Genome Comparison
Measures and Approaches

Christoph Dieterich

Department of Evolutionary Biology
Max Planck Institute for Developmental Biology
Introduction

Gene content

Gene order

Signed permutations

Genomes

Problem

What is a genome? The complete inventory of all heritable nucleic acids that determines the genetic identity of an organism is called genome.

1. Viral genomes - DNA and RNA viruses.
2. Bacteria - Circular DNA.
3. Eukaryotes - distributed over linear DNA pieces (chromosomes).
Types of comparison

- **Within-genome comparisons** focus on the genome of a single species. Variations on base composition, $k$-tuple frequency, gene density, numbers and kinds of transposable elements and segmental duplications.

- **Between-genome comparisons** employ closely related species for identifying conserved genes, gene structure and organization and control elements. More distantly related species are used for phylogenetic profiling.
Compositional measures

$k$-tuple compositions of genomes are not uniformly distributed along the genome. Take $k = 1$ as an example: In the human genome, gene-rich regions typically have a higher $\%G+C$ content than gene-poor regions.
Introduction

Gene content

Gene order

Signed permutations

GC skew

A statistic for prokaryotic genomes is the GC skew:

\[ GC_w = \frac{\# G_w - \# C_w}{\# G_w + \# C_w} \]

where \( w \) is a sequence window.
Properties of the *Yersinia pestis* genome.

The innermost circle represents GC skew. There are regions where the GC skew reverses sign. This indicates recent inversions.
Codon usage

The Genetic Code

Christoph Dieterich

Genome Comparison
Differences among species in selection on codon usage. Average of positive $\Delta RSCU$ values (differences in codon usage for highly and lowly expressed genes) per species indicate that 6 species have particularly strong selection on codon bias, spanning low, medium, and high GC-content genomes. Symbols indicate different clades within the nematode phylogeny.
Gene content

1. Prerequisite: Gene annotation (usually via HMMs; see Chapter x).
2. Coding or non-coding sequence (RNA versus protein coding genes).
Gene content

<table>
<thead>
<tr>
<th></th>
<th>H. influenzae</th>
<th>S. cerevisiae</th>
<th>C. elegans</th>
<th>D. melanogaster</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total no. of predicted genes</td>
<td>1709</td>
<td>6241</td>
<td>18424</td>
<td>13601</td>
</tr>
<tr>
<td>No. of genes duplicated(^1)</td>
<td>284</td>
<td>1858</td>
<td>8971</td>
<td>5536</td>
</tr>
<tr>
<td>Total no. of distinct families(^2)</td>
<td>1425</td>
<td>4383</td>
<td>9453</td>
<td>8065</td>
</tr>
</tbody>
</table>

\(^1\) Local duplications

\(^2\) Each protein sequence constitutes a vertex; each HSP between protein sequences is an arc, weighted by the BLAST Expect value. The algorithm identifies protein families by first breaking all arcs with an E value greater than some user-defined value (10\(^{-6}\) was used for all of the analyses reported here). The resulting graph is then split into subgraphs that contain at least two-thirds of all possible arcs between vertices. The algorithm is ”greedy”; that is, it arbitrarily chooses a starting sequence and adds new sequences to the subgraph as long as this criterion is met.
## Gene content

<table>
<thead>
<tr>
<th></th>
<th><em>S. cerevisiae</em></th>
<th><em>C. elegans</em></th>
<th><em>D. melanogaster</em></th>
<th><em>A. thaliana</em></th>
<th><em>H. sapiens</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of genes</td>
<td>5,500</td>
<td>18,400</td>
<td>13,600</td>
<td>26,400</td>
<td>25,000</td>
</tr>
<tr>
<td>% Coding</td>
<td>70</td>
<td>27</td>
<td>20</td>
<td>26.3</td>
<td>1.2</td>
</tr>
<tr>
<td>Gene size (average bp)</td>
<td>1450</td>
<td>2700</td>
<td>3250</td>
<td>1970</td>
<td>27,000</td>
</tr>
<tr>
<td>Exon size (average bp)</td>
<td>1450</td>
<td>240</td>
<td>425</td>
<td>164</td>
<td>145</td>
</tr>
<tr>
<td>Exons/gene (average)</td>
<td>1</td>
<td>6</td>
<td>4</td>
<td>5.2</td>
<td>9.5</td>
</tr>
</tbody>
</table>
Introduction

Gene content

Gene order

Signed permutations

Clusters of orthologous genes

COGs; Tatusov et al. 1997

- Start with a set of proteins from complete genomes.
- Perform an "all-against-all" series of BLASTP comparisons.
- Treat multiple hits from within-organism comparison as cluster of paralogs.
- Identify reciprocal best hits across species.
- Define a COG as the set of mutually consistent best hits in a minimum of three species.
- Expand the graph of any COG by merging graphs that have edges in common.
Clusters of orthologous genes
## Gene content comparison

### Example

What is an archaebacteria? What is an eukaryote? Using COGs we have a way of studying the distribution of proteins across the tree of life.

<table>
<thead>
<tr>
<th>Proteins</th>
<th>$OTU_1$</th>
<th>$OTU_2$</th>
<th>$OTU_3$</th>
<th>$\ldots$</th>
<th>$OTU_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_1$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>$\ldots$</td>
<td>1</td>
</tr>
<tr>
<td>$p_2$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>$\ldots$</td>
<td>1</td>
</tr>
<tr>
<td>$p_3$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$\ldots$</td>
<td>1</td>
</tr>
<tr>
<td>$p_4$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>$\ldots$</td>
<td>1</td>
</tr>
</tbody>
</table>
Clustering Gene Content

Binary vectors of the same length could be clustered via their Hamming distance.

Based on a $n \times n$ distance matrix, a hierarchical clustering can be performed. This method builds the hierarchy from the individual elements by progressively merging "closest" clusters. We distinguish:

- complete linkage clustering: $\max\{ d(x, y) : x \in A, y \in B \}$
- single linkage clustering: $\min\{ d(x, y) : x \in A, y \in B \}$
- average linkage clustering: $\frac{1}{\text{card}(A) \text{card}(B)} \sum_{x \in A} \sum_{y \in B} d(x, y)$
Turnip vs. Cabbage

Look and taste different but share a recent common ancestor.
Turnip vs. Cabbage: Comparing gene content

- Comparing gene sequence yields no evolutionary information.
- In 1980s Jeffrey Palmer studied evolution of plant organelles by comparing mitochondrial genomes of the cabbage and turnip.
- 99% similarity between genes.
- These surprisingly identical gene sequences differed in gene order.
Turnip vs Cabbage: Different mtDNA Gene Order

Evolution is manifested as the divergence in gene order.

B. oleracea
(cabbage)

B. campestris
(turnip)
Genome rearrangement

- What are the similarity blocks and how to find them?
- What is the architecture of the ancestral genome?
- What is the evolutionary scenario for transforming one genome into the other?
Reversals

Blocks represent conserved genes (e.g. pairwise reciprocal best BLAST hits)
A reversal of the brownish elements introduces two breakpoints (disruptions in order).
### Types of rearrangements

1. **Reversal:** \(1 \ 2 \ 3 \ 4 \ 5 \ 6 \rightarrow 1 \ 2 \ -5 \ -4 \ -3 \ 6\)

2. **Translocation:** \(1 \ 2 \ 3, \ 4 \ 5 \ 6 \rightarrow 1 \ 2 \ 6, \ 4 \ 5 \ 3\)

3. **Fusion or Fission:** \(1 \ 2 \ 3 \ 4, \ 5 \ 6 \leftrightarrow 1 \ 2 \ 3 \ 4 \ 5 \ 6\)
Gene order is represented by a permutation $p$: 

$$p = p_1, \ldots, p_i, p_{i+1}, \ldots, p_{j-1}, p_j, \ldots, p_n$$

$$r(i, j) \downarrow$$

$$p = p_1, \ldots, p_j, p_{j-1}, \ldots, p_{i+1}, p_i, \ldots, p_n$$
Reversal distance problem

**Goal:** Given two permutations, find the shortest series of reversals that transforms one into another.

**Input:** Permutations $p$ and $s$

**Output:** A series of reversals $r_1, \ldots, r_t$ transforming $p$ into $s$, such that $t$ is minimum.
Sorting by reversals problem

**Goal:** Given a permutation, find a shortest series of reversals that transforms it into the identity permutation \((12 \ldots n)\)

**Input:** Permutation \(p\)

**Output:** A series of reversals \(r_1, \ldots, r_t\) transforming \(p\) into the identity permutation, such that \(t\) is minimum.
Sorting by reversals: a greedy algorithm

If sorting permutation $p = 123645$, the first three elements are already in order so it does not make any sense to break them.

This results in an idea for a greedy algorithm: increase the length of sorted prefix(p) at every step.

\[
123645 \rightarrow 123465 \rightarrow 123456
\]

Number of steps to sort the permutation of length $n$ is at most $n - 1$. 
Simple reversal sort is not optimal

A simple ReversalSort does not guarantee the smallest number of reversals and takes five steps on $p = 612345$.

However, the permutation can be sorted in ? steps. How can we do better ?
Breakpoint graph

- Represent the elements of the permutation $\pi = 2 \ 3 \ 1 \ 4 \ 6 \ 5$ as vertices in a graph (ordered along a line). Add caps such that the order is preserved.
- Connect vertices in order given by $\pi$ with black edges (black path).
- Connect vertices in order given by $1 \ 2 \ 3 \ 4 \ 5 \ 6$ with grey edges (grey path).
- Superimpose black and grey paths.
A breakpoint graph is balanced because the number of solid and dashed edges is the same.
A breakpoint graph can be decomposed into cycles that have edges with alternating patterns (solid / dashed).

What effects have reversal on these cycles?

- **Case 1**: Both edges belong to the same cycle.
- **Case 2**: Both edges belong to different cycles.
Effects of Reversals

**Case 1:** Both edges belong to the same cycle. A reversal either increases the number of cycles by one or leaves the number unchanged. A reversal is called a **proper reversal** if there's a cycle increase.

**Case 2:** Both edges belong to different cycles. A reversal decreases the number of cycles by one.
Expression for reversal distance

\[ d(\pi, \gamma) \geq n + 1 - c(\pi, \gamma) = \tilde{d}(\pi, \gamma) \]

where \( n + 1 \) is the number of breakpoints and \( c \) is the number of cycles. The equality holds for most biological systems.
Signed permutations

- Genes are directed fragments of DNA and we represent a genome by a signed permutation.
- If genes are in the same position but their orientations are different, they do not have the equivalent gene order.
- For example, these two permutations have the same order, but each gene’s orientation is the reverse; therefore, they are not equivalent gene sequences.
Basic sorting

As usual, we will assume that $\pi$ is framed by 0 and $n + 1$, and that those extra elements are always positive:

$$\pi = (0 \, \pi_1 \, \pi_2 \ldots \, \pi_n \, n + 1)$$

An oriented pair $(\pi_i, \pi_j)$ is a pair of consecutive integers, that is $|\pi_i| - |\pi_j| = \pm 1$, with opposite signs.
Oriented pairs

Example

\[(0 \ 3 \ 1 \ 6 \ 5 \ -2 \ 4 \ 7)\]
\[(0 \ 3 \ 1 \ \underline{6} \ 5 \ -2 \ 4 \ 7)\] # pair (1,-2) induces reversal
\[(0 \ 3 \ 1 \ 2 \ -5 \ -6 \ 4 \ 7)\]

In general, the reversals by an oriented pair will be:

\[\rho(i, j - 1), \text{ if } \pi_i + \pi_j = +1\]

\[\rho(i + 1, j), \text{ if } \pi_i + \pi_j = -1\]
The score of an (oriented) reversal is defined as the number of oriented pairs in the resulting permutation.

**Example**

\[(0 \ 3 \ 1 \ 6 \ 5 \ -2 \ 4 \ 7)\]

\[(0 \ -5 \ -6 \ -1 \ -3 \ -2 \ 4 \ 7)\] scores 4 !
First sorting strategy

**Algorithm 1:** As long as $\pi$ has an oriented pair, choose the oriented reversal that has maximal score.

**Example**

Step 1: Two oriented pairs $(1,-2)$ and $(3,-2)$ of permutation $(0 3 1 6 5 -2 4 7)$ score 2 and 4.

Step 2: $(0 -5 -6 -1 -3 -2 4 7)$ pairs $(0,-1)$, $(-3,4)$, $(-5,4)$ and $(-6,7)$

Step 3: $(0 -5 -6 -1 2 3 4 7)$
First sorting strategy - cont.

Example

(0  -5  -6  1  2  3  4  7)
(0  -5  -4  -3  -2  -1  6  7)
(0  1  2  3  4  5  6  7)

This elementary strategy of Algorithm 1 is sufficient to optimally sort almost all permutations that arise from biological data.
Claim 1: Algorithm 1 applies \( k \) reversals to a permutation \( \pi \), yielding a permutations \( \pi' \) such that \( d(\pi) = d(\pi') + k \).
Let’s consider know signed permutations with positive elements only. Such permutations are called *reduced* if they do not contain consecutive elements.

**Framed interval:** encompasses all integers between \( i \) and \( i+k \) belong to the interval \([i \ldots i + k]\)

Consider permutation: \((0\, 2\, 5\, 4\, 3\, 6\, 1\, 7)\). The whole permutation is a framed interval, as well as 25436 and, by circularity, 61702.
**Tough regions: Hurdle**

- **hurdle** in $\pi$ is a framed interval that contains no shorter framed interval.
- When a permutation has only one or two hurdles, one reversal is sufficient to create enough oriented pairs to completely sort the permutation with Algorithm 1.
- Two operations break hurdles: *hurdle cutting* and *hurdle merging*.
Breaking Hurdles

- **Hurdle cutting**: Reversing segment between $i$ and $i+1$ of a hurdle:

  $$\boxed{i\ldots i + 1 \ldots i + k}$$

  $$(0 \underbrace{2 4 3 1 5}_{0 2 4 3 1 5}) \rightarrow (0 \underbrace{3 \ 4 \ 2 1 5}_{3 4 2 1 5})$$

  which can be sorted in 4 reversals.

- **Hurdle merging**: Merging the end points of two hurdles.

  $$\boxed{i \ldots i + k \ldots i' \ldots i' + k'}$$

  $$(02543617) \rightarrow (02543 \underbrace{- 617}_{3 4 2 1 5})$$

  which can be sorted in 5 reversals.
Super Hurdles

A simple hurdle is a hurdle whose cutting decreases the number of hurdles. Hurdles that are not simple are called super hurdles.

Example

(0 2 5 4 3 6 1 7) → (0 2 3 4 5 6 1 7) collapses to (0 2 1 3) and has only one hurdle.
(0 2 4 3 5 1 6 8 7 9) → (0 2 1 3 5 4 6) has still two hurdles.
**Algorithm 2:** If a permutation has $2k$ hurdles, $k \geq 2$, merge any two non-consecutive hurdles. If a permutation has $2k + 1$ hurdles, $k \geq 1$, then if it has one simple hurdle, cut it; If it has none, merge two non-consecutive hurdles, or consecutive ones if $k = 1$.

Now we have to prove all the algorithms and claims.
Breakpoint graph revisited

Turn a signed permutation into an unsigned one.

- Each positive element $x$ in permutation $\pi$ is replaced by $2x - 1$ $2x$
- Each negative element $-x$ is replaced by $2x$ $2x - 1$.

**Example**

$\pi = (0 \ -1 \ 3 \ 5 \ 4 \ 6 \ -2 \ 7)$ becomes $\pi' = (0 \ 2 \ 1 \ 5 \ 6 \ 9 \ 10 \ 7 \ 8 \ 11 \ 12 \ 4 \ 3 \ 13)$
Breakpoint graph revisited

Every connected component is a cycle, which is a consequence of the fact that each vertex has exactly two incident edges. The above graph has $? \text{ cycles.}$
Terms and Definitions:

**Support** of an arc is the interval of elements of $\pi'$ between, and including, its endpoints. Two arcs **overlap** if their support intersect, without proper containment. An arc is **oriented** if its support contains an odd number of elements. An Arc overlap graph of the previous breakpoint graph is shown below:
Fact 1: A vertex has an odd degree if and only if it is oriented.

Proof: Let $2x - 1, 2x, 2x + 1,$ and $2x + 2,$ be four integers associated with the oriented pair $\pi_i, \pi_j.$ Since $\pi_i$ and $\pi_j$ have different signs, the positions $2x$ and $2 + 1$ will not have the same parity in the unsigned permutations. Thus, the interval between $2x$ and $2x + 1$ has an odd length, implying that it overlaps an odd number of other intervals.
**Fact 2:** If one performs the reversal corresponding to an oriented vertex \( v \), the effect on the overlap graph will be to complement the subgraph of \( v \) and its adjacent vertices.

**Proof:** The reversal corresponding to an oriented vertex \( v \) has the effect of collapsing the associated interval, thus \( v \) will become isolated.
Fact 3: If one performs the reversal corresponding to an oriented vertex \( v \), each vertex adjacent to \( v \) will become oriented.

Proof: Since \( v \) is oriented, it has an odd number of \( 2k + 1 \) of adjacent vertices. Let \( w \) be a vertex adjacent to \( v \), with \( j \) neighbors also adjacent to \( v \). With the reversal, \( w \) will loose \( j + 1 \) neighbors, and gain \( 2k - j \) new ones.
Fact 4: The score of the oriented reversal corresponding to an oriented vertex $\nu$ is given by:

\[ T + U - O - 1 \]

where $T$ is the total number of oriented vertices in the graph, $U$ is the number of unoriented vertices adjacent to $\nu$, and $O$ is the number of oriented vertices adjacent to $\nu$.

Proof: This follows from the preceding facts.
**Safe reversals**

A safe reversal is a reversal that does not create new unoriented components in the arc overlap graph, except isolated vertices.

**Theorem 1:** (Hannenhalli and Pevzner, 1995) Any sequence of oriented safe reversals is optimal.

**Theorem 2:** An oriented reversal of maximal score is safe.

*Proof:* Suppose that vertex \( v \) has maximal score, and that the reversal induced by \( v \) creates a new unoriented component \( C \) containing more than one vertex. At least one of the vertices in \( C \) must have been adjacent to \( v \), since the only edges affected by the reversal are those between vertices adjacent to \( v \).
Safe reversals - cont.

Let \( w \) be a vertex formerly adjacent to \( v \) and contained in \( C \), and consider the scores of \( v \) and \( w \):

\[
\text{score}(v) = T + U - O - 1
\]
\[
\text{score}(w) = T + U' - O' - 1
\]

All unoriented vertices formerly adjacent to \( v \) must have been adjacent to \( w \). Indeed, an unoriented vertex adjacent to \( v \) and not to \( w \) will become oriented, and connected to \( w \). Thus, \( U' \geq U \).
Safe reversals - cont.

All oriented vertices formerly adjacent to \( w \) must have been adjacent to \( v \). If this was not the case, an oriented vertex adjacent to \( w \) but not to \( v \) would remain oriented, again contradicting the fact that \( C \) is unoriented. Thus, \( O' \leq O \).

If both \( O' = O \) and \( U' = U \), vertices \( v \) and \( w \) have the same set of adjacent vertices, and completing the subgraph of \( v \) and its adjacent vertices will isolate both \( v \) and \( w \). Therefore, we must have that \( \text{score}(w) \geq \text{score}(v) \).
Step-by-step

- Consider signed permutation: 0 3 1 6 5 -2 4 7
- The unsigned form looks like:
  0 5 6 1 2 11 12 9 10 4 3 7 8 13
- Best reversal \( r() \) yields:
  0 5 4 10 9 12 11 2 1 6 2 7 8 13

![Diagram of gene order and signed permutations]
Where do we stand?

We have shown that Algorithm 1 produces an optimal (minimal) path of reversal operations.

If you are interested in more details, please refer to: A very elementary presentation of the Hannenhalli-Pevzner Theory by Anne Bergeron
http://citeseer.ist.psu.edu/599900.html

Maximal exposure can be obtained from: Efficient algorithms for multichromosomal genome rearrangements by Glen Tesler
http://math.ucsd.edu/~gptesler/pub_jcss.html

Instructive examples are found here:
http://www.math.tau.ac.il/~rshamir/GR/