Genome Matrices and The Median Problem

Joao Meidanis[1]

University of Campinas, Brazil

December 2017

[1] Supported by FAPESP

Summary

1. Genomes and Matrices
2. Rank Distance and the Median Problem
3. Approximation Algorithm
4. Orthogonal Algorithm
5. M_I Algorithm
6. Future work

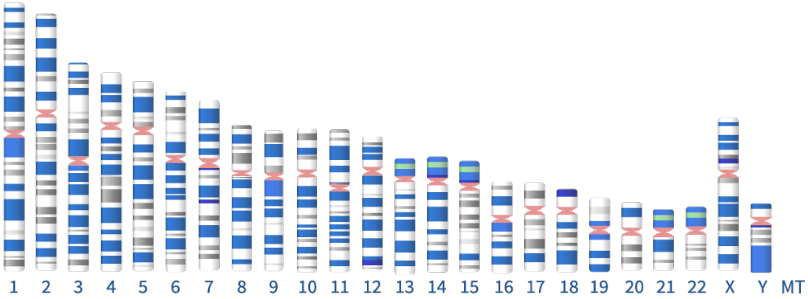
Relevant papers

- Zanetti, J.P.P., Biller, P., Meidanis, J.
  Median approximations for genomes modeled as matrices.
  Bull Math Biol (2016) 78: 786.

- Chindelevitch, L., Meidanis, J.
  On the Rank-Distance Median of 3 Permutations.
  RECOMB-CG Workshop (2017) LNCS 10562: 256.
  Extended version submitted to BMC Bioinformatics.

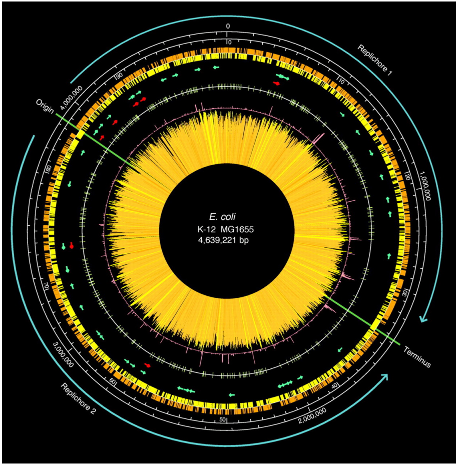
- Chindelevitch, L., Meidanis, J.
  An exact polynomial-time algorithm for the rank median of three genomes.
  Submitted to RECOMB 2018.

The Human Genome



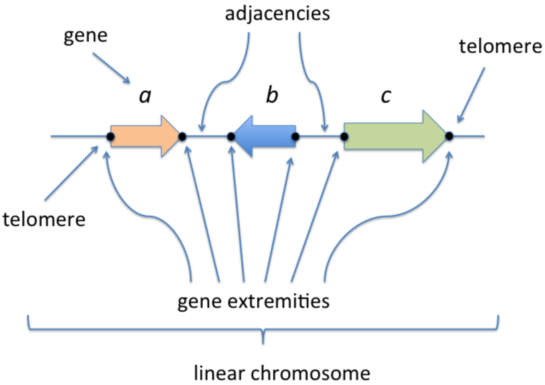
Source: National Center for Biotechnology Information (NCBI), USA

A Bacterial Genome: *E. coli*



Source: Science, 05 Sep 1997: Vol. 277, Issue 5331, pp. 1453-1462

Genome elements



- Adjacencies: $\{a_h, b_h\}, \{b_t, c_t\}$; telomeres: a_t, c_h

Representing genomes as matrices

- Adjacencies: $\{a_h, b_h\}, \{b_t, c_t\}$; telomeres: a_t, c_h

$$\begin{array}{c} a_t \\ a_h \\ b_t \\ b_h \\ c_t \\ c_h \end{array} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Properties

- 0-1 matrices, satisfy $A = A^t = A^{-1}$
- even dimension

Rank Distance

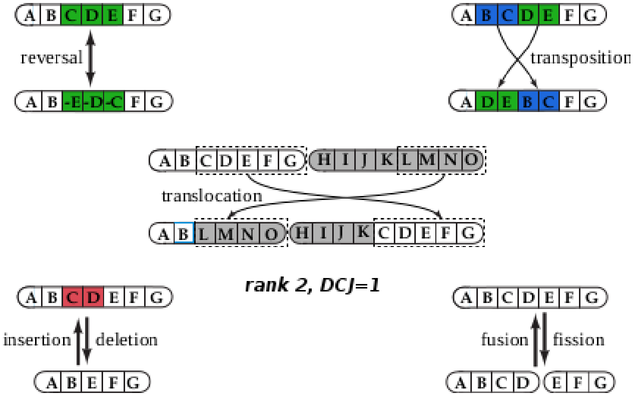
- Distance between two genome matrices is the rank of their difference

$$d(A, B) = r(A - B)$$

Properties

- $r(A + B) \leq r(A) + r(B)$
- Hence, $d(A, C) \leq d(A, B) + d(B, C)$
- $d(A, B) = 0$ if and only if $A = B$
- $d(A, B) = d(B, A)$

Biological Relevance and Double Cut-and-Join (DCJ)



rank 2, DCJ=1

rank 4, DCJ=2

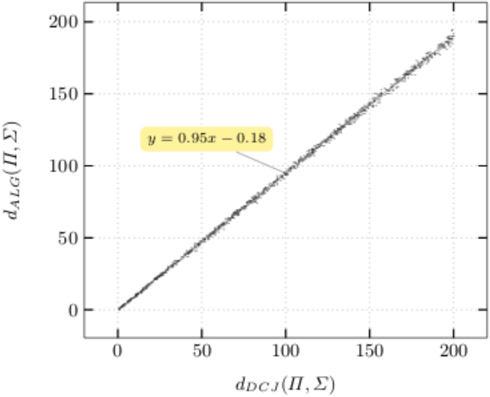
rank 2, DCJ=1

N/A

rank 1, DCJ=1

DCJ vs Rank Distance in Random Genomes

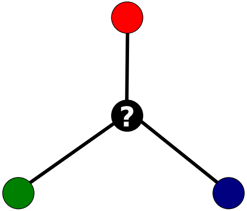
Rank = 2×Algebraic



Source: IEEE/ACM Trans Comput Biol Bioinform, 2013 10(4):819-31.

Matrix Median Problem

Useful for ancestor reconstruction



Definition

Given three input genome matrices A, B, and C, find matrix M minimizing $d(M,A) + d(M,B) + d(M,C)$.

- Polynomial? NP-hard? Nobody knows.

Median may not be genomic

$$\begin{tikzcd}
\begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \arrow[dr] & \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \arrow[d] & \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \arrow[dl] \\
& \begin{bmatrix} -0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & -0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & -0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & -0.5 \end{bmatrix} &
\end{tikzcd}$$

- Need a way to go back from matrices to genomes

Properties of the Median

- Lower Bound

$$d(M, A) + d(M, B) + d(M, C) \geq \frac{d(B, A) + d(C, B) + d(A, C)}{2}.$$

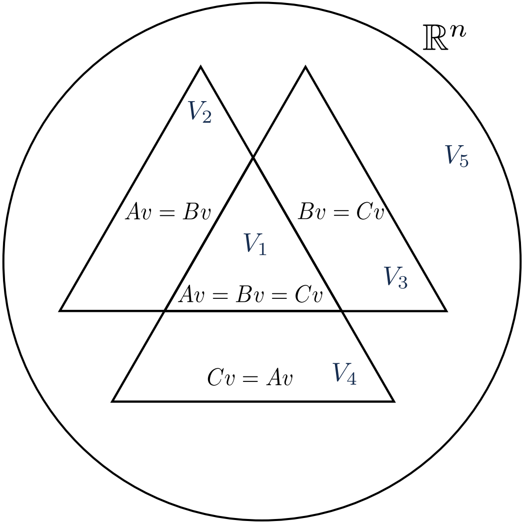
- If M reaches lower bound,

$$d(X, M) + d(M, Y) = d(X, Y),$$

$$M = SX + (I - S)Y,$$

for all $X, Y \in \{A, B, C\}$, and $X \neq Y$

Division into subspaces



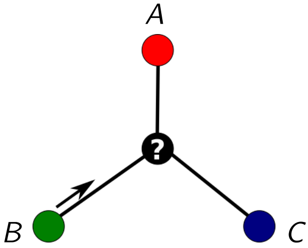
Approximation Algorithm

$$\begin{tikzcd}
\text{Subspaces} & V_1 \arrow[d] & V_2 \arrow[d] & V_3 \arrow[d] & V_4 \arrow[d] & V_5 \arrow[d] \\
\text{Orthonormal Bases} & B_1 \arrow[d] & B_2 \arrow[d] & B_3 \arrow[d] & B_4 \arrow[d] & B_5 \arrow[d] \\
\text{Projection Matrices} & P_1 \arrow[d] & P_2 \arrow[d] & P_3 \arrow[d] & P_4 \arrow[d] & P_5 \arrow[d] \\
 & M_A = AP_1 & + AP_2 & + BP_3 & + AP_4 & + AP_5 \\
\text{Median Candidates} & M_B = BP_1 & + BP_2 & + BP_3 & + AP_4 & + BP_5 \\
 & M_C = CP_1 & + BP_2 & + CP_3 & + CP_4 & + CP_5
\end{tikzcd}$$

- $\frac{4}{3}$ approximation factor for genome matrices
- if $V_5 = \{0\}$ then each candidate is a median (reaches lower bound)

Orthogonal matrices

- Tests with small matrices suggested looking at **orthogonal matrices**
- Exact, polynomial-time algorithm



- “Walk towards the median”
- Find rank 1 matrix H such that $B + H$ is closer to both A and C
- Always possible!

Orthogonal matrices

- Algorithm

```
while $d(A, B) + d(B, C) > d(A, C)$ do
    Find non-zero $u \in \mathrm{im}(A - B) \cap \mathrm{im}(C - B)$
    $B \leftarrow B - 2uu^T B / u^T u$
end
return $B$
```

- Nondeterministic
- Does it reach all medians?

Implementation

Software

- GNU Octave 3.8.1
- Chooses matrix closer to median to "walk"
- Computes $\mathrm{im}(A - B) \cap \mathrm{im}(C - B)$ as:

  $$V = \text{null([(null(A'-B'))'; (null(C'-B'))'])}$$

  where X' is X^T, the transpose of X
- Tries all columns of V
- Also code in R, python

Hardware

- Laptop, 8 GB memory, 4 cores, AMD A8-7410
- Windows 10 + WSL

Data Sets

Simulation

- Start with random genome
- Apply random rearrangement operations
- Repeat to get A, B, C

Parameters

- sizes: 12, 16, 20, 30, 50, 100, 200, 300, 500 extremities
- type of operation: Add/remove adjacencies (near) or DCJ (far)
- number of operations: 5% to 30%
- 10 $\times$ each
- 1,080 instances

Results

Near

- For all instances, the algorithm finds a median

Far

- For all but 5, the algorihtm finds a median
- Five instances **do not converge**: sizes 16, 20, and 30
- Not the biggest sizes!!

Times to run all instances of a given size

size	Near	Far
500	27 min	**7:30 h**
300	4 min	0:50 h
200	2 min	0:13 h
100	1 min	0:01 h

Alternative approach

Drawbacks of Orthogonal Algorithm

- Lack of convergence
- Not fast enough

Insights from Orthogonal Algorithm

- Medians reach the lower bound
- For any median M:

$$Xv = Yv \implies Mv = Xv = Yv,$$

for $X, Y \in \{A, B, C\}$ and $X \neq Y$

M_I Median

- M_I follows majority in V_1 through V_4
- M_I follows I in V_5

$$\begin{tikzcd}
\text{Subspaces} & V_1 \arrow[d] & V_2 \arrow[d] & V_3 \arrow[d] & V_4 \arrow[d] & V_5 \arrow[d] \\
\text{Bases} & B_1 \arrow[d] & B_2 \arrow[d] & B_3 \arrow[d] & B_4 \arrow[d] & B_5 \arrow[d] \\
\text{Projection Matrices} & P_1 \arrow[d] & P_2 \arrow[d] & P_3 \arrow[d] & P_4 \arrow[d] & P_5 \arrow[d] \\
\text{Median} & M_I = AP_1 & + AP_2 & + BP_3 & + AP_4 & + IP_5
\end{tikzcd}$$

Efficient Computation

Technical Improvements

- B_5 not needed
- B_i don't need to be orthonormal, $i = 1..4$
- B_i's computed from permutation **vectors** and DFS
- B_i's all binary
- Improved formula

$$M_I = I + ([AB_1, AB_2, BB_3, AB_4] - B_{14})(B_{14}^T B_{14})^{-1} B_{14}^T$$

where B_i is a basis of V_i for $i = 1..4$ and

$$B_{14} = [B_1, B_2, B_3, B_4]$$

Results

Near

- For all 540 cases, the algorithm finds a median
- Median is genomic in 535 cases

Far

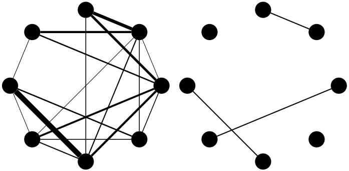
- For all 540 cases, the algorihtm finds a median
- Median is genomic in 254 cases

Times to run all instances of a given size

size	o-Near	o-Far	mi-Near	mi-Far
500	27 min	7:30 h	9:52 min	8:24 min
300	4 min	0:50 h	3:26 min	2:34 min
200	2 min	0:13 h	1:40 min	1:08 min
100	1 min	0:01 h	0:30 min	0:24 min

From matrices back to genomes

$$\begin{bmatrix} 0.2 & 0.8 & 0.5 & 0 & 0 & 0.4 & 0 & 0.1 \\ 0.4 & 0 & 0 & 0 & 0 & 0.3 & 0 & 0.6 \\ 0.3 & 0 & 0.5 & 0.2 & 0 & 0 & 0 & 0.3 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0.1 & 0 & 0 & 0.1 & 0.1 & 0.4 & 0.2 & 0.7 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0.3 & 0 & 0 & 0.5 & 0.1 & 0 & 0.4 & 0.1 \\ 0 & 0.8 & 0.2 & 0 & 0 & 0.8 & 0.2 & 0.3 \end{bmatrix}$$



$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

- Assign weight $|a_{ij}| + |a_{ji}|$ to edge ij
- Take a maximum weight matching as your solution
- A genome is a matching of gene extremities

Future work

- Get genomes from medians
- Tests with mammals, bacteria, plants, etc.
- Try **minimax** matrices

$$\text{minimize } \max\{d(A, M), d(B, M), d(C, M)\}$$

- Determine all sorting scenarios (done for DCJ)
- Extension for gene deletions and insertions (done for DCJ)
- Extension for duplicated genes (done for DCJ)
- Technical issues: NP-hardness, convergence, etc.

Get this presentation:

`http://www.ic.unicamp.br/~meidanis/research/rear/`