Genomic Compression: Storage, Transmission, and Analytics

Noah M. Daniels
Bonnie Berger
Genomic data are growing exponentially
A bigger cloud?
A bigger cloud?
Still constrained by Moore
Genomic data are growing exponentially.
Redundancy in data

Unique Data

D. simulans

D. sechellia

D. melanogaster

D. yakuba

D. erecta
Outline

Compression background

Storage and transmission

Analysis
Limits to compression

Source coding theorem [Shannon, 1948]

$N$ i.i.d. random variables from source $X$
each with entropy $H(X)$
cannot be compressed into fewer than $NH(X)$
bits without loss of information

$$H(X) = - \sum_i P(x_i) \log_b P(x_i)$$
What can we do?

Approach Shannon limit as efficiently as possible (fast, lossless compression)

Be willing to throw away information (lossy compression)
Approaching Shannon limit

Huffman coding [1952]

Arithmetic coding [Rissanen 1976]

Lempel-Ziv [1977]

Burrows-Wheeler Transform [1994]
Huffman coding
Huffman coding

Variable-length prefix code
Huffman coding

Variable-length prefix code

More frequent symbols take fewer bits
Huffman coding

Variable-length prefix code

More frequent symbols take fewer bits

DNA (A, C, T, G)
Huffman coding

Variable-length prefix code

More frequent symbols take fewer bits

DNA (A,C,T,G)

2 bits/symbol: 00, 01, 10, 11
Huffman coding

Variable-length prefix code

More frequent symbols take fewer bits

DNA (A,C,T,G)

2 bits/symbol: 00, 01, 10, 11

Suppose AT bias
Huffman coding

Variable-length prefix code
More frequent symbols take fewer bits
DNA (A,C,T,G)
2 bits/symbol: 00, 01, 10, 11
Suppose AT bias

P. falciparum 20% GC
A 0.4
T 0.4
C 0.1
G 0.1
A 0.4
T 0.4
C 0.1
G 0.1
A 0.4
T 0.4
C 0.1 0.2
G 0.1
Huffman coding

Naive encoding: 2 bits/base
Huffman coding: 1.8 bits/base

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>T</th>
<th>G</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>0.4</td>
<td>0.4</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>Codeword</td>
<td>0</td>
<td>10</td>
<td>110</td>
<td>111</td>
</tr>
<tr>
<td>Bits</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Avg Bits</td>
<td>0.4</td>
<td>0.8</td>
<td>0.3</td>
<td>0.3</td>
</tr>
</tbody>
</table>
Arithmetic coding

Encode entire message as a number

<table>
<thead>
<tr>
<th>F</th>
<th>A</th>
<th>T</th>
<th>G</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interval</td>
<td>[0,0.4)</td>
<td>[0.4,0.8)</td>
<td>[0.8,0.9)</td>
<td>[0.9,1)</td>
</tr>
<tr>
<td>F</td>
<td>0.4</td>
<td>0.4</td>
<td>0.1</td>
<td>0.1</td>
</tr>
</tbody>
</table>
GATTACA
GATTACA
GATTACA
GATTACA
GATTACA
GATTACA
GATTACA
Arithmetic coding
Arithmetic coding

How do we know when the message ends?
Arithmetic coding

How do we know when the message ends?
I lied; we need a stop symbol
Arithmetic coding

How do we know when the message ends?
I lied; we need a stop symbol
Bigger problem: floating point precision
Arithmetic coding

How do we know when the message ends?
I lied; we need a stop symbol
Bigger problem: floating point precision
Integer version
Arithmetic coding

How do we know when the message ends?
I lied; we need a stop symbol
Bigger problem: floating point precision
Integer version
Rodionov & Volkov, 2007 & 2010 (p-adic arithmetic)
Arithmetic coding

How do we know when the message ends?
I lied; we need a stop symbol
Bigger problem: floating point precision
Integer version
Rodionov & Volkov, 2007 & 2010 (p-adic arithmetic)
Huffman coding is a specialized case, less likely to be optimal
Lempel-Ziv

Slide a fixed-length window along sequence
Replace already-seen patterns (of some maximum length)
Used by gzip (& others)
Lempel-Ziv

Slide a fixed-length window along sequence
Replace already-seen patterns (of some maximum length)
Used by gzip (& others)

GATTACATTA

GATTACATTA
Lempel-Ziv

Slide a fixed-length window along sequence
Replace already-seen patterns (of some maximum length)
Used by gzip (& others)

GATTACATTA
Lempel-Ziv

Slide a fixed-length window along sequence
Replace already-seen patterns (of some maximum length)
Used by gzip (& others)

GATTACATTA
Lempel-Ziv

Slide a fixed-length window along sequence
Replace already-seen patterns (of some maximum length)
Used by gzip (& others)

GATTACATTAGATTACATTA
Lempel-Ziv

Slide a fixed-length window along sequence
Replace already-seen patterns (of some maximum length)
Used by gzip (& others)

GATTACA\textcolor{blue}{CATTA}
Lempel-Ziv

Slide a fixed-length window along sequence
Replace already-seen patterns (of some maximum length)
Used by gzip (& others)

GATTACA GATTA
Lempel-Ziv

Slide a fixed-length window along sequence
Replace already-seen patterns (of some maximum length)
Used by gzip (& others)

GATTACA\textcolor{blue}{\texttt{ATTA}}
GATTAC<1,4>
Lempel-Ziv

Slide a fixed-length window along sequence
Replace already-seen patterns (of some maximum length)
Used by gzip (& others)

GATTAC  ATTA

GATTAC<1, 4>

Pointer
Lempel-Ziv

Slide a fixed-length window along sequence
Replace already-seen patterns (of some maximum length)
Used by gzip (& others)

GATTACATTA

GATTAC<1, 4>

Pointer  Length
Burrows-Wheeler Transform (BWT)
Burrows-Wheeler Transform (BWT)

block-sorting compression
Burrows-Wheeler Transform (BWT)

block-sorting compression

not itself a compressor, but a way to sort input, reversibly
Burrows-Wheeler Transform (BWT)

block-sorting compression

not itself a compressor, but a way to sort input, reversibly

Lends itself to run-length encoding
Burrows-Wheeler Transform (BWT)

block-sorting compression

not itself a compressor, but a way to sort input, reversibly

Lends itself to run-length encoding
Genomic compressors
Genomic compressors

Why not just gzip it?
Genomic compressors

Why not just gzip it?

sequence
Genomic compressors

Why not just gzip it?

sequence

quality scores (one per base)
Genomic compressors

Why not just gzip it?

sequence

quality scores (one per base)

sequence: 4-letter alphabet
Genomic compressors

Why not just gzip it?

sequence

quality scores (one per base)

sequence: 4-letter alphabet

quality scores: 40-value range
Genomic compressors

Why not just gzip it?

sequence

quality scores (one per base)

sequence: 4-letter alphabet

quality scores: 40-value range

sequence: must be lossless
Why not just gzip it?

sequence

quality scores (one per base)

sequence: 4-letter alphabet

quality scores: 40-value range

sequence: must be lossless

quality scores: lossy might be ok
Genomic compressors
Genomic compressors

Take advantage of structure
Genomic compressors

Take advantage of structure

Store differences from a reference
Genomic compressors

Take advantage of structure

Store differences from a reference

Cluster similar reads (reference-free)
Genomic compressors

Take advantage of structure

Store differences from a reference

Cluster similar reads (reference-free)

don’t have references for all species
SCALCE [Hach et al. 2012]
SCALCE [Hach et al. 2012]

Reference-free
SCALCE [Hach et al. 2012]

Reference-free

Locally Consistent Encoding
SCALCE [Hach et al. 2012]

Reference-free

Locally Consistent Encoding

Buckets reads by substrings
SCALCE [Hach et al. 2012]

Reference-free

Locally Consistent Encoding

Buckets reads by substrings improves LZ77 runtime AND compression
SCALCE [Hach et al. 2012]
Identify local maxima
SCALCE [Hach et al. 2012]

Identify local maxima
Identify local minima
Identify local maxima
Identify local minima
Partition at each side
Identify local maxima
Identify local minima
Partition at each side
Extend L and R to “core blocks”
Identify local maxima
Identify local minima
Partition at each side
Extend L and R to “core blocks”
SCALCE [Hach et al. 2012]

Identify local maxima
Identify local minima
Partition at each side
Extend L and R to “core blocks”

21312032102021312032102
SCALCE [Hach et al. 2012]

Identify local maxima
Identify local minima
Partition at each side
Extend L and R to “core blocks”

21312032102021312032102
SCALCE [Hach et al. 2012]

Identify local maxima
Identify local minima
Partition at each side
Extend L and R to “core blocks”

```
21312032102021312032102
```

```
| 213 | 12 | 03 | 2102 | 02 | 13 | 12 | 03 | 2102 |
```
SCALCE [Hach et al. 2012]

Identify local maxima
Identify local minima
Partition at each side
Extend L and R to “core blocks”

X0X

21312032102021312032102

| 213 | 12 | 03 | 2102 | 02 | 13 | 12 | 03 | 2102 |

2131, 3120, 2032, 321020, 2021, 2131, 3120, 2032, 32102
Core blocks between 8 & 20
99% of all HTS reads of length ≥50
include at least one core of length ≤14

Identify overlapping reads by core blocks
Reorder reads to favor LZ77
DeeZ [Hach, et al. 2014]
DeeZ [Hach, et al. 2014]

Given a mapping (reads to reference)
Given a mapping (reads to reference)

Partition reads into blocks according to locus
Given a mapping (reads to reference)
Partition reads into blocks according to locus
Build contig covering all reads within a block
DeeZ [Hach, et al. 2014]

Given a mapping (reads to reference)
Partition reads into blocks according to locus
Build contig covering all reads within a block
Encode locus for each read within contig
DeeZ [Hach, et al. 2014]

Given a mapping (reads to reference)
Partition reads into blocks according to locus
Build contig covering all reads within a block
Encode locus for each read within contig
Encode rare differences (sequencing errors)
DeeZ [Hach, et al. 2014]

ACGTGCTAAACGTCGT
T
TACA
G
TCTACAGA
GTCGT
C
TACA

AAC
CTCGT
CTAC

GTCTACA
TCTA
ACGTGCTAAACGTCGT
T
TACAGTCTACAGA
DeeZ [Hach, et al. 2014]

GTCGTCTACATT
CGTCGTCTACA
AACCTCGTCTAC
GTCTACA TTCTACAGA
ACGTGCTAAACGTCGTCTACA TCTACAGA
DeeZ [Hach, et al. 2014]
DeeZ [Hach, et al. 2014]
DeeZ [Hach, et al. 2014]

Tokenization of read names
DeeZ [Hach, et al. 2014]

Tokenization of read names

LZ77
DeeZ [Hach, et al. 2014]

Tokenization of read names

LZ77

Lossy QS compression from SCALCE
DeeZ [Hach, et al. 2014]

Tokenization of read names

LZ77

Lossy QS compression from SCALCE

Random access
Reference-based
– but no aligning
– statistical, generative model of reads
For RNA-seq data (but need not be)
PathEnc [Kingsford & Patro 2015]

GAUU

GAUU AGAUUG
PathEnc [Kingsford & Patro 2015]

GAUU

AUUA

GAUUAGAUUUG
PathEnc [Kingsford & Patro 2015]

GAUU
AUUA
UUAG
GAUUAGAUUG
PathEnc [Kingsford & Patro 2015]

GAUU
→ AUUA
→ UUAG
→ UAGA

GAUUUAGAGAUUG
PathEnc [Kingsford & Patro 2015]

GAUUAGAUUG

GAUU

AUUA

UUAG

UAGA

AGAU

GAUUAGAUUG
PathEnc [Kingsford & Patro 2015]

GAUU
↓
AUUA
↓
UUAG
↓
UAGA
↓
AGAU

GAUU UAGA AGAU
PathEnc [Kingsford & Patro 2015]
Path encoding in graph $G$
Path encoding in graph $G$

1 node per k-mer
Path encoding in graph $G$

- 1 node per k-mer

- edge between k-mers $(u, v)$ if $v$ follows $u$
Path encoding in graph G

1 node per k-mer

dge between k-mers \((u,v)\) if \(v\) follows \(u\)

Each read encoded as a path in G
PathEnc [Kingsford & Patro 2015]
PathEnc [Kingsford & Patro 2015]

1st node of each read path (read head)
PathEnc [Kingsford & Patro 2015]

1st node of each read path (read head)

4-ary tree of depth $k$
PathEnc [Kingsford & Patro 2015]

1st node of each read path (read head)

4-ary tree of depth k

edges removed for nonexistent k-mers
PathEnc [Kingsford & Patro 2015]

1st node of each read path (read head)

4-ary tree of depth k

edges removed for nonexistent k-mers

traverse in fixed order, emitting 1 for edge
PathEnc [Kingsford & Patro 2015]

1st node of each read path (read head)

4-ary tree of depth k

edges removed for nonexistent k-mers

traverse in fixed order, emitting 1 for edge

resulting bit string gzipped
PathEnc [Kingsford & Patro 2015]
PathEnc [Kingsford & Patro 2015]

remaining nodes (read tails)
remaining nodes (read tails)
arithmetic coding
PathEnc [Kingsford & Patro 2015]

remaining nodes (read tails)

arithmetic coding

probability distribution per node in G
PathEnc [Kingsford & Patro 2015]
MINCE [Patro & Kingsford 2015]
MINCE [Patro & Kingsford 2015]

Reference-free
MINCE [Patro & Kingsford 2015]

Reference-free

Buckets reads based on $k$-mers ($k=15$)
MINCE [Patro & Kingsford 2015]

Reference-free

Buckets reads based on $k$-mers ($k=15$)

Replace common $k$-mer with pointer
Reference-free

Buckets reads based on $k$-mers ($k=15$)

Replace common $k$-mer with pointer

Reorder reads within each bucket
Reference-free

Buckets reads based on $k$-mers ($k=15$)

Replace common $k$-mer with pointer

Reorder reads within each bucket

This boosts Izip performance
Split-swap read transformation

Suffix of bucket label

offset = 4

r = ATCAAGCCCATAGGT

common bucket label

enc(r) = TAGGTATCA, 4
The Quality Score Problem
The Quality Score Problem
The Quality Score Problem
The Quality Score Problem

Given reads from a genome, can we **efficiently** compress quality scores while maintaining or even improving accuracy?
Quality scores in SNP-calling
Quality scores in SNP-calling

**Raw Reads**

- FASTQ file

**Compressed Quality**

- Smoothed FASTQ file

**Mapped Reads**

- Bowtie 2 or BWA

**Variant Calls**

- GATK or Samtools

<table>
<thead>
<tr>
<th>Variant Calls</th>
<th>Quality Scores</th>
</tr>
</thead>
<tbody>
<tr>
<td>102</td>
<td>452</td>
</tr>
<tr>
<td>146</td>
<td>713</td>
</tr>
<tr>
<td>278</td>
<td>843</td>
</tr>
<tr>
<td>343</td>
<td>901</td>
</tr>
<tr>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>G</td>
<td>G</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Quality score compressors
Quality score compressors

Greater dynamic range than sequence
Quality score compressors

Greater dynamic range than sequence

Can be lossless or lossy
Quality score compressors

Greater dynamic range than sequence

Can be lossless or lossy

Focus here on lossy
8-binning [Illumina]

Universally collapse dynamic range 40 => 8
Small effect on SNP calling error
Available in CRAM

<table>
<thead>
<tr>
<th>QS bins</th>
<th>New value</th>
</tr>
</thead>
<tbody>
<tr>
<td>N (no call)</td>
<td>N (no call)</td>
</tr>
<tr>
<td>2-9</td>
<td>6</td>
</tr>
<tr>
<td>10-19</td>
<td>15</td>
</tr>
<tr>
<td>20-24</td>
<td>22</td>
</tr>
<tr>
<td>25-29</td>
<td>27</td>
</tr>
<tr>
<td>30-34</td>
<td>33</td>
</tr>
<tr>
<td>35-39</td>
<td>37</td>
</tr>
<tr>
<td>≥40</td>
<td>40</td>
</tr>
</tbody>
</table>
SCALCE [Hach et al. 2012]
SCALCE [Hach et al. 2012]

Lossy compression of quality scores
SCALCE [Hach et al. 2012]

Lossy compression of quality scores

Neighboring QSs often similar
Lossy compression of quality scores

Neighboring QSs often similar

Reduce dynamic range
Lossy compression of quality scores

Neighboring QSs often similar

Reduce dynamic range

Arithmetic coding
Lossy compression of quality scores

Neighboring QSs often similar

Reduce dynamic range

Arithmetic coding

Small loss (<0.1%) of SNP calling accuracy
QVZ [Malysa, et al. 2015]
Based on rate-distortion theory

Discard as little information as possible for a desired bit rate

Key insight: neighboring QSs are likely to be correlated

Illumina reads often have lower QSs at end
QVZ [Malysa, et al. 2015]
1. Compute the empirical transition probabilities of a Markov-1 model
1. Compute the empirical transition probabilities of a Markov-1 model

2. Construct a codebook using the Lloyd-Max algorithm
1. Compute the empirical transition probabilities of a Markov-1 model

2. Construct a codebook using the Lloyd-Max algorithm

3. Quantize the input using the codebook, use arithmetic encoder
QVZ [Malysa, et al. 2015]
Quartz [Yu, et al. 2015]

Preprocessing
Quartz [Yu, et al. 2015]

Preprocessing

Compression
Quartz [Yu, et al. 2015]

Preprocessing

Compression
Quartz [Yu, et al. 2015]

Preprocessing

Compression
Quartz [Yu, et al. 2015]

Preprocessing

Compression
Quartz [Yu, et al. 2015]

Preprocessing

Compression
Approximate k-mer search
Naïve approaches
Approximate k-mer search
Naïve approaches

• Need to quickly find all $3k$ Hamming neighbors of each k-mer from the read in the dictionary to identify mis-matched bases.
Approximate k-mer search
Naïve approaches

• Need to quickly find all $3k$ Hamming neighbors of each k-mer from the read in the dictionary to identify mis-matched bases.

• Naïve approaches
  – Sorted list
    • Memory efficient but binary search is CPU and cache inefficient
  – Hash tables
    • Faster CPU-wise, but memory and cache inefficient
Approximate k-mer search
Locality sensitive hashing

An \((R, cR, P_1, P_2)\)-sensitive LSH family \(F\) of hash functions \(h : M \rightarrow S\) is defined if

\[\forall p, q \in M, \text{ a uniformly random } h \in F \text{ satisfies:}\]
\[\begin{align*}
\text{if } \|p - q\| \leq R \text{ then } & \mathbb{P}(h(p) = h(q)) \geq P_1 \\
\text{if } \|p - q\| \geq cR \text{ then } & \mathbb{P}(h(p) = h(q)) \leq P_2
\end{align*}\]

Project \(k\)-mers onto random \(\frac{k}{2}\)-mers, forming a \((1, c, \frac{1}{2}, 2^{c-1})\)-sensitive family of hash functions under the Hamming metric
Approximate k-mer search
Double hashing for fun and profit
Approximate k-mer search
Double hashing for fun and profit

• Notice that each $h : M \rightarrow S$ comes with an orthogonal projection $h': M \rightarrow S$
Approximate k-mer search
Double hashing for fun and profit

• Notice that each \( h : M \to S \) comes with an orthogonal projection \( h' : M \to S \).

• Also, if \( \|p, q\| \leq 1 \) then by counting, at least one of \( h(p) = h(q) \) or \( h'(p) = h'(q) \) must hold.
Approximate k-mer search
Double hashing for fun and profit

• Notice that each $h : M \rightarrow S$ comes with an orthogonal projection $h' : M \rightarrow S$.

• Also, if $\|p, q\| \leq 1$ then by counting, at least one of $h(p) = h(q)$ or $h'(p) = h'(q)$ must hold.

• Thus, by double hashing, all Hamming neighbors of a $k$-mer can be found by looking in just two hash buckets.

  – Better cache-efficiency

  – Cheating by carefully choosing the projection and sorting the buckets gives also processor-efficiency.
Approximate k-mer search
Fast retrieval of Hamming neighbors
Approximate k-mer search
Fast retrieval of Hamming neighbors
Approximate k-mer search
Fast retrieval of Hamming neighbors

Queries

Dictionary

Clustered by front half of k-mer

C G G C A G A C
C G G C C G A C
C G G C A G T C
Approximate k-mer search
Fast retrieval of Hamming neighbors

Clusters by front half of k-mer:
- C G G C A G A C
- C G G C C G A C
- C G G C A G T C

Clusters by back half of k-mer:
- A G G C A G A C
- G G G C A G A C
- C C G C A G A C
- T A T A A G A C
Approximate k-mer search
Fast retrieval of Hamming neighbors

Queries

Dictionary

Clustered by front half of k-mer
C G G C A G A C
C G G C C G A C
C G G C A G T C

Clustered by back half of k-mer
A G G C A G A C
G G G C A G A C
C C G C A G A C
T A T A A G A C
△ = C G G C A G A C
Approximate k-mer search
Fast retrieval of Hamming neighbors

Queries

Dictionary

Clustered by front half of k-mer
\[ \begin{align*}
C & \ G & \ G & \ C & \ A & \ G & \ A & \ C \\
C & \ G & \ G & \ C & \ C & \ G & \ A & \ C \\
C & \ G & \ G & \ C & \ A & \ G & \ T & \ C \\
\end{align*} \]

Clustered by back half of k-mer
\[ \begin{align*}
A & \ G & \ G & \ C & \ A & \ G & \ A & \ C \\
G & \ G & \ G & \ C & \ A & \ G & \ A & \ C \\
C & \ C & \ G & \ C & \ A & \ G & \ A & \ C \\
T & \ A & \ T & \ A & \ A & \ G & \ A & \ C \\
\end{align*} \]= \[ \begin{align*}
C & \ G & \ G & \ C & \ A & \ G & \ A & \ C \\
\end{align*} \]
## Result Highlights

### Comparison with other methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Bits/Q</th>
<th>Time (s)</th>
<th>Area Under ROC Curve</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncompressed</td>
<td>8</td>
<td>N/A</td>
<td>0.8254</td>
</tr>
<tr>
<td>Quartz</td>
<td>0.3564</td>
<td>2,696</td>
<td>0.8288</td>
</tr>
<tr>
<td>QualComp</td>
<td>0.5940</td>
<td>33,316</td>
<td>0.8053</td>
</tr>
<tr>
<td>Janin et al.</td>
<td>0.5376</td>
<td>164,702</td>
<td>0.8019</td>
</tr>
</tbody>
</table>

![ROC Curve](image)

- **False Positive Rate**
- **True Positive Rate**
## Result Highlights

### Comparison with other methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Bits/Q</th>
<th>Time (s)</th>
<th>Area Under ROC Curve</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncompressed</td>
<td>8</td>
<td>N/A</td>
<td>0.8254</td>
</tr>
<tr>
<td>Quartz</td>
<td>0.3564</td>
<td>2,696</td>
<td>0.8288</td>
</tr>
<tr>
<td>QualComp</td>
<td>0.5940</td>
<td>33,316</td>
<td>0.8053</td>
</tr>
<tr>
<td>Janin et al.</td>
<td>0.5376</td>
<td>164,702</td>
<td>0.8019</td>
</tr>
</tbody>
</table>

Quartz is orders of magnitude faster.
A word on k-mers

MINCE: 15-mers optimal (8-mer labels!)
using labels as search heuristic; don’t want too many

Quartz: 32-mers optimal
in a genome, most 8-mers may exist; most 32-mers will not
and they tend to be unique

<table>
<thead>
<tr>
<th></th>
<th>k=16</th>
<th>k=32</th>
</tr>
</thead>
<tbody>
<tr>
<td>unique</td>
<td>21.8%</td>
<td>85.7%</td>
</tr>
<tr>
<td>unique at Hamming 1</td>
<td>0.0008%</td>
<td>79.3%</td>
</tr>
</tbody>
</table>
Uniqueness of k-mers (hg19)
Why not longer k-mers?

Want to ensure at most 1 sequencing error per k-mer

Assume independence of errors
Error rate of $p$

Likelihood of $l$ errors

$\binom{k}{l}(1 - p)^{k-l}p^l$

1% error rate

$k=32 \quad l \geq 2 \quad 2\%$

$k=64 \quad l \geq 2 \quad 13\%$

2% error rate

$k=32 \quad l \geq 2 \quad 13\%$

$k=64 \quad l \geq 2 \quad 36\%$
Compression for speed

![Graph showing the increase in computing power and genomic data over time.](image-url)
Compressive genomics

caBLAST [Loh, et al. 2012]

caBLASTP [Daniels, et al. 2013]

BLAST uses seed-and-extend

but must extend on many fruitless seeds

Use compression to reduce the search space
How does this compression work?
How does this compression work?

STAQEPKSAEDSLRARD
How does this compression work?

STAQEPKSAEDSLRARD

Coarse Database

STAQEPKSAEDSLRARD
How does this compression work?

Seed Table

```
... STAP STAQ STAR STAS ...
```

Seed Table → STAQEPKSAEDSLRARD → Coarse Database

Coarse Database

STAQEPKSAEDSLRARD
How does this compression work?

Seed Table

```
STAP STAQ
STAR STAS
```

Coarse Database

```
STAQEPKSAEDSLRARD
```

```
LQSTAQEPKSAEQRDSVNARDRQRNVIIIAQE
```
How does this compression work?

Seed Table

- STAP STAQ
- STAR STAS

STAQEPKSAEDSLRARD

Coarse Database

STAQEPKSAEDSLRARD

LQST

AQEPKSAEQRSVNARQRNVIIAQE
How does this compression work?

Seed Table

- STAP STA Q
- STAR STAS

STAQEPKSAEDSLRARD

Coarse Database

STAQEPKSAEDSLRARD

LQSTAQE PKSAE QRD SVN ARDQRN VI IAQE
How does this compression work?
How does this compression work?

<table>
<thead>
<tr>
<th>Seed Table</th>
<th>Coarse Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>STAQEPKSAE...</td>
</tr>
<tr>
<td>STAP</td>
<td>STAQEPKSAE...</td>
</tr>
<tr>
<td>STAQ</td>
<td>STAQEPKSAE...</td>
</tr>
<tr>
<td>STAR</td>
<td>STAQEPKSAE...</td>
</tr>
<tr>
<td>STAS</td>
<td>STAQEPKSAE...</td>
</tr>
<tr>
<td>...</td>
<td>STAQEPKSAE...</td>
</tr>
</tbody>
</table>

STAQEPKSAE

DSLRA

LQSTAQEPKSAEQRDSVNARDRQRNVI I AQE
How does this compression work?

Seed Table

Coarse Database

```
STAP STAQ STAR STAS ...

STAP STAQPKSAEDSLRARD...

STAQEPKSAEDSLRARD

LQSTAQEPKSAEQRDSVNARDRQRNVIIIAQE
```
How does this compression work?

Seed Table

---
STAP STAQ
STAR STAS
---

STAQEPKSAEDSLRARD

---

Coarse Database

STAQEPKSAEDSLRARD

LQSTAQEPKSAEQRDSVNARDRQRNVIIIAQE

QRDSVNARD

LQSTAQEPKSAEQRDSVNARDRQRNVIIIAQE

---

DSLRARD

---
How does this compression work?

Seed Table

<table>
<thead>
<tr>
<th>STAQ</th>
<th>STAQ</th>
<th>STAQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>STAQ</td>
<td>STAQ</td>
<td>STAQ</td>
</tr>
</tbody>
</table>

---

Coarse Database

[Diagram showing compression process]

Example: "STAQEPKSAE DSLRARD"

LQ QR VN
How does this compression work?

Seed Table

Coarse Database

---

STAQEPKSASEDSLRARD

QRDSVNARD

LQ

STAQEPKSASERQRDSVNARDRQRNVIIAQE
How does this compression work?
How does this compression work?

Lossless compression!
How does compressed search work?
How does compressed search work?

Query Sequence

STAQEPKSAEDSVNARD

Coarse Database

STAQEPKSAEDSPSSNG
VNARD
IGGVAYLREQYESVSK
How does compressed search work?

Diagram:

- **Query Sequence**: STAQPSEKSAEDSVNARD
- **Coarse Database**: STAQPSEKSAEDSPSSNG VNARD IGGVAYREQYESVK
- **Coarse Hits**: STAQPSEKSAEDSPSSNG VNARD

 coherent search
How does compressed search work?

**Query Sequence**

STAQEPKSAEDSVNARD

**Coarse Database**

STAQEPKSAEDSPSSNG VNARD IGGVAYLREQFYESVK

**Coarse Hits**

STAQEPKSAEDSPSSNG VNARD

**Candidates**

STAQEPKSAEDSPSSNG STAQEPKSAEDSVNARD

---

coarse search  reconstruction
How does compressed search work?
Simulated data growth

Daniels, Gallant, Peng, Baym, Cowen, Berger 2013
Simulated data growth

Daniels, Gallant, Peng, Baym, Cowen, Berger 2013
Real data growth on a protein database

Date of protein database

Search time (seconds)

Compressive-accelerated BLASTP
BLASTP
Metagenomics
Problem: Given reads from a microbiome, match each read with similar proteins.

BLASTX [Altschul, et al. 1990]
RapSearch2 [Zhao, et al. 2012]
DIAMOND [Buchfink, et al. 2015]
Metagenomics

BLASTX [Altschul, et al. 1990]
RapSearch2 [Zhao, et al. 2012]
DIAMOND [Buchfink, et al. 2015]
Metagenomics

Metagenomic Reads

>read1
ACGTGGCTATCAACTCGCTAACTAA
>read2
ACGTGGCTATCAACTCGCTAACTAA
>read3
ACGTGGCTATCAACTCGCTAACTAT
...
>readk
TCGTCAACTACATTACATTTACAG
>readk+1
TCGTCAACTACATTACAAATACAG
...
>readm
GCTCGGACTATATATAGGCCTAGAA
...

BLASTX [Altschul, et al. 1990]
RapSearch2 [Zhao, et al. 2012]
DIAMOND [Buchfink, et al. 2015]
Metagenomics

**Metagenomic Reads**

```
>read1
ACGTGGCTATCAACTCGCTAATA
>read2
ACGTGGCTATCAACTCGCTAATA
>read3
ACGTGGCTATCAACTCGCTAATAT
...
>readk
TCGTCGAACTACATTACATTACAG
>readk+1
TCGTCGAACTACATTACAAATACAG
...
>readm
GCTCGGACTATATATAGGCCCTAGAA
...
```

**Translated ORFs**

```
>read1-1
TWLSTR
>read1-2
RGYQLAN
>read1-3
VAINSLT
>read1-r1
LVSELIAT
>read1-r2
LAV
>read1-r3
RVDSH
>read2-1
SSNYITFT
>read2-2
RRTTLHLQ
>read2-3
VIELHYIY
>read2-r1
CSSST
>read2-r2
CKCNVVR
>read2-r3
VNV
...
```
### Metagenomics

<table>
<thead>
<tr>
<th>Metagenomic Reads</th>
<th>Translated ORFs</th>
<th>Protein Database (NR)</th>
</tr>
</thead>
</table>
| >read1
ACGTGGCTATCAAATCGCTAATAA
>read2
ACGTGGCTATCAAATCGCTAATAA
>read3
ACGTGGCTATCAAATCGCTAATAA
... |
| >read1–1
TWLSTR
>read1–2
RGYQLAN
>read1–3
VAINSILT
>read1–r1
LVSELJAT
>read1–r2
LAS
>read1–r3
RVDSH
>read2–1
SSNYITFT
>read2–2
RRTTLHLQ
>read2–3
VELHYIY
>read2–r1
CSST
>read2–r2
CKCNYVRR
>read2–r3
VNV
... |
| >protein 1
MRVLVINSGSSSIKYQIEM
>protein 2
EILGKKLEELKIITCHIGNASVAAVKY
>protein 3
LKKLLESSGCRVRYGNILIG
... |

BLASTX [Altschul, et al. 1990]

RapSearch2 [Zhao, et al. 2012]

DIAMOND [Buchfink, et al. 2015]
MICA [Yu, Daniels, et al. 2015]

American gut microbiome project NGS reads, searching NCBI NR database

Running time (minutes)

- **BLASTX**: 58,215m
- **DIAMOND**: 54m
- **MICA-BLASTX**: 21.9m
- **MICA-DIAMOND**: 15.6m

- >90% recall vs. BLASTX (no loss vs. DIAMOND)
- ~3700x BLASTX
- ~3.5x DIAMOND
Compression = clustering
Compression = clustering

Cluster similar entities
Compression = clustering

Cluster similar entities

or sub-entities
Compression = clustering

Cluster similar entities

or sub-entities

Only store the common parts once
Compression = clustering

Cluster similar entities

or sub-entities

Only store the common parts once

Only analyze the common parts once
Triangle inequality allows this
Triangle inequality allows this cluster radius $r_c$
Triangle inequality allows this cluster radius $r_c$
Triangle inequality allows this

cluster radius $r_c$
search radius $r$
Triangle inequality allows this

cluster radius $r_c$
search radius $r$
Triangle inequality allows this

cluster radius $r_c$
search radius $r$
Triangle inequality allows this

cluster radius $r_c$

search radius $r$
Triangle inequality allows this

circle radius $r_c$
space radius $r$

Triangle inequality: $r + r_c > r$
Large-scale NGS Analysis
Large-scale NGS Analysis

Raw Reads

Bowtie 2 or BWA

Mapped Reads
to reference

GATK or Samtools

Variant Calls

102 ..... A
146 ..... C
278 ..... G
343 ..... T
452 ..... A
713 ..... C
843 ..... G
901 ..... T
Large-scale NGS Analysis

Raw Reads

Mapped Reads
to reference

Variant Calls
102 ..... A
146 ..... C
278 ..... G
343 ..... T
452 ..... A
713 ..... C
843 ..... G
901 ..... T

Bowtie 2
or BWA

GATK or
Samtools
Large-scale NGS Analysis

- **Raw Reads**
- **Mapped Reads**
  - to reference
- **Variant Calls**
  - 102 .... A
  - 146 .... C
  - 278 .... G
  - 343 .... T
  - 452 .... A
  - 713 .... C
  - 843 .... G
  - 901 .... T

Methods:
- Bowtie 2 or BWA
- GATK or Samtools
NGS-Mapping in Personal Genomics Era

Reads | Reference Loci
NGS-Mapping in Personal Genomics Era
NGS-Mapping in Personal Genomics Era

Key Intuition: Reads have lots of redundancy!
NGS-Mapping in Personal Genomics Era

Reads

Reference Loci

Best Mapping
NGS-Mapping in Personal Genomics Era

Reads

Reference Loci

All Mapping
NGS-Mapping in Personal Genomics Era

Reads

Reference Loci

All Mapping
CORA [Yorukoglu, et al. accepted for publication]
CORA [Yorukoglu, et al. accepted for publication]
Reads  Reference Loci
CORA [Yorukoglu, et al. accepted for publication]
CORA [Yorukoglu, et al. accepted for publication]
CORA [Yorukoglu, et al. accepted for publication]

Reads

Reference Loci
CORA [Yorukoglu, et al. accepted for publication]
Result Highlights

CORA: 1000 Genomes Project
Result Highlights
CORA: 1000 Genomes Project

All Mapping (with Indels)
Result Highlights
CORA: 1000 Genomes Project

All Mapping (with Indels)
Result Highlights
CORA: 1000 Genomes Project

All Mapping (with Indels)

All Mapping (Hamming)
Result Highlights
CORA: 1000 Genomes Project

All Mapping (with Indels)

All Mapping (Hamming)
Result Highlights
CORA: 1000 Genomes Project

All Mapping (with Indels)

Runtime (hours)

FIN1 (~4x)
FIN2 (~8x)
FIN4 (~16x)

All Mapping (Hamming)
Result Highlights
CORA: 1000 Genomes Project

All Mapping (with Indels)

Scales sublinearly with very high sensitivity on real genomic data
Data constrained by physical process
Grid search
Grid search
Grid search
Grid search
Grid search
The search tree
The search tree

course search
The search tree

coarse search
The search tree

fine search
The search tree

\[ O(\text{coarse} + \text{fine}) \]
The search tree

\[ O(\ k \ + \ fine) \]
Metric entropy
Metric entropy

\[ N_{r_c}^{ent}(D) := \]
Metric entropy

\[ N_{r_c}^{ent}(D) := \text{largest number of points } x_1, \ldots, x_n \in D \]
Metric entropy

\[ N_{rc}^{ent}(D) := \text{largest number of points } x_1, \ldots, x_n \in D \]

s.t. \[ \|x_i - x_j\| \geq \rho, \forall i \neq j \]
Metric entropy

\[ N_{r_c}^{ent}(D) := \text{largest number of points } x_1, \ldots, x_n \in D \]

s.t. \[ \|x_i - x_j\| \geq \rho, \forall i \neq j \] for some radius \( \rho \)
Metric entropy

\[ N_{rc}^{ent}(D) := \text{largest number of points } x_1, \ldots, x_n \in D \]

s.t. \[ \| x_i - x_j \| \geq \rho, \forall i \neq j \] for some radius \( \rho \)

\[ \dim_{\text{Minkowski}}(D) := \lim_{\rho \to 0} \frac{\log N_{\rho}(D)}{\log 1/\rho} \]
Discrete fractal dimension
Discrete fractal dimension

\[ d := \arg \max_{d^*} \left\{ N_\rho(D) \propto \rho^{d^*} \mid \rho \in [\rho_1, \rho_2] \right\} \]

\(d\) is the dimension that maximizes the metric entropy
Discrete fractal dimension

\[ d := \arg \max_{d^*} \left\{ N_\rho(D) \propto \rho^{d^*} \mid \rho \in [\rho_1, \rho_2] \right\} \]

\( d \) is the dimension that maximizes the metric entropy

\[ \rho = r_c \]
Discrete fractal dimension

\[ d := \arg \max_{d^*} \{ N_{\rho}(D) \propto \rho^{d^*} \mid \rho \in [\rho_1, \rho_2] \} \]

\( d \) is the dimension that maximizes the metric entropy

\[ \rho = r_c \]

\( k \) clusters
Discrete fractal dimension

\[ d := \arg \max_{d^*} \{ N_\rho(D) \propto \rho^{d^*} \mid \rho \in [\rho_1, \rho_2] \} \]

\( d \) is the dimension that maximizes the metric entropy

\[ \rho = r_c \]

\( k \) clusters \hspace{1cm} \( k \leq N_{r_c}^{\text{ent}}(D) \)
Discrete fractal dimension

\[ d := \arg \max_{d^*} \left\{ N_\rho(D) \propto \rho^{d^*} \mid \rho \in [\rho_1, \rho_2] \right\} \]

\( d \) is the dimension that maximizes the metric entropy

\[ \rho = r_c \]

\( k \) clusters

\[ k \leq N_{r_c}^{ent}(D) \]

\[ O(k + \text{fine}) \]
Discrete fractal dimension

\[ d := \arg \max_{d^*} \left\{ N_\rho(D) \propto \rho^{d^*} \mid \rho \in [\rho_1, \rho_2] \right\} \]

\( d \) is the dimension that maximizes the metric entropy

\[ \rho = r_c \]

\( k \) clusters

\[ k \leq N_{r_c}^{ent}(D) \]

\[ O(N_{r_c}^{ent}(D) + \text{fine}) \]
Fine search bounds

$O(N_{rc}^{ent} (D) + |F|)$
Fine search bounds

\[ O(N_{r_c}^{ent}(D) + |F|) \quad F = \bigcup_{c \in B_C(q,r+r_c)} c \]
Fine search bounds

\[ O(N_{r_c}^{ent}(D) + |F|) \quad F = \bigcup_{c \in B_C(q, r+r_c)} c \]

\[ F \subset B_D(q, r + 2r_c) \]
Fine search bounds

\[ O(N_{r_c}^{ent}(D) + |F|) \]

\[ F = \bigcup_{c \in B_C(q, r + r_c)} c \]

\[ F \subseteq B_D(q, r + 2r_c) \]

\[ |F| \leq |B_D(q, r + 2r_c)| \]
Fine search bounds

\[ O(N_{r_c}^{ent}(D) + |F|) \quad F = \bigcup_{c \in B_C(q,r+r_c)} c \]

\[ F \subset B_D(q,r + 2r_c) \]

\[ |F| \leq |B_D(q,r + 2r_c)| \sim |B_D(q,r)| \left( \frac{r + 2r_c}{r} \right)^d \]
Fine search bounds

\[ O\left(N_{r_c}^{\text{ent}}(D) + |F|\right) \quad F = \bigcup_{c \in B_C(q, r + r_c)} c \]

\[ F \subset B_D(q, r + 2r_c) \]

\[ |F| \leq |B_D(q, r + 2r_c)| \sim |B_D(q, r)| \left(\frac{r + 2r_c}{r}\right)^d \]

Doubling search radius increases hits by at most \(2^d\)
Fine search bounds

\[ O(\sum_{r_c}^{|F|}) \]

\[ F = \bigcup_{c \in B_C(q, r+r_c)} c \]

\[ F \subset B_D(q, r + 2r_c) \]

\[ |F| \leq |B_D(q, r + 2r_c)| \sim |B_D(q, r)| \left( \frac{r + 2r_c}{r} \right)^d \]

Doubling search radius increases hits by at most \(2^d\)

\[ O \left( \sum_{r_c}^{|F|} \right) \left( \frac{r + 2r_c}{r} \right)^d \]
Entropy-scaling data structure

\[ O \left( N_{r_c}^{\text{ent}}(D) + |B_D(q, r)| \left( \frac{r + 2r_c}{r} \right)^d \right) \]
Entropy-scaling data structure

\[
O \left( N_{r_c}^{ent}(D) + |B_D(q, r)| \left( \frac{r + 2r_c}{r} \right)^d \right)
\]
Entropy-scaling data structure

\[ O \left( N_{r_c}^{ent}(D) + |B_D(q, r)| \left( \frac{r + 2r_c}{r} \right)^d \right) \]
Entropy-scaling data structure

\[ O \left( N_{r_c}^{ent}(D) + |B_D(q, r)| \left( \frac{r + 2r_c}{r} \right)^d \right) \]
Entropy-scaling data structure

\[ O \left( N_{r_c}^{ent}(D) + |B_D(q, r)| \left( \frac{r + 2r_c}{r} \right)^d \right) \]
Summary
Summary

Lossless compression bounded by Shannon
Summary

Lossless compression bounded by Shannon
Lossy compression can go further
Lossless compression bounded by Shannon
Lossy compression can go further
Clever preprocessing of reads can boost standard compressors
Summary

Lossless compression bounded by Shannon
Lossy compression can go further
Clever preprocessing of reads can boost standard compressors
All comes down to structure of the data:
Summary

Lossless compression bounded by Shannon

Lossy compression can go further

Clever preprocessing of reads can boost standard compressors

All comes down to *structure of the data*:
minimize bits/base
Summary

Lossless compression bounded by Shannon
Lossy compression can go further
Clever preprocessing of reads can boost standard compressors
All comes down to *structure of the data*:
minimize bits/base
identify common substrings
Summary

Lossless compression bounded by Shannon
Lossy compression can go further
Clever preprocessing of reads can boost standard compressors
All comes down to structure of the data:
minimize bits/base
identify common substrings
find correlations among quality scores
Going further
Going further

Many more challenges:
Going further

Many more challenges:

New sequencing technologies
Going further

Many more challenges:

New sequencing technologies

longer, more error-prone reads
Going further

Many more challenges:

New sequencing technologies

longer, more error-prone reads

different error models
Going further

Many more challenges:

New sequencing technologies

longer, more error-prone reads

different error models

Can we get the benefits of compressive acceleration with a more general model