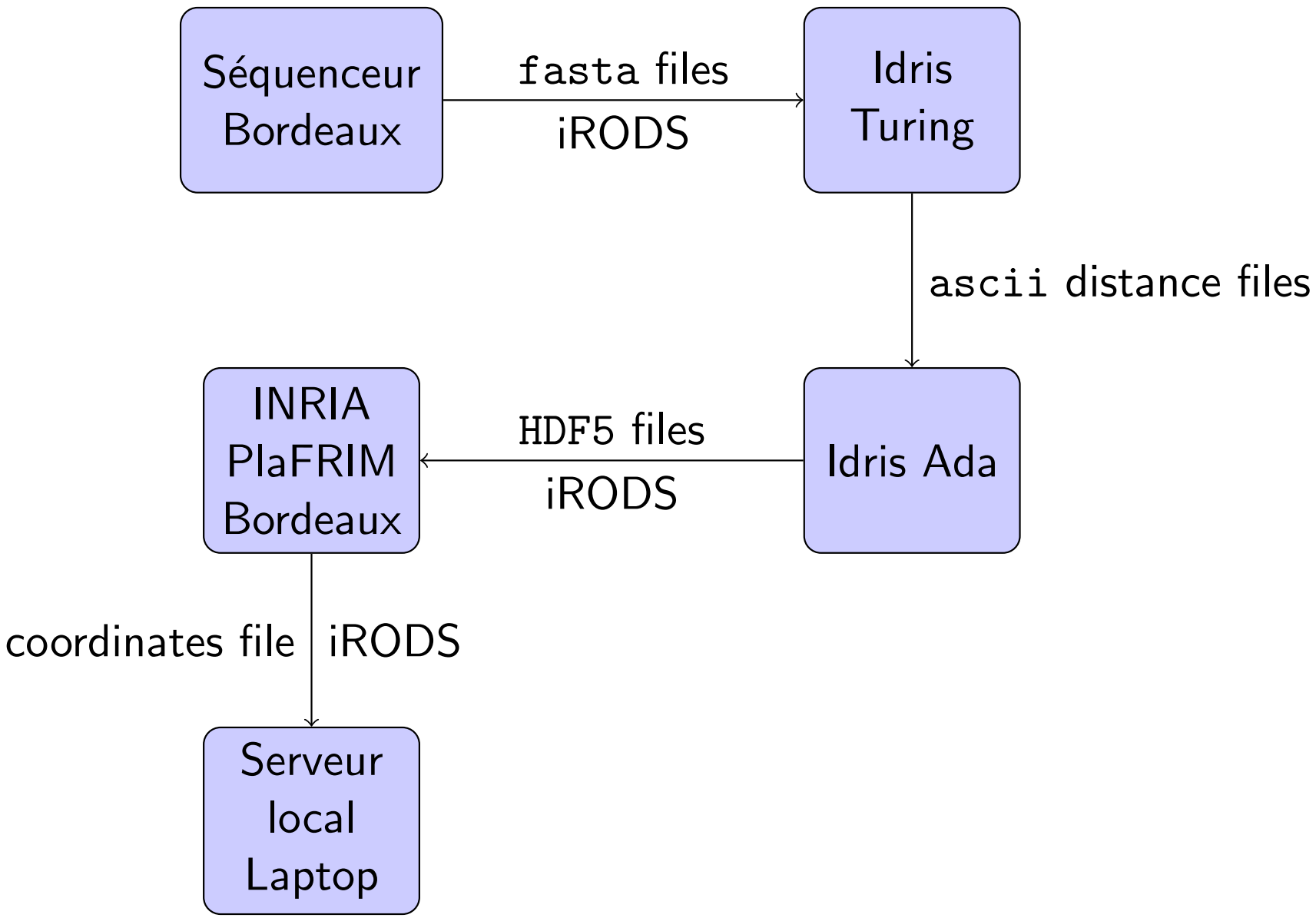
$$(1-\epsilon)\|x-y\|^2 \leq \|f(x)-f(y)\|^2 \leq (1+\epsilon)\|x-y\|^2$$

$$\Downarrow$$

Méthode de projection aléatoire

- voir Halko & al., SIAM Review, 2011
- si ϵ fixé, se ramène à une SVD dans la dimension k
- avec indicateurs de contrôle de la qualité

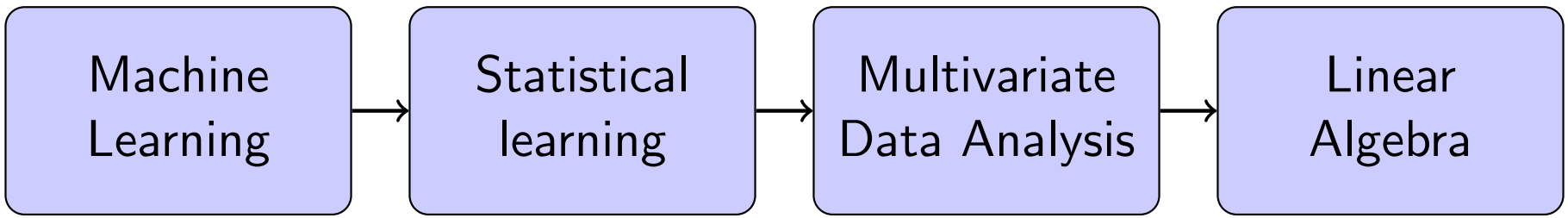
MDS


Séquenceur Bordeaux
fasta files
iRODS
Idris Turing
ascii distance files
INRIA PlaFRIM Bordeaux
HDF5 files
iRODS
Idris Ada
coordinates file
iRODS
Serveur local Laptop

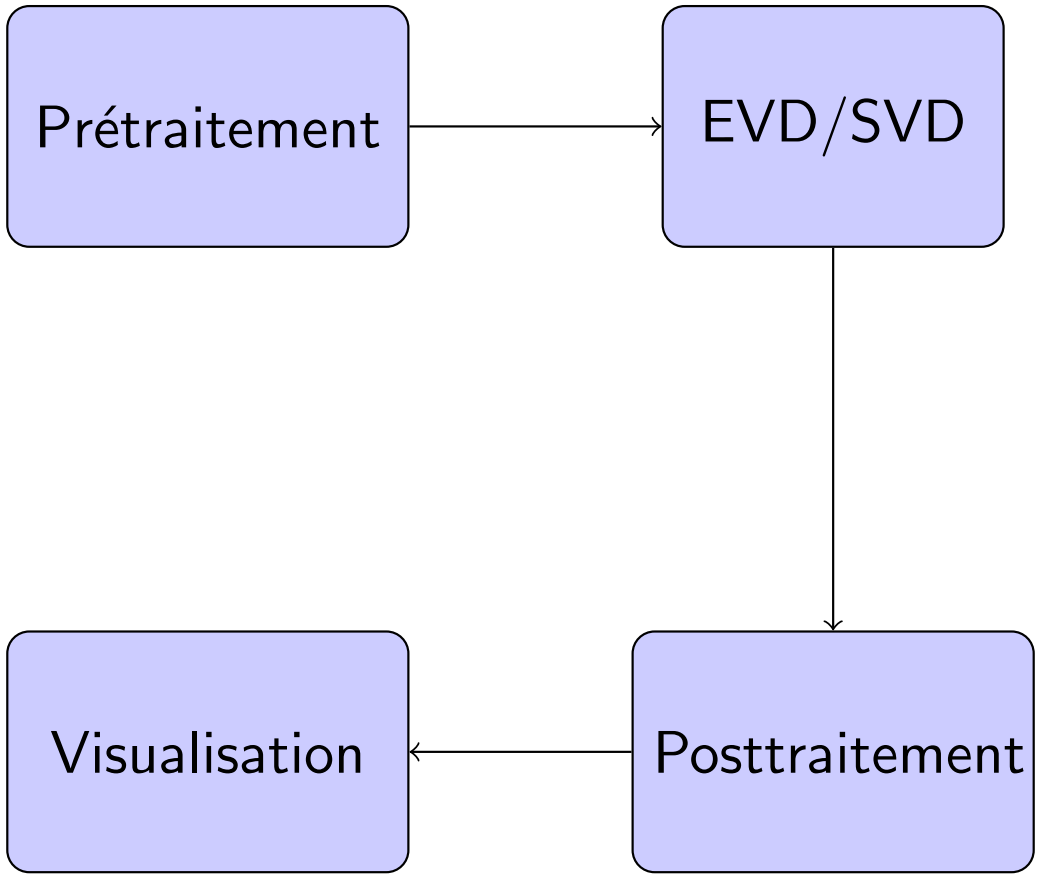
Pont entre l'apprentissage et l'inférence statistique (Tibshirani)

Machine learning	**Statistics**
weights	parameters
learning	fitting
supervised learning	regression/classification
unsupervised learning	density estimation, clustering

... avec l'algèbre linéaire



L'essentiel des méthodes de réduction de la dimension



Quelques exemples

ACP

$$
\begin{aligned}
(U, \Sigma, V) &= \mathrm{SVD}(A) \\
\Lambda &= \Sigma^2 \\
Y &= U\Lambda
\end{aligned}
$$

Canonical Analysis

$$
\begin{aligned}
T &= A'B \\
D_{\mathrm{A}} &= (A'A)^{-1}, \quad D_{\mathrm{B}} = (B'B)^{-1} \\
M_{\mathrm{A,B}} &= D_{\mathrm{A}}^{1/2} T D_{\mathrm{B}}^{1/2} \\
(W_{\mathrm{B}}, \Psi) &= \mathrm{EVD}(M_{\mathrm{A,B}}) \\
W_{\mathrm{A}} &= M W_{\mathrm{B}} \\
U_{\mathrm{A}} &= D_{\mathrm{A}}^{1/2} W_{\mathrm{A}}, \quad U_{\mathrm{B}} = D_{\mathrm{B}}^{1/2} W_{\mathrm{B}} \\
Y_{\mathrm{A}} &= AU_{\mathrm{A}}, \quad Y_{\mathrm{B}} = BU_{\mathrm{B}}
\end{aligned}
$$

MDS

$$
\begin{aligned}
G &= \mathrm{GRAM}(D) \\
(U, \Sigma, U) &= \mathrm{SVD}(G) \\
Y &= U\Sigma^{1/2}
\end{aligned}
$$

AFC

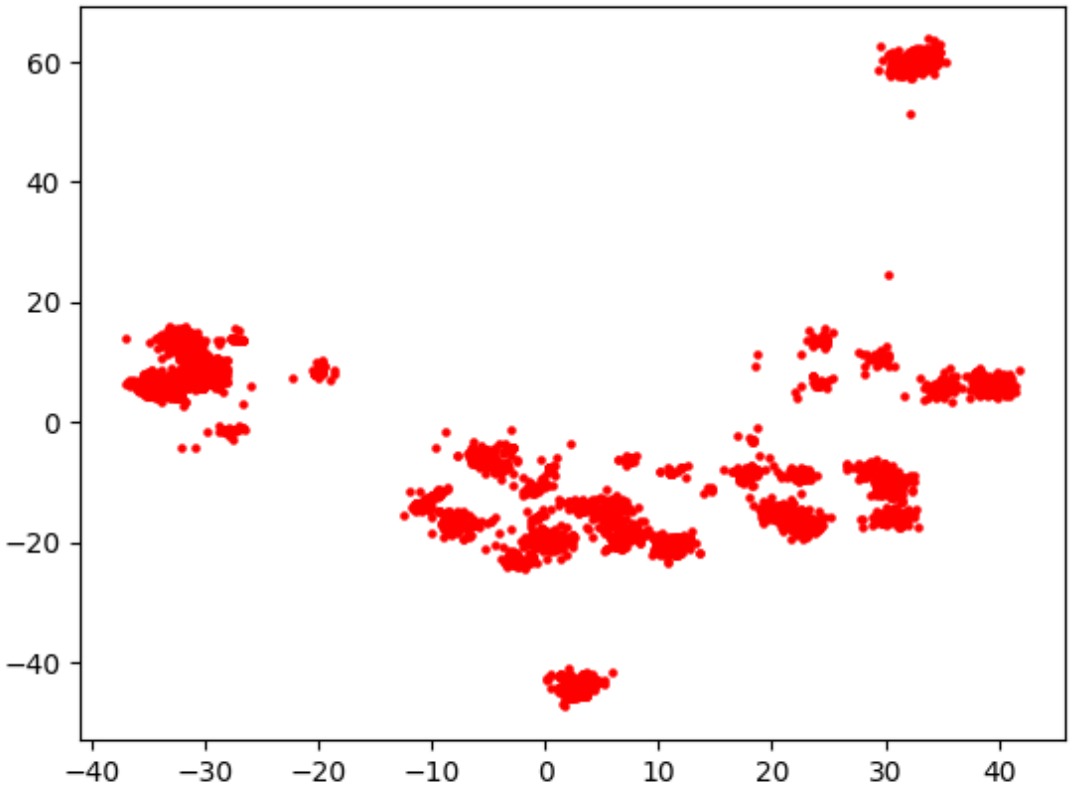
$$
\begin{aligned}
r &= A\mathbf{1}_m \\
c &= A^{\mathrm{T}}\mathbf{1}_m \\
M &= \mathrm{diag}\,(1/\sqrt{r_i}) \\
Q &= \mathrm{diag}\,(1/\sqrt{c_j}) \\
A_{\mathrm{M,Q}} &= M(A - r \otimes c \\
(Z, X, \Lambda) &= \mathrm{PCA}(A_{\mathrm{M,Q}}) \\
\ldots &\quad \ldots
\end{aligned}
$$

Projet *Diodon*

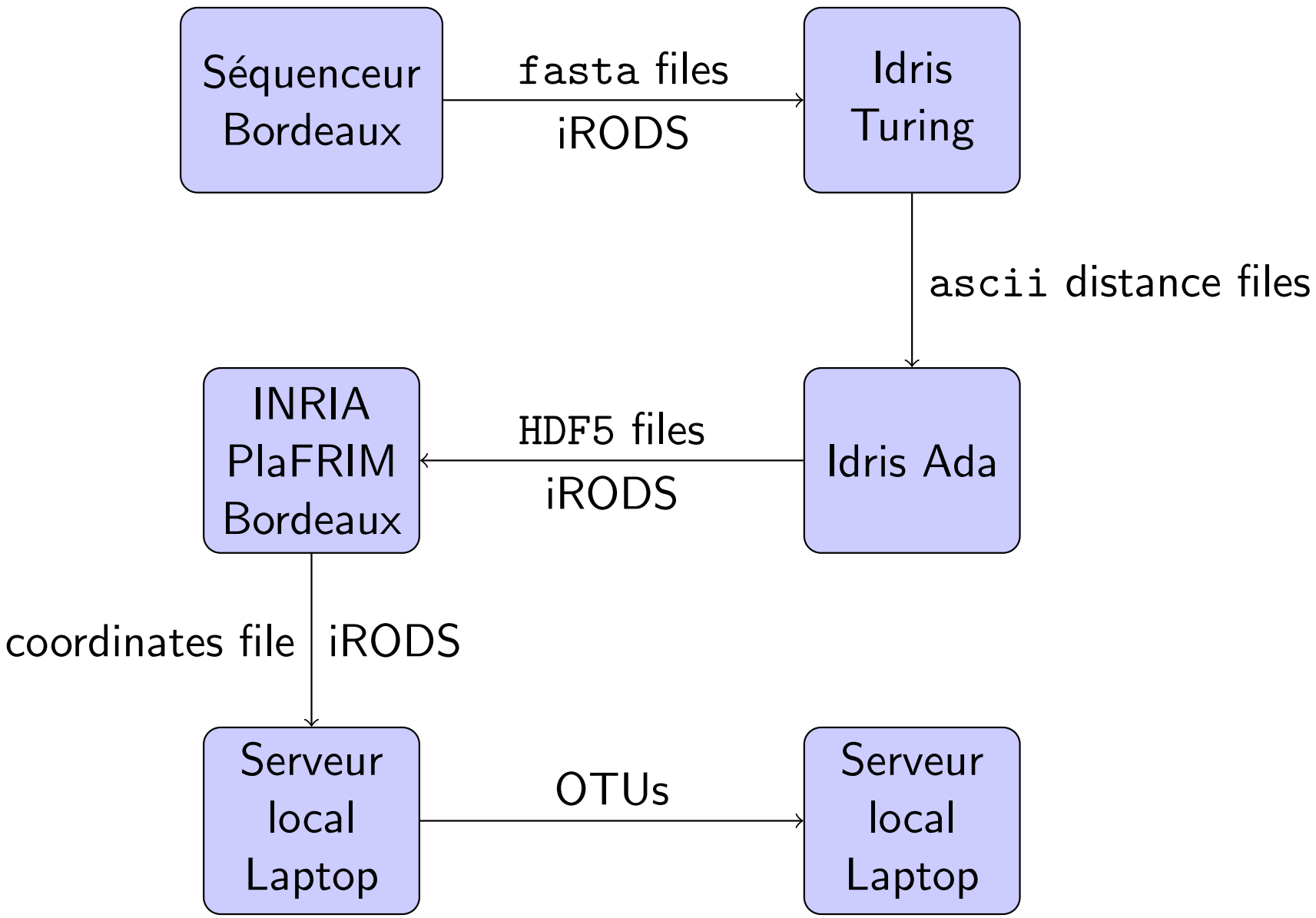
Un projet de librairie ...

- englobant l'essentiel des méthodes dites "multivariate Anaysis"
- basée sur un noyau de SVD par projection aléatoire
- avec vectorisaton et optimisation des phases de prétraitement
- écrite en `python/numpy`, `julia`, `C++`
- actuellement en mémoire partagée sur un nœud
- en cours de développement vers une mémoire distribuée (MPI)
- avec la pile logicielle Inria Bordeaux (deux projets démarrent en 2019)

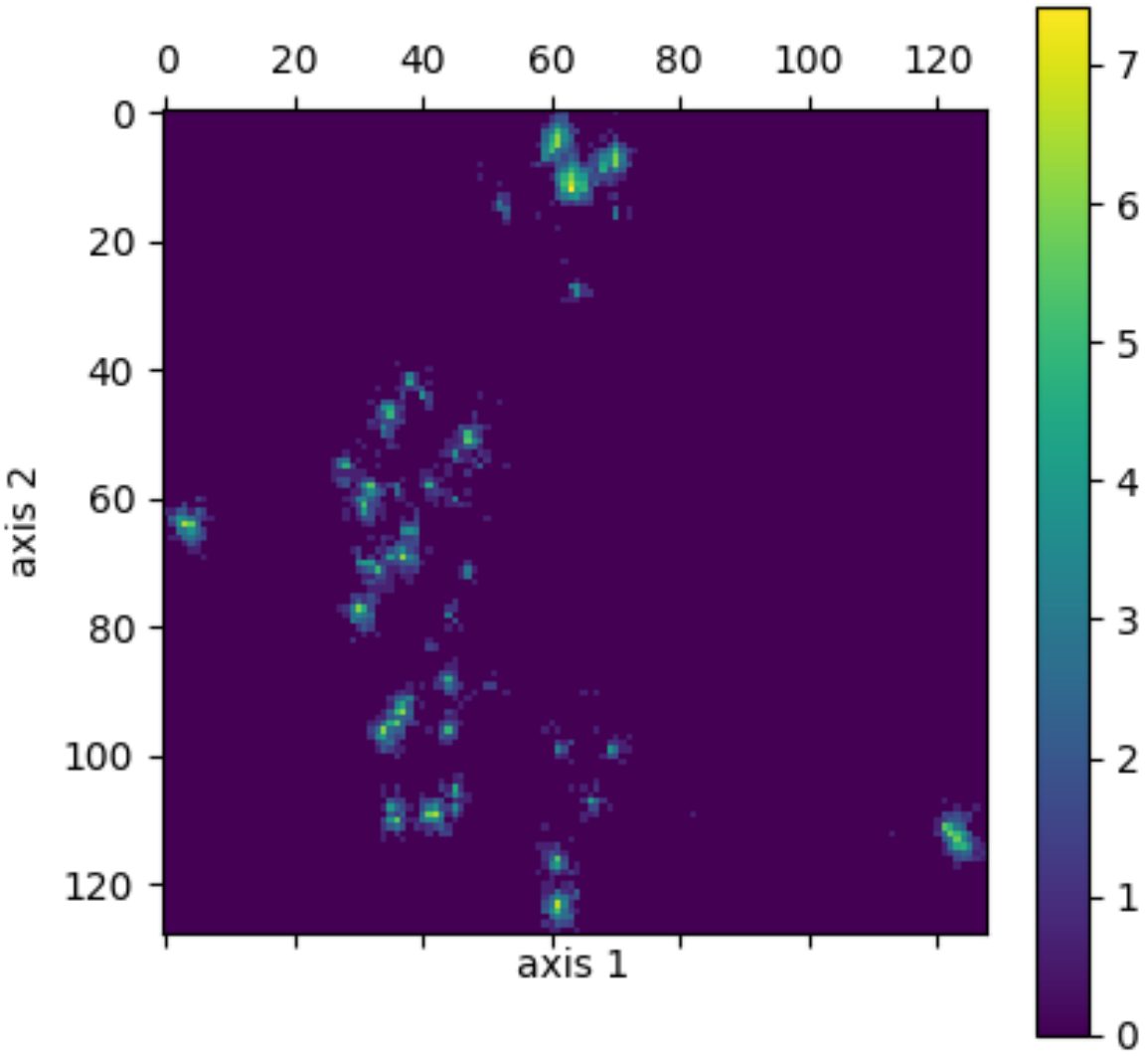
Nuage de points



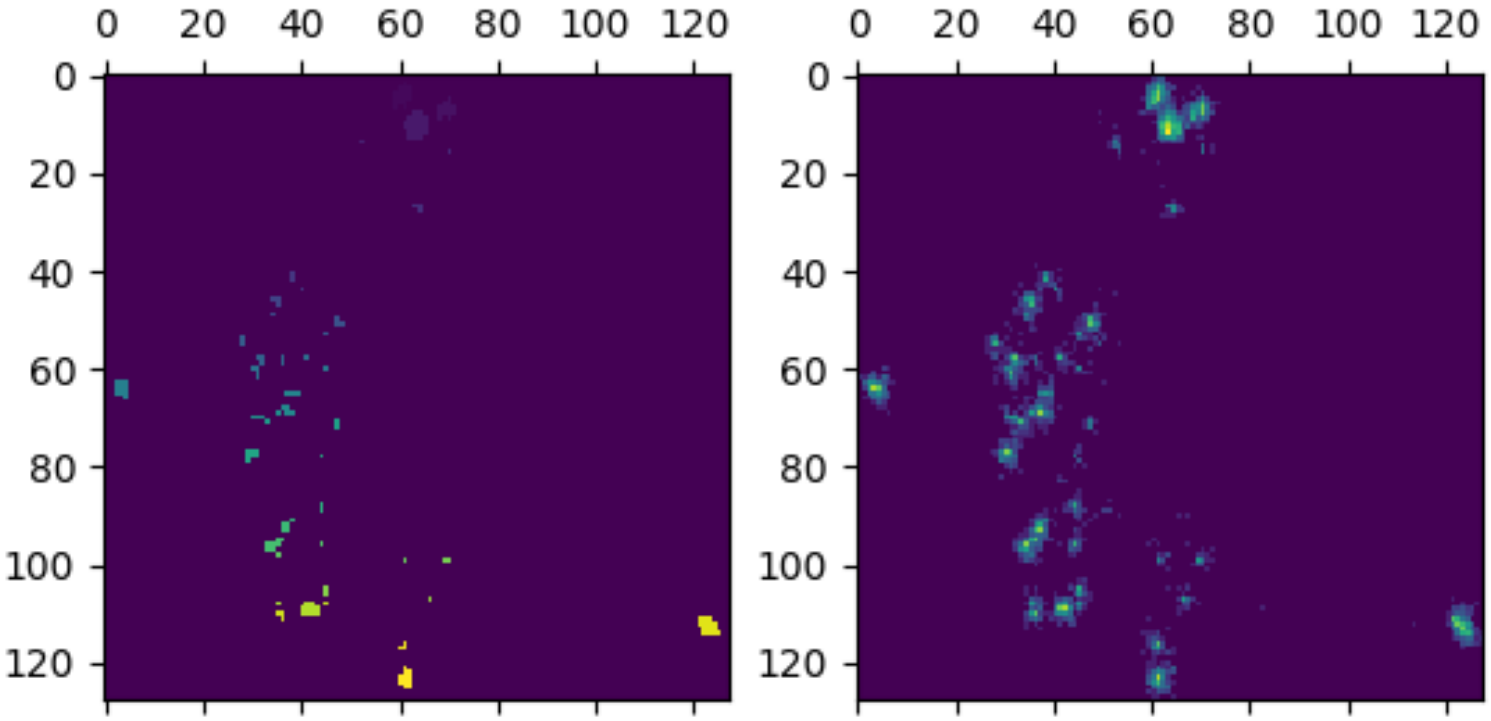
MDS


Séquenceur Bordeaux
fasta files
iRODS
Idris Turing
ascii distance files
INRIA PlaFRIM Bordeaux
HDF5 files
iRODS
Idris Ada
coordinates file
iRODS
Serveur local Laptop
OTUs
Serveur local Laptop

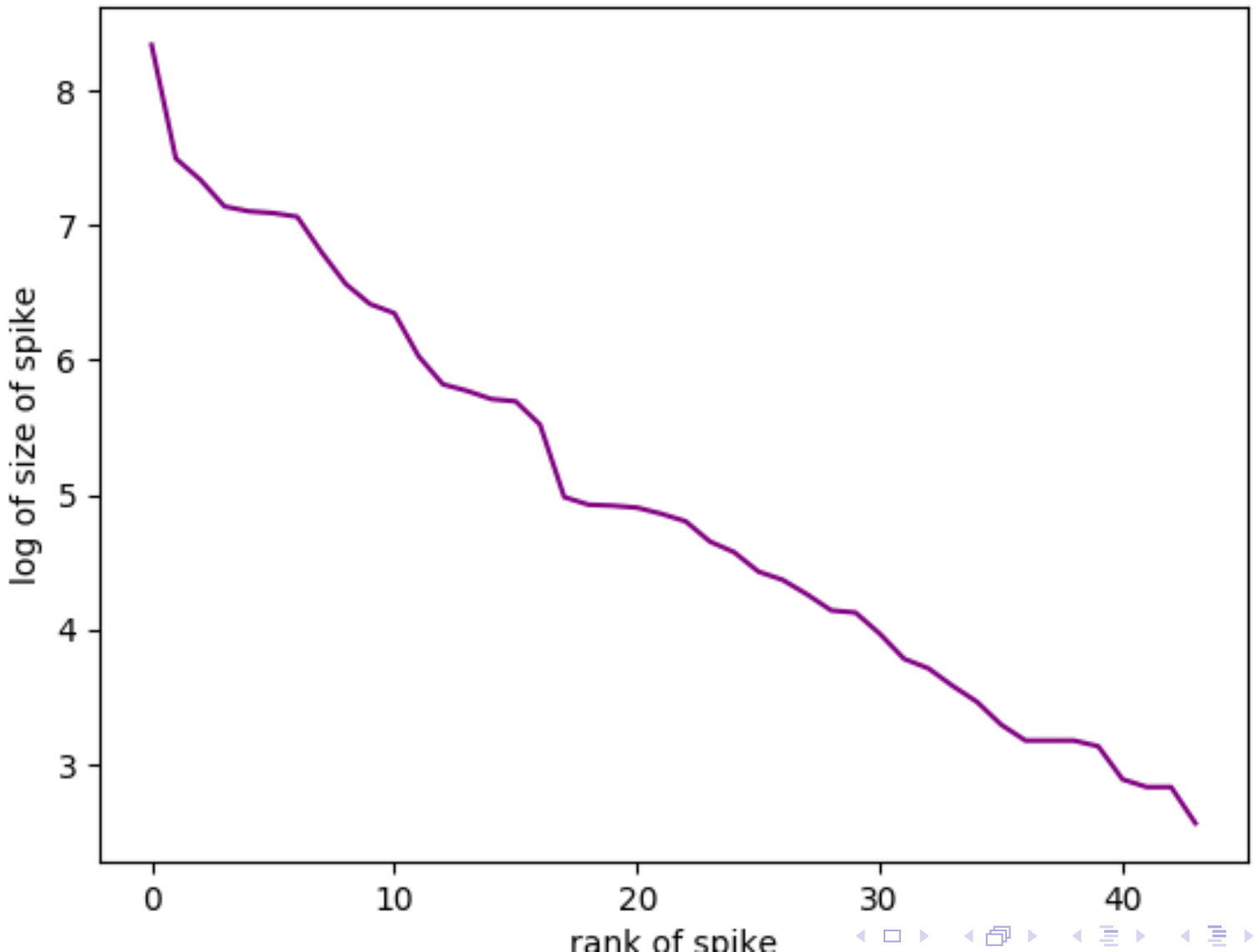
Nuage de points, en densité



Nuage de points, spikes



Distribution rang/taille

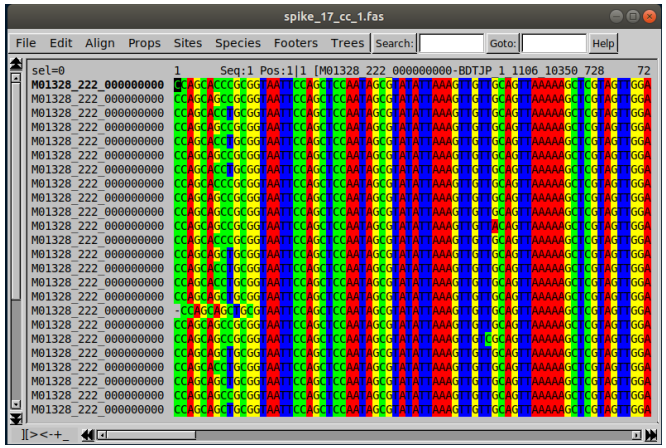


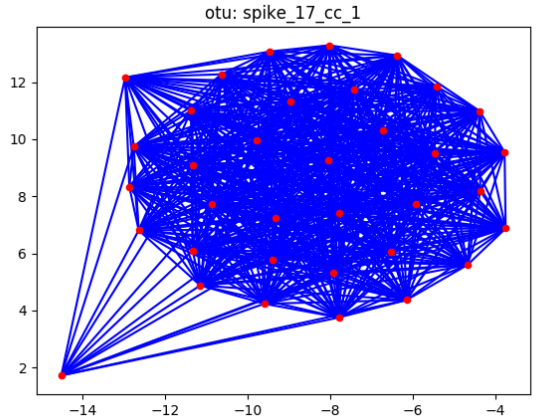
Construction des OTUs

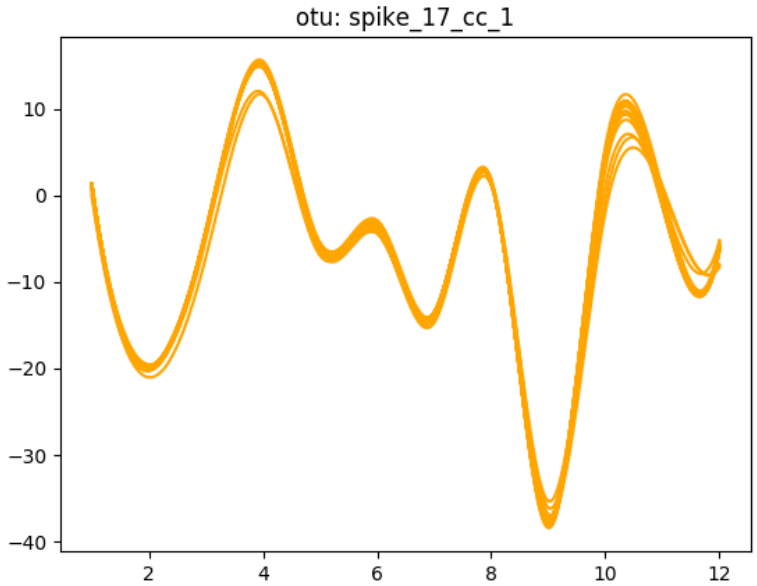
Algorithm 3 pseudocode to build OTUs from MDS coordinates

input Coordinate file, loaded as X, distance file, loaded as D, threshold α, minimum size of an OTU m
build spikes as sets of pixels
for each spike **do**
    list all sequences projected on the spike
    extract the pairwise distances for this spike
    build the threshold graph
    build the connected components for this graph
    if the number of sequences $\geq m$ **then**
        add the cc in the list of OTUs
    end if
end for
return list of OTUs (sequences, distances, coordinates)

Un exemple d'OTU







Prochaines étapes ?

- Optimiser chaque maillon
- GPU ?
- passer à l'échelle en développant Diodon en mémoire/calcul distribué (pile logicielle Inria Bordeaux)
- Construire des bases de référence d'OTUs (intercalibrations des OTUs entre projets et échantillons)

Remerciements

- Les différentes infrastructures de calcul
  - IDRIS (projet DARI Biodiversiton 2015-2016-2018)
  - PlaFRIM (INRIA Bordeaux Sud-Ouest)

  et pour les deux le support aux utilisateurs
- Le projet européen EOSCpilot (2017-2018)
  - le WP6 "Interopérabilité" piloté par l'IN2P3
  - le "tesbed" pico2 sur l'interopérabilité
- Le COST DNAqua.Net (groupe de travail sur le traitement des données)
- Le réseau R-Syst (taxonomie moléculaire, très nombreux échanges)
- Les collègues biologistes ayant partagé leurs données pour la mise au point de ces outils (arbres de Guyane, diatomées du lac Léman et de Scandinavie, champignons phytopathogènes, ...) notamment via le projet Malabar (Bassin d'Arcachon)
- Last but not least: la plateforme PGTB (Franck Salin & Emilie Chancerel) de Pierroton