STATISTICAL METHODS IN BIOINFORMATICS

Analysis of RNA sequencing data

Stefan E Seemann
sse@sund.ku.dk

Center for non-coding RNA in Technology and Heath (http://rth.dk)
Dep. of Public Health, SUND, UCPH

19 November 2026

Programme

Nov 19	08:30 - 11:15	Stefan	Raw sequencing data to count tables
	08:30 - 10:00		*Lecture+Exercise:* Alignment methods
	10:15 - 10:45		*Lecture:* Speed up of local sequence alignments
	10:45 - 11:15		*Lecture:* From Raw Reads to Expression Matrix
	12:15 - 15:00	Nadya	STRING & biological networks

Nov 20	08:30 - 11:15	Stefan	Unsupervised and supervised data exploration
	08:30 - 09:30		*Lecture:* Unsupervised+supervised data exploration
	09:45 - 11:15		*Exercise:* Expression analysis in R
	12:15 - 15:00	Nadya	Cytoscape

Why sequencing?

- Assemble the genome and transcriptome of a species
- Find genomic variation in a population
- Find genomic and transcriptomic associations with diseases and phenotypes
- Find organisms in environmental sample $\rightarrow$ metagenomics and -transcriptomics
- Identify potential drug targets $\rightarrow$ personalized medicine
- Tracking of virus variants and mutations $\rightarrow$ vaccine development

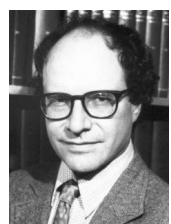
$\Rightarrow$ One of the biggest hammers in the tool box right now – always ask could this experiment be done using sequencing instead?

What is your application?

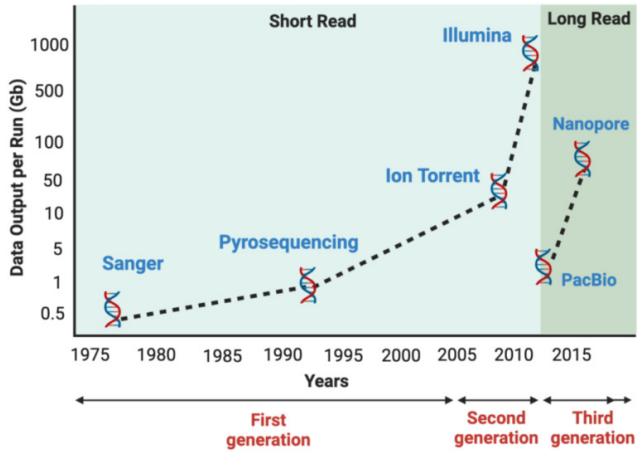
The beginning

- 1968 — The first 12 bases
- 1973 — 24 bases of the lactose-repressor binding site
  $\rightarrow$ two years of work: one base per month
- 1977 — Sanger sequencing and Gilbert sequencing
  $\rightarrow$ The Nobel Prize in Chemistry 1980 for Frederick Sanger and Walter Gilbert





Evolution of sequencing technologies



Satam et al, Biology (Basel) 2023

Alignment methods

Where do we need sequence alignments?

Where do we need sequence alignments?

- Sequence similarity
- Gene finding by similarity
- Protein structure by similarity
- RNA structure by similarity
- Motif finder
- Genome and transcriptome assembly
- **Gene expression estimation**

Evolutionary events

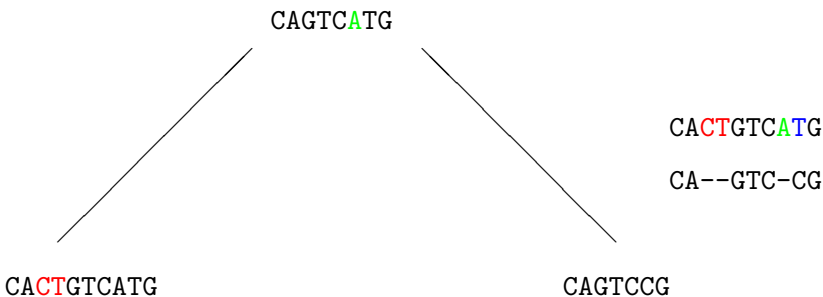
- DNA sequences change in time.
- Find evolutionary related sequences.
- Evolutionary events:

$$\texttt{CAGTCATG} \xrightarrow{\text{Insertion}} \texttt{CACTGTCATG} \xrightarrow{\text{Deletion}} \texttt{CACTGTCTG} \xrightarrow{\text{Substitution}} \texttt{CACTATCTG}$$

$$\texttt{CAGTCATG} \xrightarrow{\text{Duplication}} \texttt{CAGTGTCATG} \xrightarrow{\text{Translocation}} \texttt{CATCAGTGTG} \xrightarrow{\text{Inversion}} \texttt{CATTGACGTG}$$

Evolutionary tree

- Finding common ancestors.
- Parsimony principle: Evolution uses minimum number of operations.



- Probabilistic approaches (max. likelihood or sampling).

Alignments: optimize a score

Score of a given alignment:

```
CACTGTCATG
CA--GTC-CG
```

$$S_{\text{tot}} = S\begin{bmatrix}\texttt{C}\\\texttt{C}\end{bmatrix} + S\begin{bmatrix}\texttt{A}\\\texttt{A}\end{bmatrix} + S\begin{bmatrix}\texttt{C}\\\texttt{-}\end{bmatrix} + S\begin{bmatrix}\texttt{T}\\\texttt{-}\end{bmatrix} + S\begin{bmatrix}\texttt{G}\\\texttt{G}\end{bmatrix} + S\begin{bmatrix}\texttt{T}\\\texttt{T}\end{bmatrix} + S\begin{bmatrix}\texttt{C}\\\texttt{C}\end{bmatrix} + S\begin{bmatrix}\texttt{A}\\\texttt{-}\end{bmatrix} + S\begin{bmatrix}\texttt{T}\\\texttt{C}\end{bmatrix} + S\begin{bmatrix}\texttt{G}\\\texttt{G}\end{bmatrix}$$

- Score: substituting a residue in one seq. with a residue in another.
- Find the alignment that have the highest score.
- Try out all alignment combinations? (we deal with this soon)
- So speed of sequence comparisons **matters**!

Score matrices for DNA

- Identity: 8
- Transition: 2 (eg. {A,G} →: {A,G}; purine to purine)
- Transversion: -3 (eg. A→ {C,T}; purine to pyrimidine).

	A	C	G	T
A				
C				
G				
T				

What about gaps?

- Gap cost. Cost of indel (Eg. $d = 10$).
- Initiation and elongation.

Dynamical programming

- Find the alignment between `CACTGTCATG` and `CAGTCTG` that has the maximal score?
- What would be a trivial way?

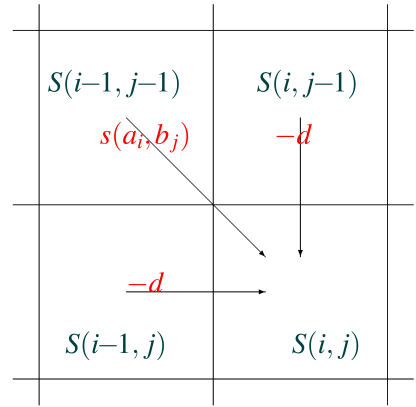
Dynamical programming

- Find the alignment between `CACTGTCATG` and `CAGTCTG` that has the maximal score?
- What would be a trivial way?
- Use sub sequences! $\rightarrow$ **Dynamic Programming**

Pairwise global alignments (Needleman–Wunsch)

Comparing sequences a and b. Given a substitution score $s(x, y)$ of replacing letter x with letter y, the highest scoring alignment can be found by the following recursion:

$$S(i,j) = \max \begin{cases} S(i\text{–}1, j\text{–}1) + s(a_i, b_j) \\ S(i\text{–}1, j) - d \\ S(i, j\text{–}1) - d \end{cases}$$



a_i residue at position i in seq. a
b_j residue at position j in seq. b
$i = 1, \ldots, N$; $j = 1, \ldots, M$

Initialization: $S(0,0) = 0$. Hence: $S(i,0) = -id$, $S(0,j) = -jd$.
Note: the alignment takes time $O(NM)$.

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0										
C											
A											
G											
T											
C											
T											
G											

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0	−10	−20	−30	−40	−50	−60	−70	−80	−90	−100
C	−10										
A	−20										
G	−30										
T	−40										
C	−50										
T	−60										
G	−70										

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0	−10	−20	−30	−40	−50	−60	−70	−80	−90	−100
C	−10	8	−2	−12	−22	−32	−42	−52	−62	−72	−82
A	−20	−2									
G	−30	−12									
T	−40	−22									
C	−50	−32									
T	−60	−42									
G	−70	−52									

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0	-10	-20	-30	-40	-50	-60	-70	-80	-90	-100
C	-10	8	-2	-12	-22	-32	-42	-52	-62	-72	-82
A	-20	-2	16	6	-4	-14	-24	-34	-44	-54	-64
G	-30	-12	6	13	3	4	-6	-16	-26	-36	-46
T	-40	-22	-4	8							
C	-50	-32	-14	4							
T	-60	-42	-24	-6							
G	-70	-52	-34	-16							

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0	−10	−20	−30	−40	−50	−60	−70	−80	−90	−100
C	−10	8	−2	−12	−22	−32	−42	−52	−62	−72	−82
A	−20	−2	16	6	−4	−14	−24	−34	−44	−54	−64
G	−30	−12	6	13	3	4	−6	−16	−26	−36	−46
T	−40	−22	−4	8	21	11	12	2	−8	−18	−28
C	−50	−32	−14	4	11	18	13	20	10	0	−10
T	−60	−42	−24	−6	12	8	26	16	17	18	8
G	−70	−52	−34	−16	2	20	10	23	18	14	26

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0	−10	−20	−30	−40	−50	−60	−70	−80	−90	−100
C	−10	8	−2	−12	−22	−32	−42	−52	−62	−72	−82
A	−20	−2	16	6	−4	−14	−24	−34	−44	−54	−64
G	−30	−12	6	13	3	4	−6	−16	−26	−36	−46
T	−40	−22	−4	8	21	11	12	2	−8	−18	−28
C	−50	−32	−14	4	11	18	13	20	10	0	−10
T	−60	−42	−24	−6	12	8	26	16	17	18	8
G	−70	−52	−34	−16	2	20	10	23	18	14	**26**

Back-tracking ...

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0	−10	−20	−30	−40	−50	−60	−70	−80	−90	−100
C	−10	8	−2	−12	−22	−32	−42	−52	−62	−72	−82
A	−20	−2	16	6	−4	−14	−24	−34	−44	−54	−64
G	−30	−12	6	13	3	4	−6	−16	−26	−36	−46
T	−40	−22	−4	8	21	11	12	2	−8	−18	−28
C	−50	−32	−14	4	11	18	13	20	10	0	−10
T	−60	−42	−24	−6	12	8	26	16	17	**18**	8
G	−70	−52	−34	−16	2	20	10	23	18	14	**26**

Back-tracking ...

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0	−10	−20	−30	−40	−50	−60	−70	−80	−90	−100
C	−10	8	−2	−12	−22	−32	−42	−52	−62	−72	−82
A	−20	−2	16	6	−4	−14	−24	−34	−44	−54	−64
G	−30	−12	6	13	3	4	−6	−16	−26	−36	−46
T	−40	−22	−4	8	21	11	12	2	−8	−18	−28
C	−50	−32	−14	4	11	18	13	20	**10**	0	−10
T	−60	−42	−24	−6	12	8	26	16	17	**18**	8
G	−70	−52	−34	−16	2	20	10	23	18	14	**26**

Back-tracking ...

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0	−10	−20	−30	−40	−50	−60	−70	−80	−90	−100
C	−10	8	−2	−12	−22	−32	−42	−52	−62	−72	−82
A	−20	−2	16	6	−4	−14	−24	−34	−44	−54	−64
G	−30	−12	6	13	3	4	−6	−16	−26	−36	−46
T	−40	−22	−4	8	21	11	12	2	−8	−18	−28
C	−50	−32	−14	4	11	18	13	**20**	**10**	0	−10
T	−60	−42	−24	−6	12	8	26	16	17	**18**	8
G	−70	−52	−34	−16	2	20	10	23	18	14	**26**

Back-tracking ...

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0	−10	−20	−30	−40	−50	−60	−70	−80	−90	−100
C	−10	8	−2	−12	−22	−32	−42	−52	−62	−72	−82
A	−20	−2	16	6	−4	−14	−24	−34	−44	−54	−64
G	−30	−12	6	13	3	4	−6	−16	−26	−36	−46
T	−40	−22	−4	8	21	11	**12**	2	−8	−18	−28
C	−50	−32	−14	4	11	18	13	**20**	**10**	0	−10
T	−60	−42	−24	−6	12	8	26	16	17	**18**	8
G	−70	−52	−34	−16	2	20	10	23	18	14	**26**

Back-tracking ...

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0	−10	−20	−30	−40	−50	−60	−70	−80	−90	−100
C	−10	8	−2	−12	−22	−32	−42	−52	−62	−72	−82
A	−20	−2	16	6	−4	−14	−24	−34	−44	−54	−64
G	−30	−12	6	13	3	**4**	−6	−16	−26	−36	−46
T	−40	−22	−4	8	21	11	**12**	2	−8	−18	−28
C	−50	−32	−14	4	11	18	13	**20**	**10**	0	−10
T	−60	−42	−24	−6	12	8	26	16	17	**18**	8
G	−70	−52	−34	−16	2	20	10	23	18	14	**26**

Back-tracking ...

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0	−10	−20	−30	−40	−50	−60	−70	−80	−90	−100
C	−10	8	−2	−12	−22	−32	−42	−52	−62	−72	−82
A	−20	−2	16	6	**-4**	−14	−24	−34	−44	−54	−64
G	−30	−12	6	13	3	**4**	−6	−16	−26	−36	−46
T	−40	−22	−4	8	21	11	**12**	2	−8	−18	−28
C	−50	−32	−14	4	11	18	13	**20**	**10**	0	−10
T	−60	−42	−24	−6	12	8	26	16	17	**18**	8
G	−70	−52	−34	−16	2	20	10	23	18	14	**26**

Back-tracking ...

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0	−10	−20	−30	−40	−50	−60	−70	−80	−90	−100
C	−10	8	−2	−12	−22	−32	−42	−52	−62	−72	−82
A	−20	−2	16	**6**	**-4**	−14	−24	−34	−44	−54	−64
G	−30	−12	6	13	3	**4**	−6	−16	−26	−36	−46
T	−40	−22	−4	8	21	11	**12**	2	−8	−18	−28
C	−50	−32	−14	4	11	18	13	**20**	**10**	0	−10
T	−60	−42	−24	−6	12	8	26	16	17	**18**	8
G	−70	−52	−34	−16	2	20	10	23	18	14	**26**

Back-tracking ...

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

		C	A	C	T	G	T	C	A	T	G
	0	−10	−20	−30	−40	−50	−60	−70	−80	−90	−100
C	−10	8	−2	−12	−22	−32	−42	−52	−62	−72	−82
A	−20	−2	**16**	**6**	**-4**	−14	−24	−34	−44	−54	−64
G	−30	−12	6	13	3	**4**	−6	−16	−26	−36	−46
T	−40	−22	−4	8	21	11	**12**	2	−8	−18	−28
C	−50	−32	−14	4	11	18	13	**20**	**10**	0	−10
T	−60	−42	−24	−6	12	8	26	16	17	**18**	8
G	−70	−52	−34	−16	2	20	10	23	18	14	**26**

Back-tracking ...

Example of global alignment:

Align the two sequences CACTGTCATG and CAGTCTG

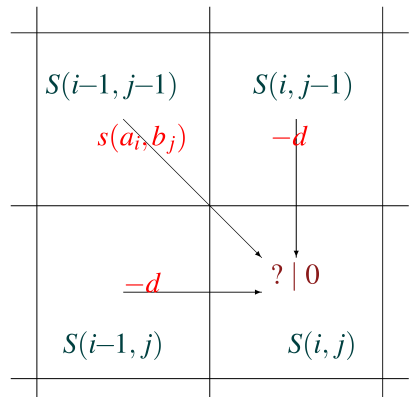
		C	A	C	T	G	T	C	A	T	G
	0	−10	−20	−30	−40	−50	−60	−70	−80	−90	−100
C	−10	**8**	−2	−12	−22	−32	−42	−52	−62	−72	−82
A	−20	−2	**16**	**6**	**-4**	−14	−24	−34	−44	−54	−64
G	−30	−12	6	13	3	**4**	−6	−16	−26	−36	−46
T	−40	−22	−4	8	21	11	**12**	2	−8	−18	−28
C	−50	−32	−14	4	11	18	13	**20**	**10**	0	−10
T	−60	−42	−24	−6	12	8	26	16	17	**18**	8
G	−70	−52	−34	−16	2	20	10	23	18	14	**26**

Back-tracking ...

Pairwise local alignments (Smith–Waterman)

Comparing sequences a and b. Given a substitution score $s(x, y)$ of replacing letter x with letter y, the highest scoring alignment can be found by the following recursion:

$$S(i,j) = \max \begin{cases} S(i\text{–}1, j\text{–}1) + s(a_i, b_j) \\ S(i\text{–}1, j) - d \\ S(i, j\text{–}1) - d \\ 0 \end{cases}$$



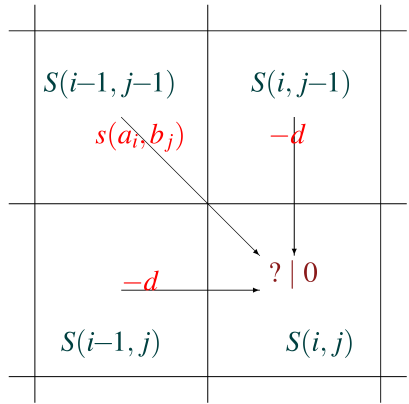
Note only positive numbers!
$i = 1, \ldots, N$; $j = 1, \ldots, M$

Initialization: $S(0,0) = 0$. Hence: $S(i,0) = ?$, $S(0,j) = ?$.

Pairwise local alignments (Smith–Waterman)

Comparing sequences a and b. Given a substitution score $s(x, y)$ of replacing letter x with letter y, the highest scoring alignment can be found by the following recursion:

$$S(i,j) = \max \begin{cases} S(i\text{–}1, j\text{–}1) + s(a_i, b_j) \\ S(i\text{–}1, j) - d \\ S(i, j\text{–}1) - d \\ 0 \end{cases}$$



Note only positive numbers!
$i = 1, \ldots, N$; $j = 1, \ldots, M$

Initialization: $S(0,0) = 0$. Hence: $S(i,0) = 0$, $S(0,j) = 0$.

Example of local alignment:

Align the two sequences AAACTGTTTAACAG and AACAGGGGAAACTG.

		A	A	A	C	T	G	T	T	T	A	A	C	A	G
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A	0														
A	0														
C	0														
A	0														
G	0														
G	0														
G	0														
G	0														
A	0														
A	0														
A	0														
C	0														
T	0														
G	0														

Example of local alignment:

Align the two sequences AAACTGTTTAACAG and AACAGGGGAAACTG.

		A	A	A	C	T	G	T	T	T	A	A	C	A	G
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A	0	8	8	8	0	0	2	0	0	0	8	8	0	8	2
A	0	8	16	16	6	0	2	0	0	0	8	16	6	8	10
C	0	0	6	13	24	14	4	4	2	2	0	6	24	14	5
A	0	8	8	14	14	21	16	6	1	0	10	8	14	32	22
G	0	2	10	10	11	11	29	19	9	0	2	12	5	22	40
G	0	2	4	12	7	8	19	26	16	6	2	4	9	12	30
G	0	2	4	6	9	4	16	16	23	13	8	4	1	11	20
G	0	2	4	6	3	6	12	13	13	20	15	10	1	3	19
A	0	8	10	12	3	0	8	9	10	10	28	23	13	9	9
A	0	8	16	18	9	0	2	5	6	7	18	36	26	21	11
A	0	8	16	24	15	6	2	0	2	3	15	26	33	34	24
C	0	0	6	14	32	22	12	4	2	4	5	16	34	30	31
T	0	0	0	4	22	40	30	20	12	10	1	6	24	31	27
G	0	2	2	2	12	30	48	38	28	18	12	3	14	26	39

Example of local alignment:

Align the two sequences AAACTGTTTAACAG and AACAGGGGAAACTG.

		A	A	A	C	T	G	T	T	T	A	A	C	A	G
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A	0	8	8	8	0	0	2	0	0	0	**8**	8	0	8	2
A	0	8	16	16	6	0	2	0	0	0	8	**16**	6	8	10
C	0	0	6	13	24	14	4	4	2	2	0	6	**24**	14	5
A	0	8	8	14	14	21	16	6	1	0	10	8	14	**32**	22
G	0	2	10	10	11	11	29	19	9	0	2	12	5	22	**40**
G	0	2	4	12	7	8	19	26	16	6	2	4	9	12	30
G	0	2	4	6	9	4	16	16	23	13	8	4	1	11	20
G	0	2	4	6	3	6	12	13	13	20	15	10	1	3	19
A	0	**8**	10	12	3	0	8	9	10	10	28	23	13	9	9
A	0	8	**16**	18	9	0	2	5	6	7	18	36	26	21	11
A	0	8	16	**24**	15	6	2	0	2	3	15	26	33	34	24
C	0	0	6	14	**32**	22	12	4	2	4	5	16	34	30	31
T	0	0	0	4	22	**40**	30	20	12	10	1	6	24	31	27
G	0	2	2	2	12	30	**48**	38	28	18	12	3	14	26	39

Example of local alignment:

Align the two sequences AAACTGTTTAACAG and AACAGGGGAAACTG.

		A	A	A	C	T	G	T	T	T	A	A	C	A	G
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A	0	8	8	8	0	0	2	0	0	0	**8**	8	0	8	2
A	0	8	16	16	6	0	2	0	0	0	8	**16**	6	8	10
C	0	0	6	13	24	14	4	4	2	2	0	6	**24**	14	5
A	0	8	8	14	14	21	16	6	1	0	10	8	14	**32**	22
G	0	2	10	10	11	11	29	19	9	0	2	12	5	22	**40**
G	0	2	4	12	7	8	19	26	16	6	2	4	9	12	30
G	0	2	4	6	9	4	16	16	23	13	8	4	1	11	20
G	0	2	4	6	3	6	12	13	13	20	15	10	1	3	19
A	0	**8**	10	12	3	0	8	9	10	10	28	23	13	9	9
A	0	8	**16**	18	9	0	2	5	6	7	18	36	26	21	11
A	0	8	16	**24**	15	6	2	0	2	3	15	26	33	34	24
C	0	0	6	14	**32**	22	12	4	2	4	5	16	34	30	31
T	0	0	0	4	22	**40**	30	20	12	10	1	6	24	31	27
G	0	2	2	2	12	30	**48**	38	28	18	12	3	14	26	39

Exercise: Dynamical programming of pairwise alignment

Complete the dynamic programming matrix of a global alignment:

Align the sequences ACGTG and AACGGTG using a match score of 1, a mismatch of -4 and a gap cost of -10.

		A	C	G	T	G
A						
A						
C						
G						
G						
T						
G						

Time is important

Dynamic programming: $O(NM)$ time complexity

Time is important

Dynamic programming: $O(NM)$ time complexity

When does time become an issue?

Summary

- Dynamic programming (DP) saves time in sequence comparisons
- Some assumptions in DP, mention some
- In many applications, heuristics are needed to further speed up the comparison, *e.g.*, seed-and-extend heuristics

Speed up of local sequence alignments

When becomes time an issue?

- Target is entire genome
- Target is all observed sequences (*e.g.* RefSeq non-redundant database)
- **Query are millions of reads**

How many reads do we get from modern Illumina sequencing?

Instrument	Flow cell	Reads	Output (Gb)	Run time	Read length
NovaSeq X Plus	25B	∼16 B	∼6,000	∼48 h	2×150 bp
NovaSeq X Plus	10B	∼6.4 B	∼2,400	∼48 h	2×150 bp
NovaSeq 6000	S4	∼8–10 B	∼3,000	∼44 h	2×150 bp
NovaSeq 6000	S2	∼3.3–4.1 B	∼1,500	∼36 h	2×150 bp
HiSeq 4000	2× FC	∼5 B	∼1,500	∼84 h	2×150 bp
NextSeq 2000	P3	∼1.1–1.2 B	∼360	∼29 h	2×150 bp
NextSeq 2000	P2	∼400 M	∼120	∼19 h	2×150 bp
MiSeq	v3	∼22–25 M	∼15	∼56 h	2×300 bp

Raw data (Sequencing reads)

The FASTQ format:

```
@ERR459145.1 DHKW5DQ1:219:D0PT7ACXX:2:1101:1590:2149/1
GATCGGAAGAGCGGTTCAGCAGGAATGCCGAGATCGGAAGAGCGGTTCAGC
+
@7<DBADDDBH?DHHI@DH>HHHEGHIIIGGIFFGIBFAAGAFHA'5?B@D
```

- @: begin header
- 2:1101 flowcell lane 2, tile 1101
- x and y coordinates: 1590:2149
- /1 single-end reads;
  /1 and /2 paired-end (mate-paired) reads
- read sequence
- quality encoded ASCII characters

What is read mapping?

Determine position of a short read on the reference genome or transcriptome.

```
Reference:  ...AA-CGCCTT...      | = match
               |:-:|||||         : = mismatch
Read:          AGGGGCCTT         - = gap
```

Naive mapping

Search for query at each position in reference genome

```
ACGTTACCGAATCGATCAAAGTCGA
GTTA
```

m = query length, n = genome length

Naive mapping

Search for query at each position in reference genome

```
ACGTTACCGAATCGATCAAAGTCGA
 GTTA
```

m = query length, n = genome length

Naive mapping

Search for query at each position in reference genome

```
ACGTTACCGAATCGATCAAAGTCGA
  GTTA                    :)
```

m = query length, n = genome length $\rightarrow$ Time: $O(mn)$

Naive mapping

- Human Genome (queries) would take far too long:
  - Illumina sequencing technology produces
    X billion reads of 150 – 300 bp length
  - Mapping these reads to a **3.2 billion bp** human genome is a challenge
- Far worse when we allow for indels (insertion+deletions) and mismatches.

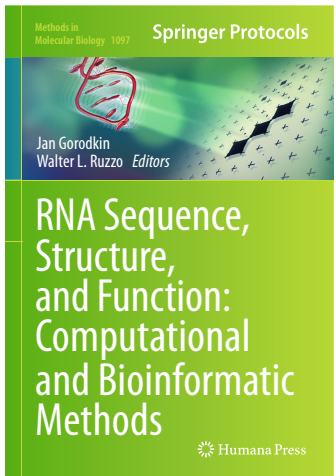
$\rightarrow$ Are optimal alignments based on quality scores still feasible?

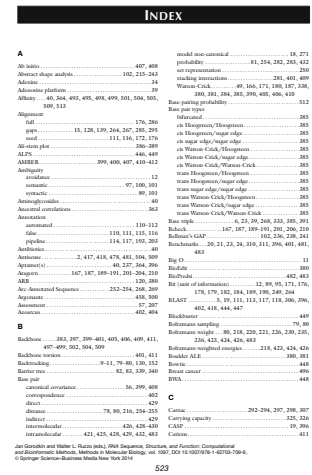
Principles of mapping reads

- Most computational time is spent on alignment
- **Many sequenced reads are redundant**
- We do not need to search the entire genome each time again

Book analog

Do not search the entire book, instead search the book **index**.





INDEX

A

Ab initio 407, 408
Abstract shape analysis 102, 215–243
Adenine 24
Adenosine platform 39
Affinity ... 40, 364, 493, 495, 498, 499, 501, 504, 505, 509, 512
Alignment
  full 176, 286
  gaps 15, 128, 129, 264, 267, 285, 295
  seed 111, 116, 172, 176
Ali-stem plot 386–389
ALPS 446, 449
AMBER 399, 400, 407, 410–412
Ambiguity
  avoidance 12
  semantic 97, 100, 101
  syntactic 89, 101
Aminoglycosides 40
Ancestral correlations 863
Annotation
  automated 110–112
  false 110, 111, 115, 116
  pipeline 114, 117, 193, 203
Antibiotics 40
Antisense 2, 417, 418, 478, 481, 504, 509
Aptamer(s) 48, 237, 364, 396
Assgorm 167, 187, 189–191, 201–204, 210
ARB 120, 380
Arc-Annotated Sequence 252–254, 268, 269
Argonaute 458, 500
Assessment 57, 207
Assurcon 402, 404

B

Backbone 383, 397, 399–401, 405, 406, 409, 411, 497–499, 502, 504, 509
Backbone torsion 401, 411
Backtracking 9–11, 79–80, 130, 152
Barrier tree 82, 83, 339, 340
Base pair
  canonical covariance 56, 399, 408
  correspondence 402
  direct 429
  distance 78, 80, 216, 254–255
  indirect 429
  intermolecular 426, 428–430
  intramolecular 421, 425, 428, 429, 432, 483
  model non-canonical 18, 271
  probability 81, 254, 282, 283, 432
  set representation 250
  stacking interactions 281, 401, 409
Watson-Crick 49, 166, 171, 180, 187, 238, 380, 381, 384, 385, 390, 405, 406, 410
Base-pairing probability 512
Base pair types
  bifurcated 285
  cis Hoogsteen/Hoogsteen 285
  cis Hoogsteen/sugar edge 285
  cis sugar edge/sugar edge 285
  cis Watson-Crick/Hoogsteen 285
  cis Watson-Crick/sugar edge 285
  cis Watson-Crick/Watson-Crick 285
  trans Hoogsteen/Hoogsteen 285
  trans Hoogsteen/sugar edge 285
  trans sugar edge/sugar edge 285
  trans Watson-Crick/Hoogsteen 285
  trans Watson-Crick/sugar edge 285
  trans Watson-Crick/Watson-Crick 285
Base triple 6, 23, 39, 268, 333, 385, 391
Bcheck 167, 187, 189–191, 201, 206, 210
Bellman's GAP 102, 236, 238, 241
Benchmarks ... 20, 21, 23, 24, 310, 311, 396, 401, 481, 483
Big O 11
BioEdit 380
BioPerl 482, 483
Bit (unit of information) 12, 89, 95, 171, 176, 178, 179, 182, 184, 189, 190, 249, 264
BLAST 5, 19, 111, 113, 117, 118, 306, 396, 402, 418, 444, 447
Blockbuster 449
Boltzmann sampling 79, 80
Boltzmann weight ... 80, 218, 220, 221, 226, 230, 235, 236, 423, 424, 426, 483
Boltzmann-weighted energies 218, 423, 424, 426
Boulder ALE 380, 381
Bowtie 448
Breast cancer 496
BWA 448

C

Camac 292–294, 297, 298, 307
Carrying capacity 225, 226
CASP 19, 396
Cations 411

Jan Gorodkin and Walter L. Ruzzo (eds.), *RNA Sequence, Structure, and Function: Computational and Bioinformatic Methods*, Methods in Molecular Biology, vol. 1097, DOI 10.1007/978-1-62703-709-9,

© Springer Science+Business Media New York 2014

Gorodkin et al. Methods Mol Biol. 2014

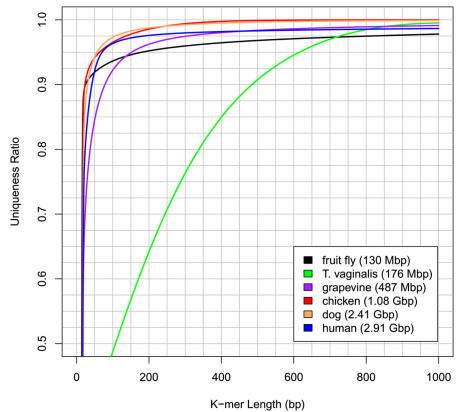
k-mer

"k-mer" is a substring of length k

For example sequence **GGCGATTCATCG**:

4-mer GGCG, GCGA CGAT

3-mer GGC, GCG, CGA, GAT



Schatz et al. Genome Res 2010

Read mapping through indexing

In principle read mapping is to map an exact piece of sequence to the genome.

Solution

An index is a data structure that improves the speed of data retrieval operations at the cost of additional storage space to maintain the index data structure.

Pros

Quick search for matches in an entire genome.

Cons

Index structure of the entire genome takes a lot of memory.

Why might an exact mapping not always be what we want?

Why might an exact mapping not always be what we want?

Which concerns might you have when mapping genomic sequences?

Which concerns might you have when mapping transcribed sequences?

Indexing problems

Flexibility and constraints:

- *Errors versus natural variation:*
  trade-off in error threshold
- *Computational efficiency (time / memory):*
  allowed mismatches / unique mappings
- Balance between speed, memory and reported mappings

Indexing method choice is crucial!

- **Hash-based** (BLAST, Salmon, Kallisto)
- **Suffix arrays** (Salmon, STAR)

  A sorted table of all suffixes (substrings) of a given string
- **Burrows-Wheeler Transform** (BWA, SOAP2, Bowtie2, Hisat)

  A compressed form of suffix arrays

Indexing is often used for seed matching

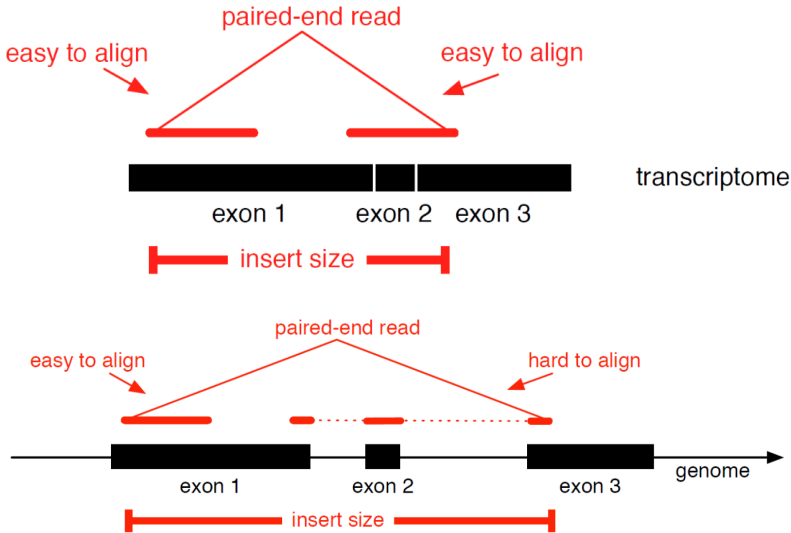
"Seed-and-extend" heuristic

1. find the best possible match of a seed in an index made up from the reference sequences
2. every matched seed is extended on both sides by optimal local alignment

Common software

STAR, HISAT2, BLAST

Transcriptome versus genome mapping



BLAST (Basic Local Alignment Search Tool)

Less accurate than Smith-Waterman, **BUT** 50 times faster.

Idea: true matches are likely to have short stretches of identity (high score).

1. List of short words of fixed length that will match the query sequence (word length: 3 for protein; 11 for nucleic acids).
2. Scan database (word hash) for these words.
3. Extend matches in both directions in an attempt to find an alignment with a score exceeding S.

Segment pairs whose scores cannot be improved by extending or trimming are called high scoring pairs (HSPs).

What are the default parameter settings of NCBI blastn and megablast?

Which differences in the alignments do you expect based on their parameters?

Alignment score statistics

Question: Given a particular scoring system, how many distinct local alignments with score $\geq$S can one expect to find by chance from the comparison of two random sequences of lengths m and n?

Or in other words, when can a local alignment be considered statistically significant?

E-values and P-values

The expected number of local alignments with a score of at least S is given by the E-value for the score S:

$$E = Kmne^{-\lambda S}$$

where K and λ are statistical parameters that depend on the BLAST parameters.

1. Doubling the length of the query sequence (m) or the size of the database (n) should double the number of local alignments.
2. E-value decreases exponentially as score S increases.

The probability of observing *at least* one alignment with score $\geq S$

$$p = 1 - e^{-E}$$

$\Rightarrow$ Sequence similarity score S is *extreme value distributed*

---

https://www.ncbi.nlm.nih.gov/BLAST/tutorial/

From Raw Reads to Expression Matrix

Counts, Norms, and Transforms

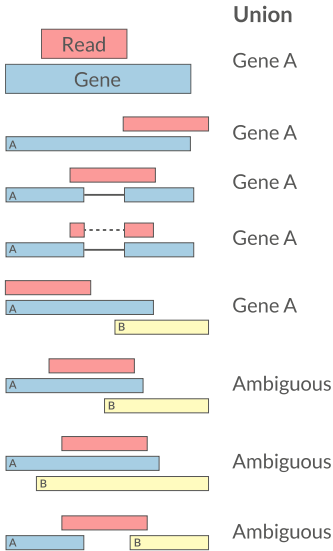
Quantification – estimation of expression

Quantification:
Count reads mapping to a genomic feature (gene, transcript, exon etc)

$\rightarrow$ genomic features can be annotated (e.g., NCBI RefSeq genome annotation) or be predicted (e.g., transcriptome assembly)

Assumption:
Number of reads produced from a feature $\sim$ feature's **relative** abundance in the sample

Gene-level quantification



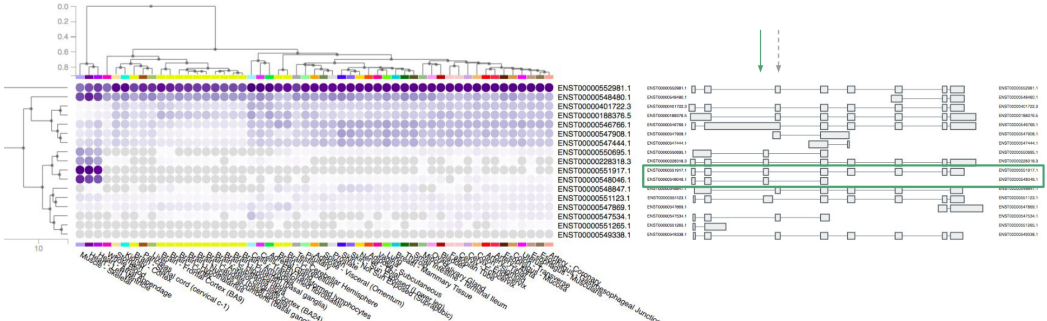
- By default ambiguous reads are not counted
- By default multi-mapping reads (reads aligning with multiple locations) are not counted

Common read counting software

featureCounts, HTSeq

Transcript-level quantification

Quantify per transcript (not just exon or gene)



- Probabilistic assignment of ambiguous reads to transcript.

Common software

RSEM, Cufflinks

Alignment-free methods ("pseudoaligners")

1. Focus: Only annotated transcripts (not entire genome!)
2. Pseudoalign: K-mer composition of reads/transcripts
3. Abundance *estimation*
4. GC-content, transcript position correction included (*e.g.* 3' end degragation)

Pros

dramatic increase in speed; improvements in accuracy for gene-level quantification

Cons

absolute reliance on a precise and comprehensive transcript annotation; no information on where each read is mapping

Common software

Salmon, Kallisto

Count matrix

	SRR1039508	SRR1039509	SRR1039512	SRR1039513	SRR1039516	SRR1039517	SRR1039520	SRR1039521
ENSG00000000003	679	448	873	408	1138	1047	770	572
ENSG00000000005	0	0	0	0	0	0	0	0
ENSG00000000419	467	515	621	365	587	799	417	508
ENSG00000000457	260	211	263	164	245	331	233	229
ENSG00000000460	60	55	40	35	78	63	76	60
ENSG00000000938	0	0	2	0	1	0	0	0
ENSG00000000971	3251	3679	6177	4252	6721	11027	5176	7995
ENSG00000001036	1433	1062	1733	881	1424	1439	1359	1109
ENSG00000001084	519	380	595	493	820	714	696	704
ENSG00000001167	394	236	464	175	658	584	360	269
ENSG00000001460	172	168	264	118	241	210	155	177
ENSG00000001461	2112	1867	5137	2657	2735	2751	2467	2905
ENSG00000001497	524	488	638	357	676	806	493	475
ENSG00000001561	71	51	211	156	23	38	134	172
ENSG00000001617	555	394	905	415	727	697	618	599
ENSG00000001626	10	2	9	2	10	6	5	5
ENSG00000001629	1660	1251	2259	1079	2462	2514	1888	1660
ENSG00000001630	59	54	66	23	84	87	31	59
ENSG00000001631	729	692	943	475	1034	1163	731	744
ENSG00000002016	201	161	256	99	268	257	160	137
ENSG00000002079	3	0	3	1	4	0	0	1

Estimation of gene expression

You are analyzing 2 genes (gene A and B) in two conditions (condition 1 and 2) on the bases of an RNA-seq experiment that resulted in the following number of reads:

	Condition 1	Condition 2
Gene A	1000	3000
Gene B	2000	4000

Are the following statements correct?

a. Both genes A and B are more expressed in condition 2.

b. Gene B is more expressed than gene A.

Estimation of gene expression

We cannot state any such thing since we do not know

a. **sequencing depth (library size)**,
   expression of all other genes within the sample

   $\rightarrow$ RNA-seq data informs about the relative abundance BUT NOT about the absolute abundance.

b. **gene length**,
   the longer the gene, more reads will be mapped

Estimation of gene expression

We cannot state any such thing since we do not know

a. **sequencing depth (library size)**,
   expression of all other genes within the sample

   $\rightarrow$ RNA-seq data informs about the relative abundance BUT NOT about the absolute abundance.

b. **gene length**,
   the longer the gene, more reads will be mapped

Solution: Control (Normalize) for

1. sequencing depth
2. compositional bias

Normalization of counts: RPKM

RPKM: Reads Per Kilobase transcript per Million mapped reads

R = Reads count for the gene
G = Gene length in kilobases
T = Total number of mapped reads in a sample

$$RPKM = \frac{\left(\frac{R}{T/10^6}\right)}{G}$$

Gene	Reads (R)	Length (G)	RPKM
A	100	1.0	100,000
B	400	2.0	200,000
C	500	2.5	200,000
T	1000		**500,000**

Normalization of counts: RPKM

RPKM: Reads Per Kilobase transcript per Million mapped reads

R = Reads count for the gene
G = Gene length in kilobases
T = Total number of mapped reads in a sample

$$RPKM = \frac{\left(\frac{R}{T/10^6}\right)}{G}$$

Gene	Reads (R)	Length (G)	RPKM
A	100	1.0	100,000
B	400	2.0	200,000
C	500	2.5	200,000
T	1000		**500,000**

FPKM: "Fragments" Per Kilobase transcript per Million mapped reads

- Same concept as RPKM with a slight difference:
- RPKM: single-end, FPKM: paired-end

Gene count comparisons between genes within a sample; NOT for between-sample comparisons

Normalization of counts: TPM

TPM: Transcripts/Tags Per Million

R = Reads count for the gene
G = Gene length in kilobases
$\sum \frac{R}{G}$ = Sum of length-normalized reads

$$TPM = \frac{\frac{R}{G}}{\left(\sum \frac{R}{G}\right)} \times 10^6 = \frac{RPKM}{\sum RPKM}$$

Gene	Reads (R)	Length (G)	TPM
A	100	1.0	200,000
B	400	2.0	400,000
C	500	2.5	400,000
$\sum \frac{R}{G}$	500		**1,000,000**

Normalization of counts: TPM

TPM: Transcripts/Tags Per Million

R = Reads count for the gene
G = Gene length in kilobases
$\sum \frac{R}{G}$ = Sum of length-normalized reads

$$TPM = \frac{\frac{R}{G}}{\left(\sum \frac{R}{G}\right)} \times 10^6 = \frac{RPKM}{\sum RPKM}$$

Gene	Reads (R)	Length (G)	TPM
A	100	1.0	200,000
B	400	2.0	400,000
C	500	2.5	400,000
$\sum \frac{R}{G}$	500		**1,000,000**

Key properties of TPM

- Total TPM per sample is always 10^6
- Enables meaningful **between-sample** comparisons

Normalization of counts: CPM

CPM: Counts Per Million

R = Reads count for the gene
T = Total number of mapped reads in a sample

$$CPM = \frac{R}{T} \times 10^6$$

Gene	Reads (R)	Length (G)	CPM
A	100	1.0	100,000
B	400	2.0	400,000
C	500	2.5	500,000
T	1000		**1,000,000**

Normalization of counts: CPM

CPM: Counts Per Million

R = Reads count for the gene
T = Total number of mapped reads in a sample

$$CPM = \frac{R}{T} \times 10^6$$

Gene	Reads (R)	Length (G)	CPM
A	100	1.0	100,000
B	400	2.0	400,000
C	500	2.5	500,000
T	1000		**1,000,000**

Key properties of CPM

- Most used metric in the field
- Normalized only by library size (T)
- Does **not** account for gene length (G)
- Total CPM per sample is always 10^6, like TPM; therefore:
- Suitable for **between-sample** comparisons

Simple normalization could fail

Genes	Control	Treated
Gene A	10	30
Gene B	30	90
Gene C	5	15
Gene D	1	3
Gene N	1000	240
Total	1046	378

DESeq2's median of ratios (sample-wise "size factor")

Assumption: most genes are not differential expressed

1. For each gene, calculate geometric mean

Genes	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5	Sample 6	Geomean
Gene 1	34	56	23	12	10	30	23
Gene 2	10	6	7	11	12	8	9
Gene n	65	78	67	34	56	23	50

2. For each gene, calculate ratio to geometric mean

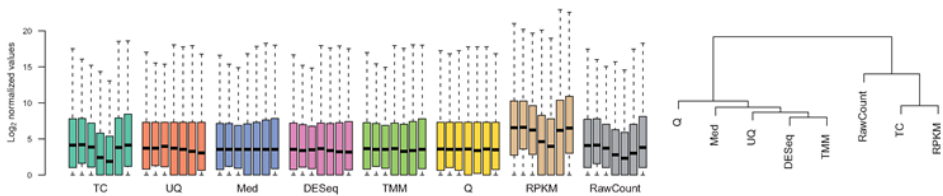
Genes	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5	Sample 6
Gene 1	1.5	2.4	1.0	0.5	0.4	1.3
Gene 2	1.1	0.7	0.8	1.3	1.4	0.9
Gene n	1.3	1.6	1.4	0.7	1.1	0.5

3. Take median of these ratios as sample normalization factor ("size factor")

1.3	1.6	1	0.7	1.1	0.9

Normalization

- Use **raw counts** for DE analysis using DGE packages (*e.g.* DESeq2, edgeR) as normalization is done internally
- RPKM/FPKM/TPM/CPM are not recommended for DE analysis.
- TPM/CPM are great for visual exploration of data (heatmaps, abundance comparisons, PCA/MDS plots)
  $\rightarrow$ sum of all TPMs/CPMs is constant (1 million)

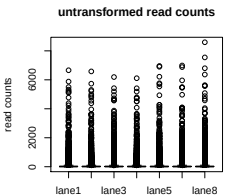


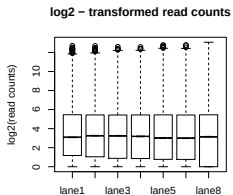
- Median of Ratios (DESeq2) and TMM (edgeR) methods perform the best for DGE
- other solutions: spike-ins/house-keeping genes

Dillies *et al.* "A comprehensive evaluation of normalization methods for Illumina high-throughput RNAseq data analysis." Brief Bioinform. 2013

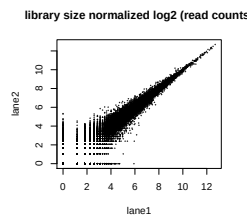
Transformation of sequencing-depth-normalized read counts

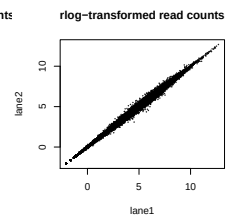
Log_2 transformation:





Transformation incl variance shrinkage:





For clustering, heatmaps etc use VST (DESeq2), VOOM (limma) or RLOG (DESeq2)

Challenge your data by different normalization methods → robustness of DGE analysis

---

Eder *et al.* "NVT: a fast and simple tool for the assessment of RNA-Seq normalization strategies" Bioinformatics 2016

Alternative: Compositional data analysis

DESeq's median of ratios and edgeR's TMM are less suitable in highly **asymmetrical** or **sparse** datasets → unacceptably high false positive DEGs

- ratio transformations capture the relationships between the features in the dataset
- centered log-ratio (clr) transformation:

  $\mathrm{x}_{clr} = [log(x_1/G(\mathrm{x})), log(x_2/G(\mathrm{x})) \ldots log(x_D/G(\mathrm{x}))]\,,$
  $G(\mathrm{x}) = \sqrt[D]{x_1 \cdot x_2 \cdot \ldots \cdot x_D}$
- clr-transformed values are scale-invariant (same ratio with few as well as many read counts)
- to calculate G(x) we have to delete, replace or **estimate** the 0 count values
- estimate technical variation within each sample using Monte-Carlo instances drawn from the Dirichlet distribution → probability vector prior to clr transformation

Gloor *et al.* "Microbiome Datasets Are Compositional: And This Is Not Optional" Front Microbiol 2017