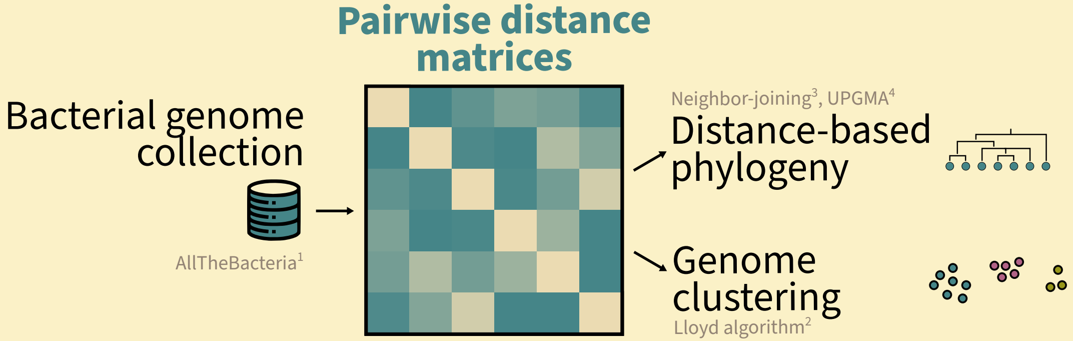
Towards space-efficient data structures for large genome-distance matrices with quick retrieval

Léo Ackermann[1], Pierre Peterlongo[1], Karel Břinda[1]

[1] Inria, Genscale, Rennes

29th November 2024

Importance of pairwise distance matrices

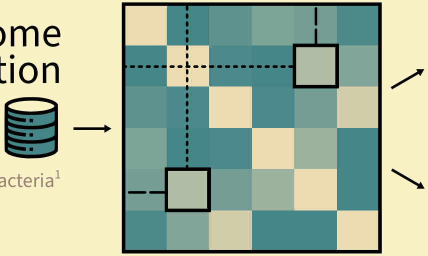

Pairwise distance matrices
Bacterial genome collection
AllTheBacteria[1]
Neighbor-joining[3], UPGMA[4]
Distance-based phylogeny
Genome clustering
Lloyd algorithm[2]

Importance of pairwise distance matrices

Bacterial genome collection

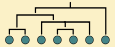
AllTheBacteria[1]

Pairwise distance matrices



Neighbor-joining[3], UPGMA[4]

Distance-based phylogeny



Genome clustering

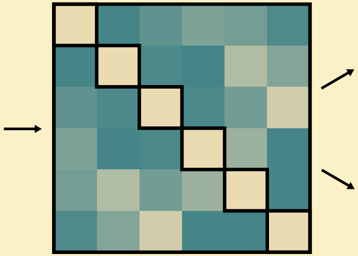
Lloyd algorithm[2]

Importance of pairwise distance matrices

Bacterial genome collection

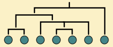
AllTheBacteria[1]

Pairwise distance matrices



Neighbor-joining[3], UPGMA[4]

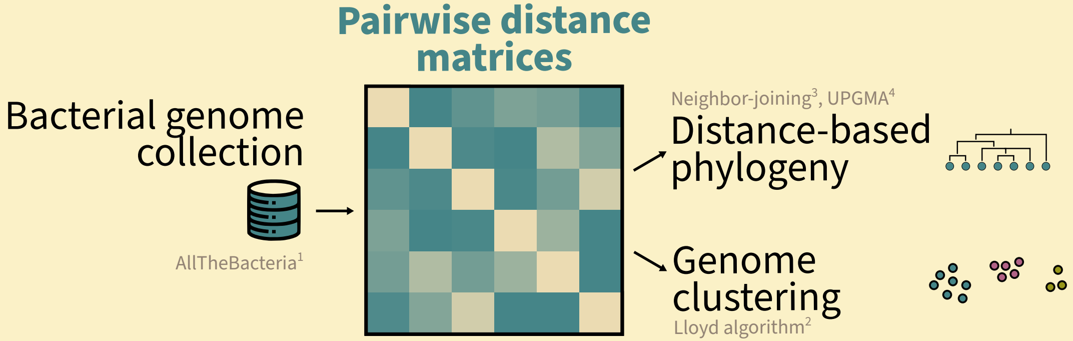
Distance-based phylogeny



Genome clustering

Lloyd algorithm[2]

Importance of pairwise distance matrices

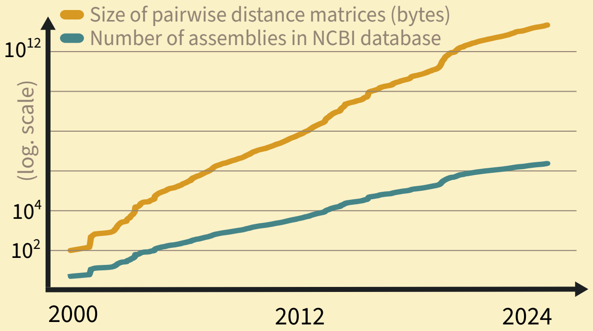

Pairwise distance matrices
Bacterial genome collection
AllTheBacteria[1]
Neighbor-joining[3], UPGMA[4]
Distance-based phylogeny
Genome clustering
Lloyd algorithm[2]

Efficient computation of distance matrices

Sketching (e.g., Mash[5], Dashing[6]) and **parallel computing** make it tractable

Storage of genome distances is challenging

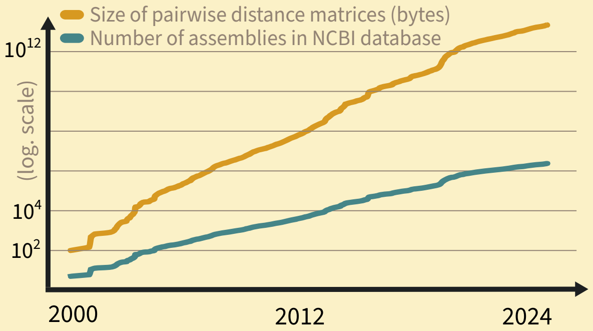
Size of bacterial collections increases exponentially



eg. AllTheBacteria[1]: 4M genomes, 23TB

Storage of genome distances is challenging

Size of bacterial collections increases exponentially



eg. AllTheBacteria[1]: 4M genomes, 23TB

Generic matrix compression techniques

Matrix-specific compression techniques are restricted to sparse and low-rank matrices, and are **not directly applicable**

Problem formulation

This work focuses on the **subquadratic storage of pairwise distance matrices**

Problem formulation

This work focuses on the **subquadratic storage of pairwise distance matrices**

Many variants can be framed

- **Dynamicity.** Whether the structure **can(not) be updated** without recomputing everything

Problem formulation

This work focuses on the **subquadratic storage of pairwise distance matrices**

Many variants can be framed

- **Dynamicity.** Whether the structure **can(not) be updated** without recomputing everything
- **Accuracy.** Whether the structure stores **exact or approximate** distances

Problem formulation

This work focuses on the **subquadratic storage of pairwise distance matrices**

Many variants can be framed

- **Dynamicity.** Whether the structure **can(not) be updated** without recomputing everything
- **Accuracy.** Whether the structure stores **exact or approximate** distances
- **Operability.** A **set of operations** to interact with the data structure, with constraints
  e.g., random access, sequencial access, nothing, . . .

Problem formulation

This work focuses on the **subquadratic storage of pairwise distance matrices**

Many variants can be framed

- **Dynamicity.** Whether the structure **can(not) be updated** without recomputing everything
- **Accuracy.** Whether the structure stores **exact or approximate** distances
- **Operability.** A **set of operations** to interact with the data structure, with constraints
  e.g., random access, sequencial access, nothing, . . .

Focus of our preliminary work (this talk)

I. Restricted to a simplified model of evolution

1. **Lossless static** data structure for pairwise distances with **constant time random access**

II. On a 10k collection of *Streptococcus pneumoniae*

2. **Lossless static** data structure for pairwise distances (without specific operation)
3. **Lossy static** data structure for pairwise distances (without specific operation)

Contrib 1. Method for a simple evolutionary model

ISM: a simple evolutionary model

∞ The infinite site model[7]

It **postulates** that mutations always arise during **vertical descent**, and on a **novel locus**

(because genomes are of *infinite* size)

ISM: a simple evolutionary model

∞ The infinite site model[7]

It **postulates** that mutations always arise during **vertical descent**, and on a **novel locus**

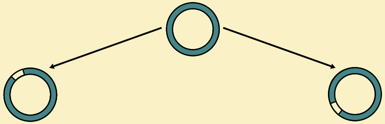
(because genomes are of *infinite* size)

ISM: a simple evolutionary model

The infinite site model[7]

It **postulates** that mutations always arise during **vertical descent**, and on a **novel locus**

(because genomes are of *infinite* size)

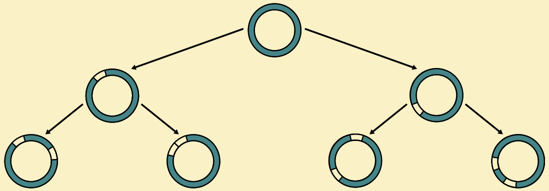


ISM: a simple evolutionary model

The infinite site model[7]

It **postulates** that mutations always arise during **vertical descent**, and on a **novel locus**

(because genomes are of *infinite* size)

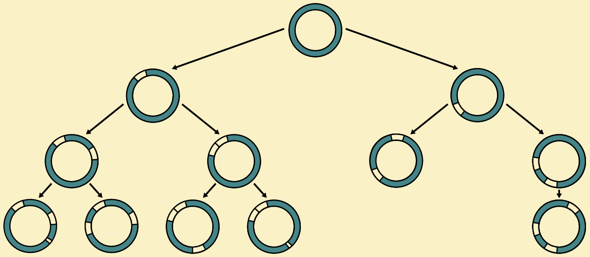


ISM: a simple evolutionary model

The infinite site model[7]

It **postulates** that mutations always arise during **vertical descent**, and on a **novel locus**

(because genomes are of *infinite* size)

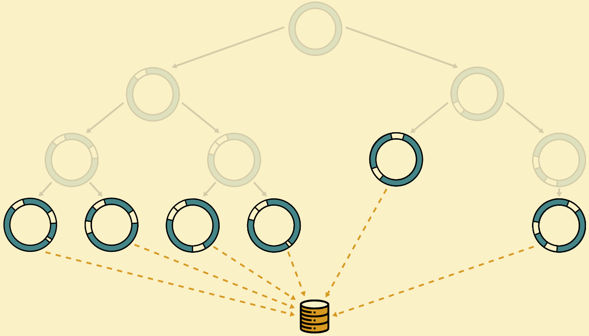


ISM: a simple evolutionary model

The infinite site model[7]

It **postulates** that mutations always arise during **vertical descent**, and on a **novel locus**

(because genomes are of *infinite* size)

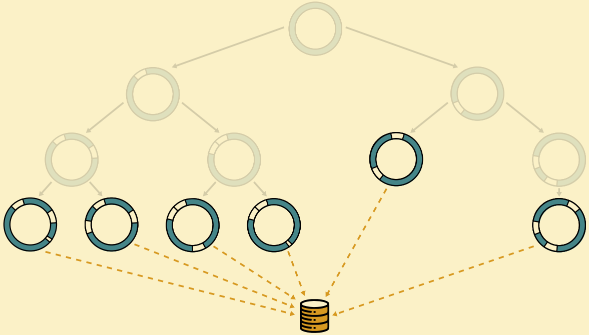


ISM: a simple evolutionary model

The infinite site model[7]

It **postulates** that mutations always arise during **vertical descent**, and on a **novel locus**

(because genomes are of *infinite* size)



The **evolutionary distance** between genomes is defined as the **Hamming distance**

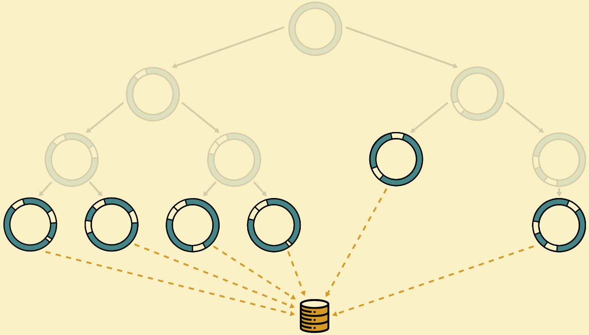
(the number of loci where the genomes differ)

ISM: a simple evolutionary model

The infinite site model[7]

It **postulates** that mutations always arise during **vertical descent**, and on a **novel locus**

(because genomes are of *infinite* size)



A simplistic model, really?

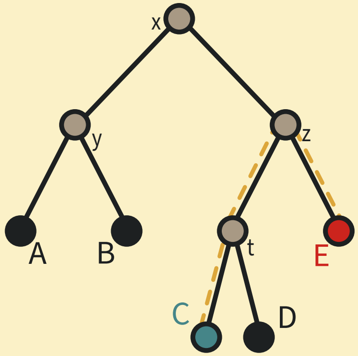
At **small time scale**, real data almost follows such model

(eg. clinical outbreak)

The **evolutionary distance** between genomes is defined as the **Hamming distance**

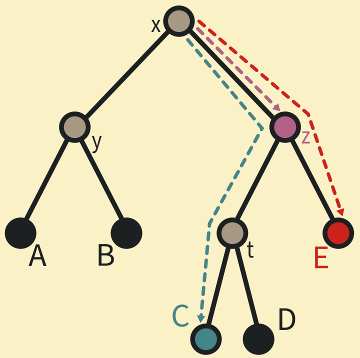
(the number of loci where the genomes differ)

Tree distance in constant-time (with precomputation)[8]



Objective. Compute $\delta(C, E)$
(naive algo. in linear time)

Tree distance in constant-time (with precomputation)[8]



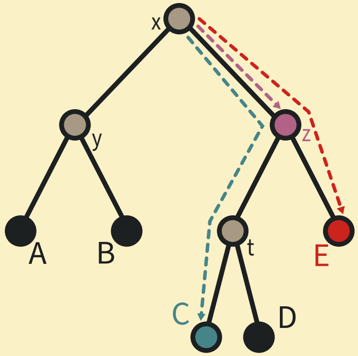
Objective. Compute $\delta(C, E)$
(naive algo. in linear time)

1. Expressing $\delta(C, E)$ with root-to-node distances

$$\delta_T(C, E) = \mathrm{rtn}(C) + \mathrm{rtn}(E) - 2 * \mathrm{rtn}(\mathrm{lca}(C, E))$$

$\rightarrow$ Storing root-to-node distances requires linear space

Tree distance in constant-time (with precomputation)[8]



Objective. Compute $\delta(C, E)$
(naive algo. in linear time)

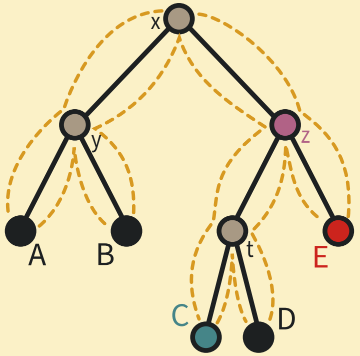
1. Expressing $\delta(C, E)$ with root-to-node distances

$$\delta_T(C, E) = \mathrm{rtn}(C) + \mathrm{rtn}(E) - 2 * \mathrm{rtn}(\mathrm{lca}(C, E))$$

$\rightarrow$ Storing root-to-node distances requires linear space

2. Recover Lowest Common Ancestor in constant time

Tree distance in constant-time (with precomputation)[8]



Objective. Compute $\delta(C, E)$
(naive algo. in linear time)

1. Expressing $\delta(C, E)$ with root-to-node distances

$$\delta_T(C, E) = \mathrm{rtn}(C) + \mathrm{rtn}(E) - 2 * \mathrm{rtn}(\mathrm{lca}(C, E))$$

$\rightarrow$ Storing root-to-node distances requires linear space

2. Recover Lowest Common Ancestor in constant time

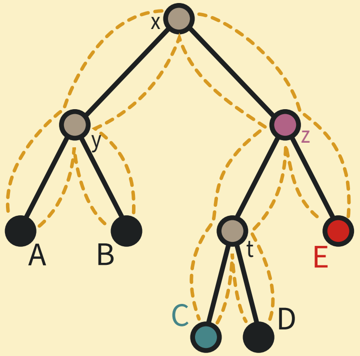
2.a Lowest Common Ancestors are range minima

Eulerian path	xyAyByxzt	CtDtzE	zx
Eulerian depth	012121012	323212	10

$$\mathrm{lca}(C,E) = \mathtt{Path}\left[\mathrm{argmin}_{k \in [\mathtt{index}(C), \mathtt{index}(E)]} \mathtt{Depth}[k]\right]$$

$\rightarrow$ Storing Path and index requires linear space

Tree distance in constant-time (with precomputation)[8]



Objective. Compute $\delta(C, E)$

(naive algo. in linear time)

1. Expressing $\delta(C, E)$ with root-to-node distances

$$\delta_T(C, E) = \mathrm{rtn}(C) + \mathrm{rtn}(E) - 2 * \mathrm{rtn}(\mathrm{lca}(C, E))$$

$\rightarrow$ Storing root-to-node distances requires linear space

2. Recover Lowest Common Ancestor in constant time

2.a Lowest Common Ancestors are range minima

Eulerian path	`xyAyByxzt` **`CtDtzE`** `zx`
Eulerian depth	`012121012` **`323212`** `10`

$$\mathrm{lca}(C,E) = \mathtt{Path}\left[\mathrm{argmin}_{k \in [\mathtt{index}(C), \mathtt{index}(E)]} \mathtt{Depth}[k]\right]$$

$\rightarrow$ Storing Path and index requires linear space

2.b Range-Minimun Queries on `Depth` in constant-time

- Store every query in a lookup table $O(n^2)$ space
- LUT + LUT for block RMQs $O(n)$ space

Contrib 1. $O(n)$-space lossless storage with $O(1)$-random access for ISM data

Lemma

There exists a linear space data structure that can statically store Hamming pairwise distances of genomes following the *infinite site model* while providing constant time random access

CONTRIB 1. $O(n)$-space lossless storage with $O(1)$-random access for ISM data

Lemma

There exists a linear space data structure that can statically store Hamming pairwise distances of genomes following the *infinite site model* while providing constant time random access

Proof

For such data, it can be shown that the **Hamming distance** is **additive**
Hence the **Neighbor-joining** algorithm **exactly recovers the tree**[3]

$\rightarrow$ Hamming distance between genomes is exactly the tree distance between those genomes

CONTRIB 1. $O(n)$-space lossless storage with $O(1)$-random access for ISM data

Lemma

There exists a linear space data structure that can statically store Hamming pairwise distances of genomes following the *infinite site model* while providing constant time random access

Proof

For such data, it can be shown that the **Hamming distance** is **additive**
Hence the **Neighbor-joining** algorithm **exactly recovers the tree**[3]

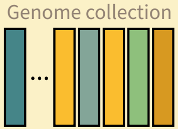
$\rightarrow$ Hamming distance between genomes is exactly the tree distance between those genomes

We **store this tree** using the data structure explained earlier ■

Contrib 2. Methods for real data *(lossless)*

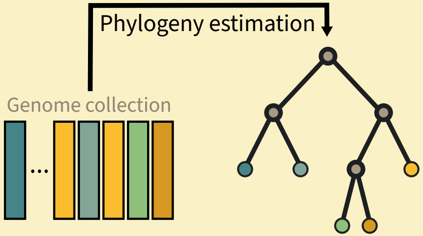
Phylogenetic compression

A phylogeny guided reordering for improved local compressibility



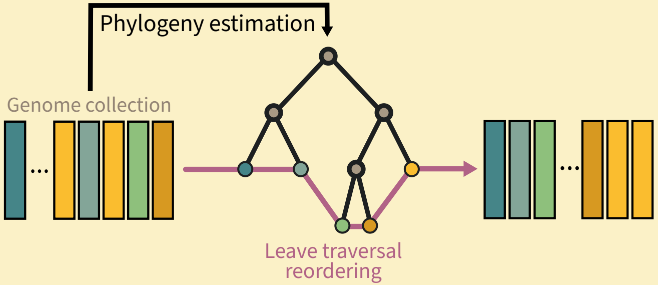
Phylogenetic compression

A phylogeny guided reordering for improved local compressibility



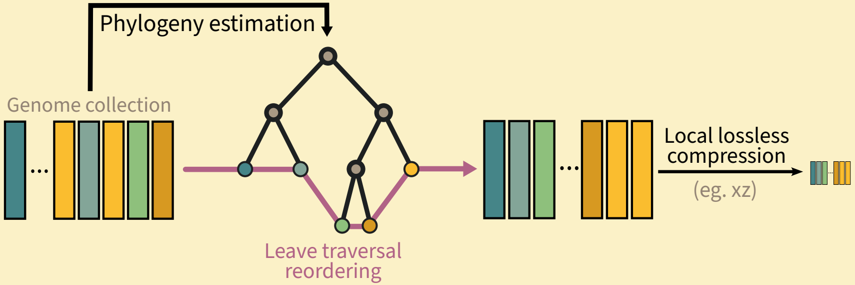
Phylogenetic compression

A phylogeny guided reordering for improved local compressibility



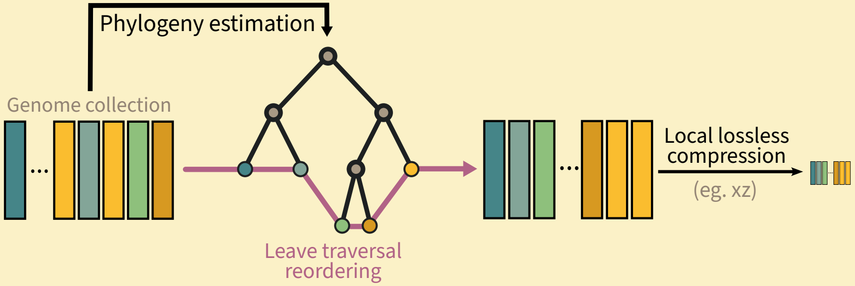
Phylogenetic compression

A phylogeny guided reordering for improved local compressibility



Phylogenetic compression

A phylogeny guided reordering for improved local compressibility

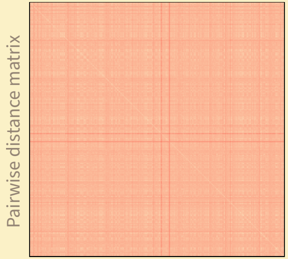


Phylogenetic compression saves orders of magnitude

Running `xz` on reordered genome collections (BIGSIdata, 661k, AllTheBacteria) **saves about two orders of magnitude**[9]

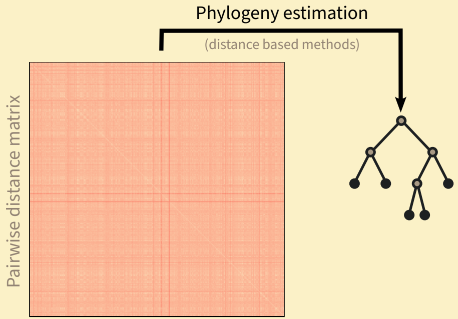
CONTRIB 2. Lossless compression of pairwise genome distances

Phylogenetic compression of pairwise distance matrices



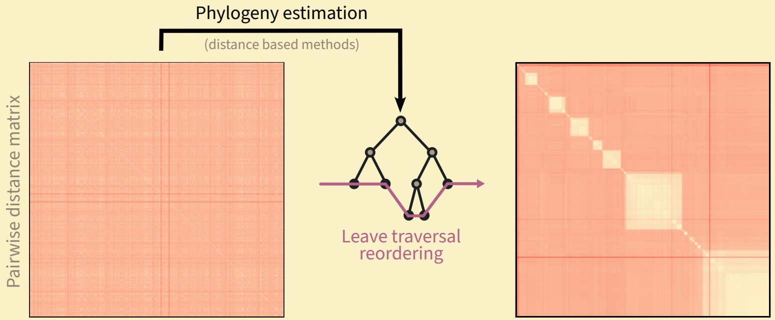
Contrib 2. Lossless compression of pairwise genome distances

Phylogenetic compression of pairwise distance matrices



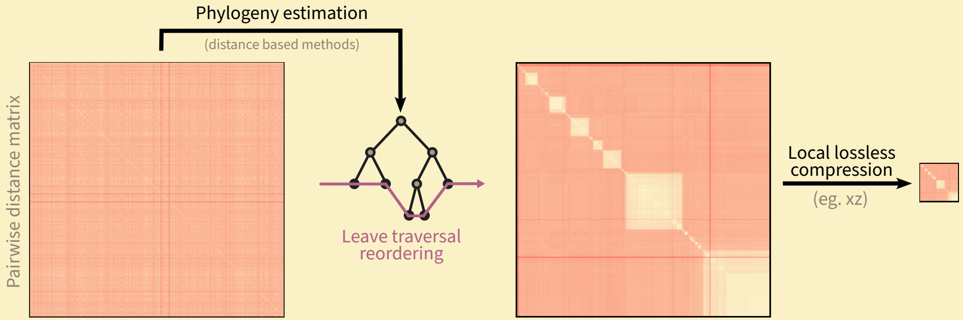
Contrib 2. Lossless compression of pairwise genome distances

Phylogenetic compression of pairwise distance matrices



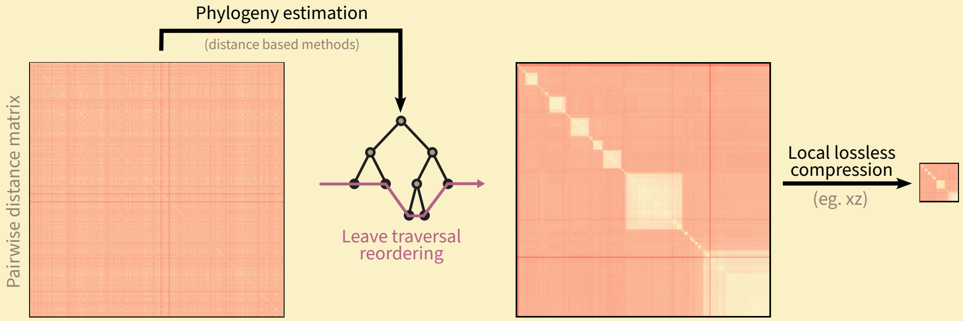
CONTRIB 2. Lossless compression of pairwise genome distances

Phylogenetic compression of pairwise distance matrices



CONTRIB 2. Lossless compression of pairwise genome distances

Phylogenetic compression of pairwise distance matrices



→ Phylogenetic compression can be **applied beyond the scope** of [9]

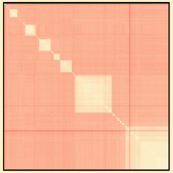
Contrib 3. Methods for real data *(lossy)*

Tree decomposition of distance matrices

⏮ The *infinite site model*, simplistic, really?

At small time scale, real data almost follows such model

$\rightarrow$ Most of the distance signal is explainable by the tree distance



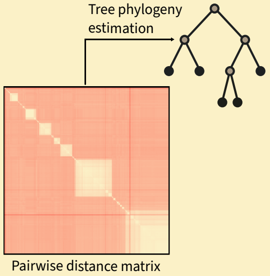
Pairwise distance matrix

Tree decomposition of distance matrices

⏮ The *infinite site model*, simplistic, really?

At small time scale, real data almost follows such model

$\rightarrow$ Most of the distance signal is explainable by the tree distance

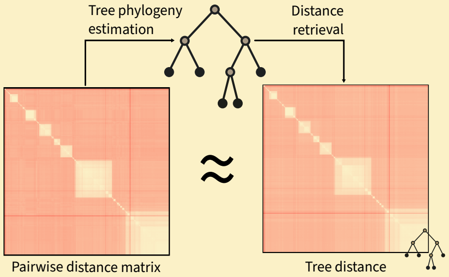


Tree decomposition of distance matrices

⏮ The *infinite site model*, simplistic, really?

At small time scale, real data almost follows such model

→ Most of the distance signal is explainable by the tree distance

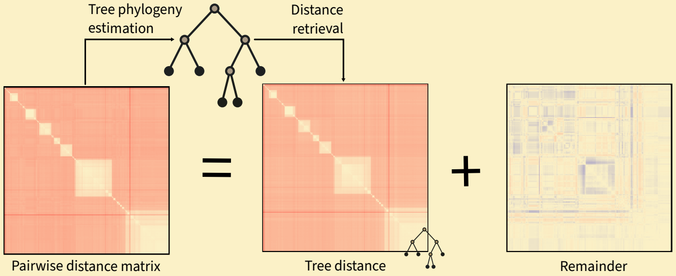


Tree decomposition of distance matrices

The *infinite site model*, simplistic, really?

At small time scale, real data almost follows such model

→ Most of the distance signal is explainable by the tree distance

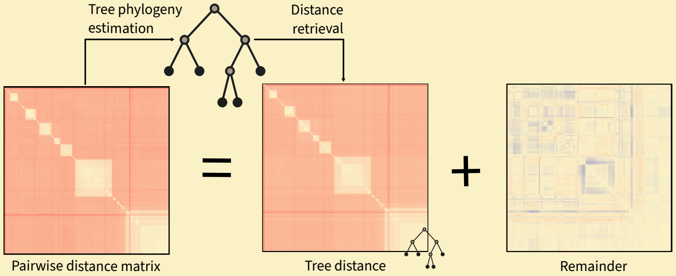


Tree decomposition of distance matrices

The *infinite site model*, simplistic, really?

At small time scale, real data almost follows such model

→ Most of the distance signal is explainable by the tree distance



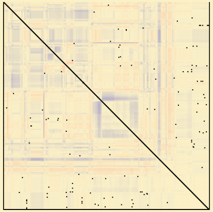
This approach cannot work for lossless compression

The **remainder has a size similar to the original distance**, and does not exhibit strong structure

Biology-informed thresholding

⓪ Thresholding small values of the reminder matrix based on ANI

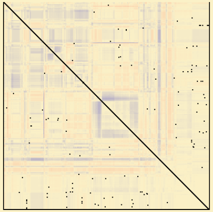
All values below a threshold are set to zero → from 8 to 1 char in the .phylip format

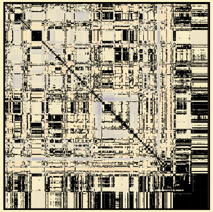


Biology-informed thresholding

⓪ Thresholding small values of the reminder matrix based on ANI

All values below a threshold are set to zero $\rightarrow$ from 8 to 1 char in the .phylip format



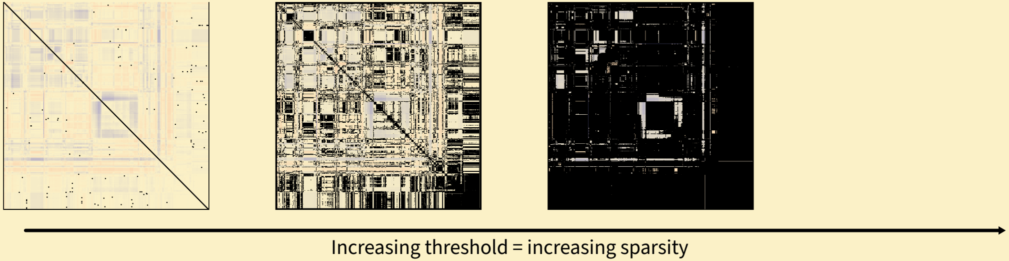


Increasing threshold = increasing sparsity

Biology-informed thresholding

⓪ Thresholding small values of the reminder matrix based on ANI

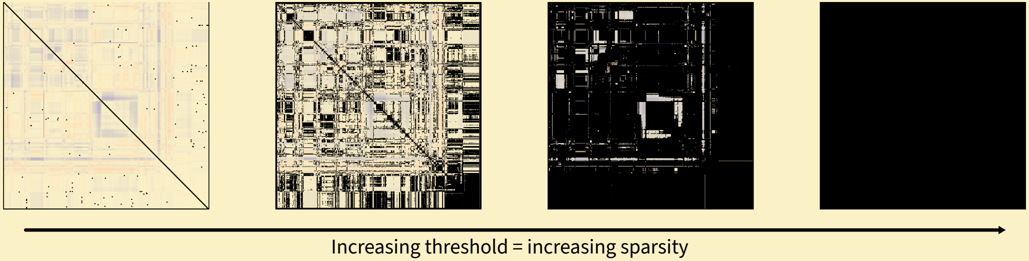
All values below a threshold are set to zero $\rightarrow$ from 8 to 1 char in the .phylip format



Biology-informed thresholding

⓪ Thresholding small values of the reminder matrix based on ANI

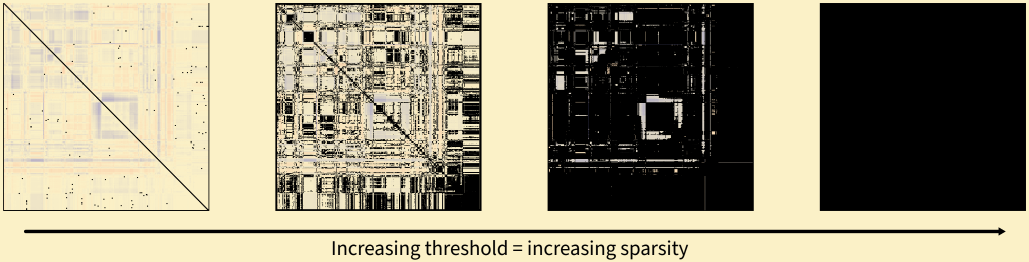
All values below a threshold are set to zero $\rightarrow$ from 8 to 1 char in the .phylip format



Biology-informed thresholding

⓪ Thresholding small values of the reminder matrix based on ANI

All values below a threshold are set to zero $\rightarrow$ from 8 to 1 char in the .phylip format

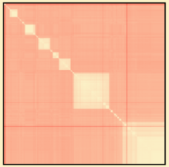


Biological meaning of thresholds (for future work)

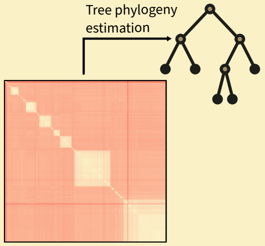
For similar enough genomes, **taxonomy can be defined with distance thresholds**[10]

e.g., Specie $\equiv$ >90% ANI $\equiv$ <0.05 Mash dist. or Strain $\equiv$ >99.99% ANI $\equiv$ <0.0001 Mash dist.

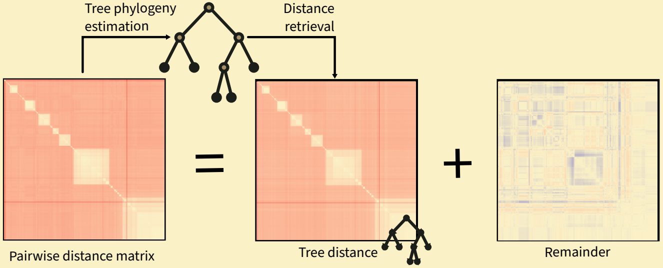
CONTRIB 3. Loss-monitored compression of pairwise genome distances


Pairwise distance matrix

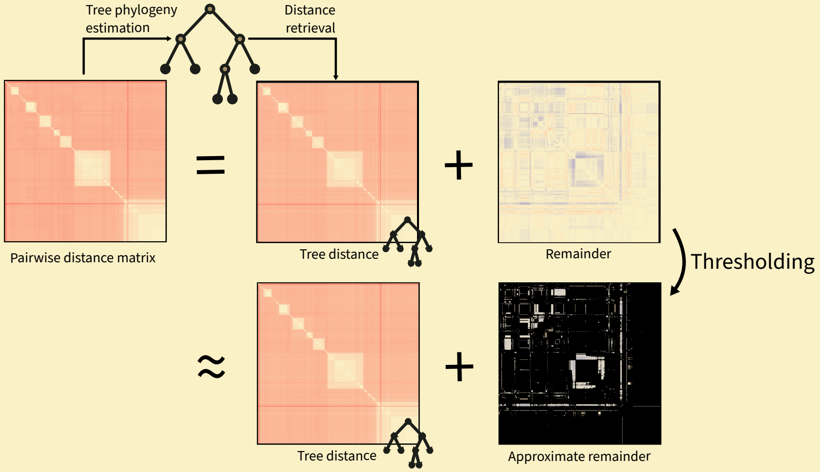
CONTRIB 3. Loss-monitored compression of pairwise genome distances



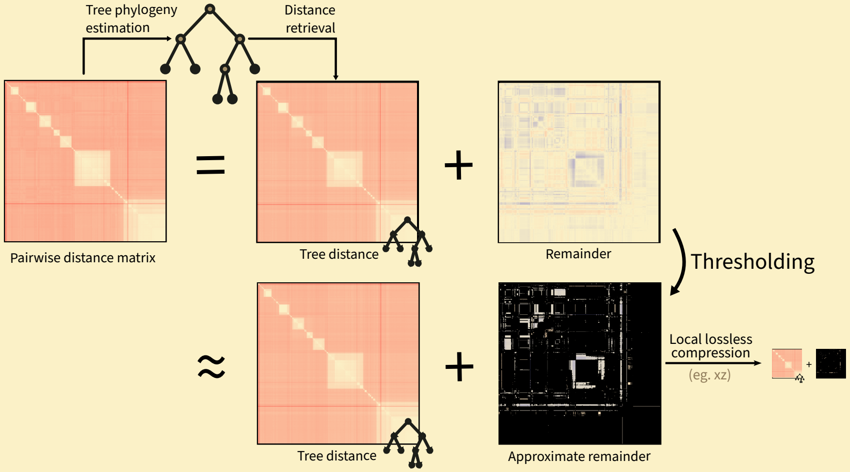
CONTRIB 3. Loss-monitored compression of pairwise genome distances



CONTRIB 3. Loss-monitored compression of pairwise genome distances



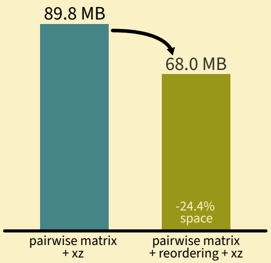
CONTRIB 3. Loss-monitored compression of pairwise genome distances



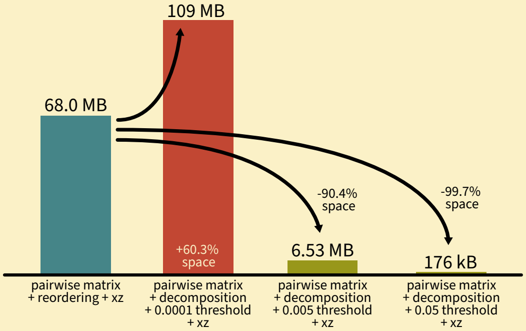
Results

Exploiting phylogeny pushes compressibility bundaries

Data	10k *Streptococcus pneumoniae* genomes from the AllTheBacteria[1] collection
Distance estimator	Mash
Phylogeny estimator	NJ algorithm, with quicktree[11]
Distance retrieval	**new tool** nwk2phy[12] (implements constant-time tree distance retrieval)
Compression scheme	xz



Lossless compression



Lossy compression

Conclusion

Conclusion

Context

Many **downstream analyses** rely on **pairwise distance matrices**, that are already challenging to store due to their **quadratic size**

Approach

We aim to leverage the **specific structure** of genomic data, that can extensively be **explained by the underlying phylogeny**

First results

- **Theory.** Pairwise matrices of genome collections following the *infinite site model* can be stored in **linear space** supporting **constant time** queries
- **Practice.** Phylogeny-aware (lossy) compression of *10k s.-pneumo.* pairwise matrices **saves over 90% space** compared to xz

Future work

- Better compressing of the remainder

  iterative decomposition, compression techniques for sparse matrices
- Theoretical guarantees via tracking of distortion

  e.g., $\forall \tau, \forall (x,y), \quad \delta(x,y) \leq \tau \Rightarrow \tilde{\delta}_\tau(x,y) \leq \tau$
- Exploring alternative phylogenetic backbones and distances

  phylogenetic networks, phylogenetic splits ; dashing, symetrical difference of kmer sets
- Generalisation to many-species collections

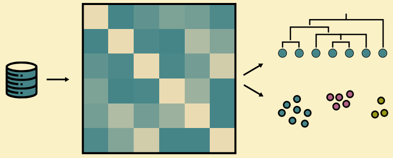
Eventually, distance data structures for **all bacterial genomes** with **monitored distortion**

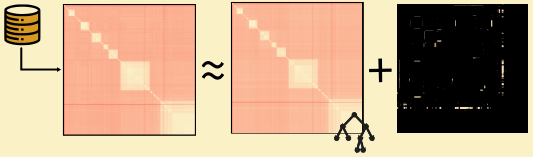
Thank you for your attention!

Léo Ackermann
Pierre Peterlongo
Karel Břinda

Towards space-efficient

data structures for large genome-distance matrices with quick retrieval





References

1. AllTheBacteria - all bacterial genomes assembled, available and searchable. Martin Hunt, Leandro Lima, Wei Shen, John Lees, Zamin Iqbal
2. Least squares quantization in PCM. Stuart P. Lloyd
3. The neighbor-joining method: a new method for reconstructing phylogenetic trees. Naruya Saitou, Masatoshi Nei
4. A Statistical Method for Evaluating Systematic Relationships. Robert Sokal, Charles Michener
5. Mash: fast genome and metagenome distance estimation using MinHash. Brian D. Ondov, Todd J. Treangen, Páll Melsted, Adam B. Mallonee, Nicholas H. Bergman, Sergey Koren, Adam M. Phillippy
6. Dashing: fast and accurate genomic distances with HyperLogLog. Daniel N. Baker, Ben Langmead
7. The Number of Heterozygous Nucleotide Sites Maintained in a Finite Population Due to Steady Flux of Mutations. Motoo Kimura
8. Genome-Scale Algorithm Design Veli Mäkinen, Fabio Cunial, Djamal Belazzougui, Alexandru I. Tomescu
9. Efficient and robust search of microbial genomes via phylogenetic compression. K. Břinda, L. Lima, S. Pignotti, N. Quinones-Olvera, K. Salikhov, R. Chikhi, G. Kucherov, Z. Iqbal, M. Baym
10. An ANI gap within bacterial species that advances the definitions of intra-species units. Luis M. Rodriguez-R, Roth E. Conrad, Tomeu Viver, Dorian J. Feistel, Blake G. Lindner, Stephanus N. Venter, Luis H. Orellana, Rudolf Amann, Ramon Rossello-Mora, Konstantinos T. Konstantinidis
11. https://github.com/khowe/quicktree
12. https://gitlab.inria.fr/lackerma/nwk2phy