Processing data from high-throughput sequencing experiments

Simon Anders

EMBL

Use-cases for HTS

- de-novo sequencing and assembly of small genomes
- transcriptome analysis (RNA-Seq, sRNA-Seq, ...)
  - identifying transcripted regions
  - expression profiling
- Resequencing to find genetic polymorphisms:
  - SNPs, micro-indels
  - CNVs
- ChIP-Seq, nucleosome positions, etc.
- DNA methylation studies (after bisulfite treatment)
- environmental sampling (metagenomics)
- reading bar codes

EMBL

Use cases for HTS: Bioinformatics challenges

Established procedures may not be suitable.

New algorithms are required for

- assembly
- alignment
- statistical tests (counting statistics)
- visualization
- segmentation
- …

Where does Bioconductor come in?

Several steps:

- Processing of the images and determining of the read sequencest
  - typically done by core facility with software from the manufacturer of the sequencing machine
- Aligning the reads to a reference genome (or assembling the reads into a new genome)
  - Done with community-developed stand-alone tools.
- Downstream statistical analyis.
  - Write your own scripts with the help of Bioconductor infrastructure.

EMBL

Solexa standard workflow

SolexaPipeline

- "Firecrest": Identifying clusters
  ⇨ typically 15..20 mio good clusters per lane

- "Bustard": Base calling
  ⇨ sequence for each cluster,
  with Phred-like scores

- "Eland": Aligning to reference

EMBL

Firecrest output

Large tab-separated text files with one row per identified cluster, specifying

- lane index and tile index
- x and y coordinates of cluster on tile
- for each cycle a group of four number, specifying the flourescence intensity for A, C, G, and T.

Bustard output

Two tab-seperated text files, with one row per cluster:

- "seq.txt" file:
  - lane and tile index, x and y coordinates
  - the called sequence as string of A, C, G, T
- "prb.txt" file:
  - Phred-like scores, ranging from -40 to 40;
  - one value per called base

Fastq format

“FASTA with Qualities”

Example:

```
@HWI-EAS225:3:1:2:854#0/1
GGGGGGAAGTCGGCAAAATAGATCCGTAACTTCGGG
+HWI-EAS225:3:1:2:854#0/1
a`abbbbabaabbababb^`[aaa`_N]b^ab^``a
@HWI-EAS225:3:1:2:1595#0/1
GGGAAGATCTCAAAAACAGAAGTAAAACATCGAACG
+HWI-EAS225:3:1:2:1595#0/1
a`abbbababbbabbbbbbabb`aaababab\aa_`
```

Fastq format

Each read is represented by four lines:

- '@', followed by read ID
- sequence
- '+', optionally followed by repeated read ID
- quality string:
  - same length as sequence
  - each character encodes the base-call quality of one base

Fastq format: Base-call quality strings

- If p is the probability that the base call is wrong, the Phred score is:

$$Q = -10 \log_{10} p$$

- The score is written with the character whose ASCII code is $Q+33$ (Sanger Institute standard).

- Solexa uses instead the character with ASCII code $Q+64$.
- Before SolexaPipeline version 1.3, Solexa also used a different formula, namely $Q = -10 \log_{10} (p/(1\text{-}p))$

EMBL

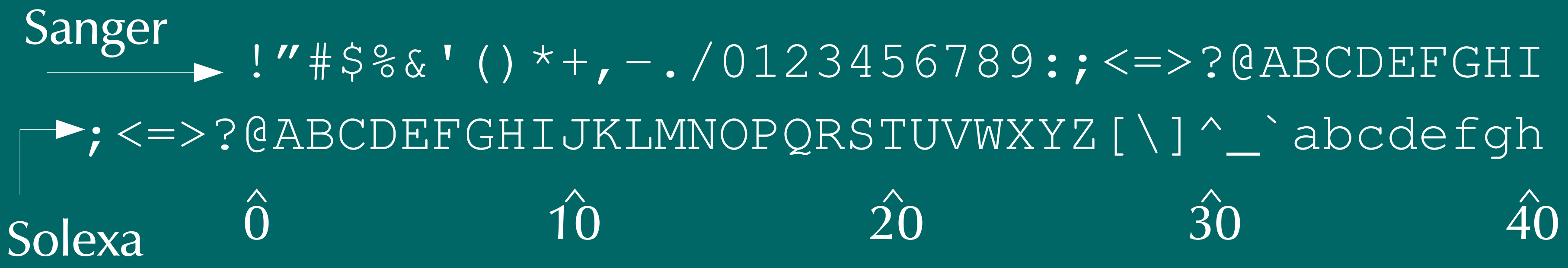
FASTQ: Phred base-call qualities

quality score Q_{phred}	error prob. p	characters
0 .. 9	1 .. 0.13	!”#$%&'()*
10 .. 19	0.1 .. 0.013	+,-./01234
20 .. 29	0.01 .. 0.0013	56789:;<=>
30 .. 39	0.001 .. 0.00013	?@ABCDEFGH
40	0.0001	I

EMBL

The Sanger / Solexa FASTQ confusion

Solexa’s encoding is different from the Sanger standard:



```
Sanger
     !”#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHI
;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefgh
     ^         ^         ^         ^         ^
     0        10        20        30        40
Solexa
```

- Most tools (e.g., Maq, Bowtie, BWA) expect Sanger scores by default, so you have to either convert the scores or tell the tool.
- Also, make sure, the tool does not use the old Solexa formula.

FASTQ and paired-end reads

Convention for paired-end runs:

The reads are reported two FASTQ files, such that the nth read in the first file is mate-paired to the nth read in the second file. The read IDs must match.

Alignment

Short read alignment: Task

Differences to conventional alignment:

- millions of very short reads, rather than a few long ones, have to be mapped to the genome
- dominant cause for mismatches is read errors, not substitutions
- base-call quality information (“phred scores”) are more important
- only small gaps are expected
- mate-paired reads require special handling
- SOLiD colour space mapping
- atypical reference sequence, e.g., bisulfite treatment

Alignment software

In the last two years, many tools for short-read alignments have been published:

- Eland
- Maq
- Bowtie
- Biostrings
- BWA
- SSAHA2, Soap, RMAP, SHRiMP, ZOOM, NovoAlign, Mosaik, Slider, ...

Which one is right for your task?

Short read alignment: Algorithms

Short-read aligners use one of these ideas to base their algorithm on:

- use spaced-seed indexing
  - hash seed words from the reference
  - hash seed words from the reads
- sort reference words and reads lexicographically
- use the Burrows-Wheeler transform (BWT)
- use the Aho-Corasick algorithm

BWT seems to be the winning idea (very fast, sufficiently accurate), and is used by the newest tools (Bowtie, SOAPv2, BWA).

Short read aligners: Differences

Alignment tools differ in

- speed
- suitability for use on compute clusters
- memory requirements
- sensitivity
    - Is a good match always found?
    - What is the maximum number of allowed mismatches?
    - Are small indels tolerated?
- ease of use
- available down-stream analysis tools
    - Are there other tools(SNP and indel callers, visualization tools, programming frameworks) that can deal with the tool's output format?

Short read aligners: Differences

Alignment tools also differ in whether they can

- make use of base-call quality scores
- estimate alignment quality
- work with paired-end data
- report multiple matches
- work with longer than normal reads
- match in colour space (for SOLiD systems)
- align data from methylation experiments
- deal with splice junctions

Popular alignment tools

- Eland (Solexa)
  - supplied by Ilumina as part of the SolexaPipeline
  - very fast
  - cannot make use of quality score
- Maq (Li *et al.*, Sanger Institute)
  - widely used
  - interpretes quality score and estimates alignment score
  - comes with downstream analysis tools (SNP, indel calling)
  - can deal with SOLiD colour space data
- Bowtie (Langmead *et al.*, Univ of Maryland) and BWA (Li et al., Sanger Institute)
  - new; based on Burrows-Wheeler transform
  - very fast, good accuracy
  - downstream tools available

Other commonly used aligners

- BWA (H. Li, Sanger Institute)
  - BWT-based
  - with gapped alignment (for indel calling)
  - Calculates alignment qualities
  - with module for longer reads: BWA-SW
- SSAHA, SSAHA2 (Sanger Institute)
  - one of the first short-read aligners
  - SSAHA2 still widely used for 454 alignment
- SOAP and SOAP2 (Beijing Genomics Institute)
  - with downstream tools
  - SOAP2 uses BWT
- NovoAlign
  - commercial, very good sensitivity

Paired-end alignment

When aligning paired-end data, the aligner can use the information that mate-paired reads have a known separation:

- Try to align the reads individually
- Then, for each aligned read, attempt to align the mate in a small window near the first read's position with a more sensitive algorithm, e.g., Smith-Waterman to allow for gaps.
  - Be sure to tell the aligner the minimal and maximal separation.
- This helps to find small and large indels and other structural variants.

The SAM format and the SAMtools

EMBL

Aligner output formats

- Most aligners use their own format to output the alignments.
- Hence, downstream tools cannot be exchanged between aligners.

- To resolve this issue, Li et al. have suggested a standardized file format:

  the Sequence Alignment/Map (SAM) format
- SAM is increasingly used in newest tools.
- Converters from legacy formats are included with the SAMtools.

A SAM file

```
[...]

HWI-EAS225_309MTAAXX:5:1:689:1485    0          XIII        863564     25          36M         *
        0           0           GAAATATATACGTTTTTATCTATGTTACGTTATATA
CCCCCCCCCCCCCCCCCCCCCCC4CCCB4CA?AAA<   NM:i:0      X0:i:1      MD:Z:36

HWI-EAS225_309MTAAXX:5:1:689:1485    16         XIII        863766     25          36M         *
        0           0           CTACAATTTTGCACATCAAAAAAGACCTCCAACTAC
=8A=AA784A9AA5AAAAAAAAAAA=AAAAAAAAAA   NM:i:0      X0:i:1      MD:Z:36

HWI-EAS225_309MTAAXX:5:1:394:1171    0          XII         525532     25          36M         *
        0           0           GTTTACGGCGTTGCAAGAGGCCTACACGGGCTCATT
CCCCCCCCCCCCCCCCCCCCC?CCACCACA7?<???   NM:i:0      X0:i:1      MD:Z:36

HWI-EAS225_309MTAAXX:5:1:394:1171    16         XII         525689     25          36M         *
        0           0           GCTGTTATTTCTCCACAGTCTGGCAAAAAAAAGAAA         7AAAAAA?
AA<AA?AAAAA5AAA<AAAAAAAAAAAA    NM:i:0      X0:i:1      MD:Z:36

HWI-EAS225_309MTAAXX:5:1:393:671     0          XV          440012     25          36M         *
        0           0           TTTGGTGATTTTCCCGTCTTTATAATCTCGGATAAA
AAAAAAAAAAAAAAA<AAAAAAAA<AAAA5<AAAA3   NM:i:0      X0:i:1      MD:Z:36

HWI-EAS225_309MTAAXX:5:1:393:671     16         XV          440188     25          36M         *
        0           0           TCATAGATTCCATATGAGTATAGTTACCCCATAGCC         ?9A?A?CC?
<ACCCCCCCCCCCCCCCCACCCCCCC    NM:i:0      X0:i:1      MD:Z:36

[...]
```

The SAM format

A SAM file consists of two parts:

- Header
  - contains meta data (source of the reads, reference genome, aligner, etc.)
  - Most current tools omit and/or ignore the header.
  - All header lines start with “@”.
  - Header fields have standardized two-letter codes for easy parsing of the information
- Alignment section
  - A tab-separated table with at least 11 columns
  - Each line describes one alignment

SAM format: Alignment section

The columns are:

- QNAME: ID of the read (“query”)
- FLAG: alignment flags
- RNAME: ID of the reference (typically: chromosome name)
- POS: Position in reference (1-based, left side)
- MAPQ: Mapping quality (as Phred score)
- CIGAR: Alignment description (gaps etc.) in CIGAR format
- MRNM: Mate reference sequence name [for paired end data]
- MPOS: Mate position [for paired end data]
- ISIZE: inferred insert size [for paired end data]
- SEQ: sequence of the read
- QUAL: quality string of the read
- extra fields

SAM format: Flag and extra fields

FLAG field: A number, encoding

- whether the read is from a paired-end run, and if so, which one
- if so, whether the read and/or its mate are mapped
- whether the read mapped to the forward or the reverse strand
- whether the read passed platform quality checks
- [and a few more things]

Extra fields:

- Always triples of the format TAG : VTYPE : VALUE
- may encode number of mismatches (“NM”), number of alignments for the same read, extra informations on quality, aligner-specific data etc.

SAM format: extended CIGAR strings

Alignments contain gaps (e.g., in case of an indel, or, in RNA-Seq, when a read straddles an intron).

Then, the CIGAR string gives details.

Example: “M10 I4 M4 D3 M12” means

- the first 10 bases of the read map (“M10”) normally (not necessarily perfectly)
- then, 4 bases are inserted (“I4”), i.e., missing in the reference
- then, after another 4 mapped bases (“M4”), 3 bases are deleted (“D4”), i.e., skipped in the query.
- Finally, the last 12 bases match normally.

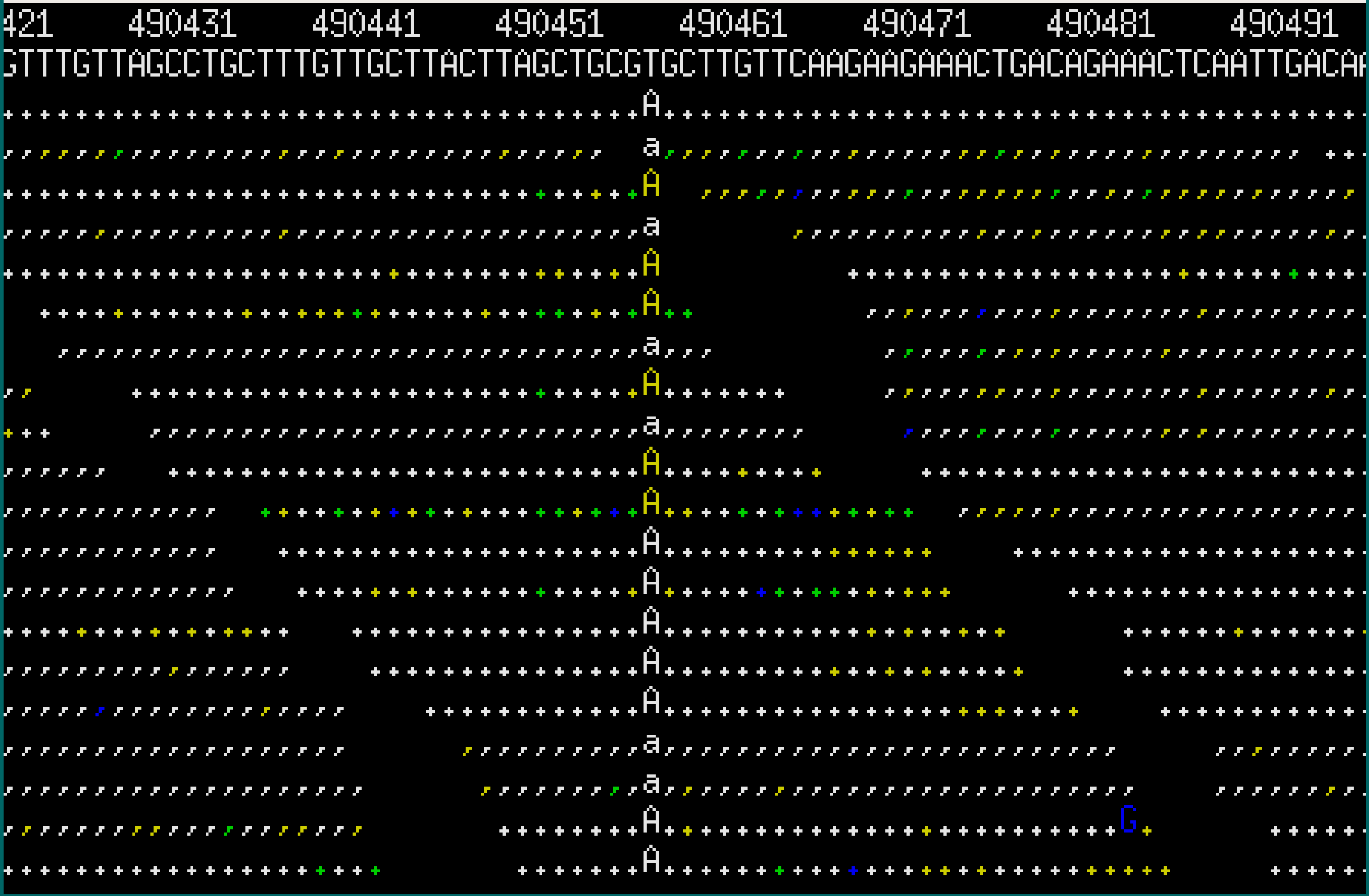
There are further codes (N, S, H, P), which are rarely used.

SAMtools

- The SAMtools are a set of simple tools to
  - convert between SAM and BAM
    - SAM: a human-readable text file
    - BAM: a binary version of a SAM file, suitable for fast processing
  - sort and merge SAM files
  - index SAM and FASTA files for fast access
  - view alignments (“tview”)
  - produce a “pile-up”, i.e., a file showing
    - local coverage
    - mismatches and consensus calls
    - indels

- The SAMtools C API facilitates the development of new tools for processing SAM files.

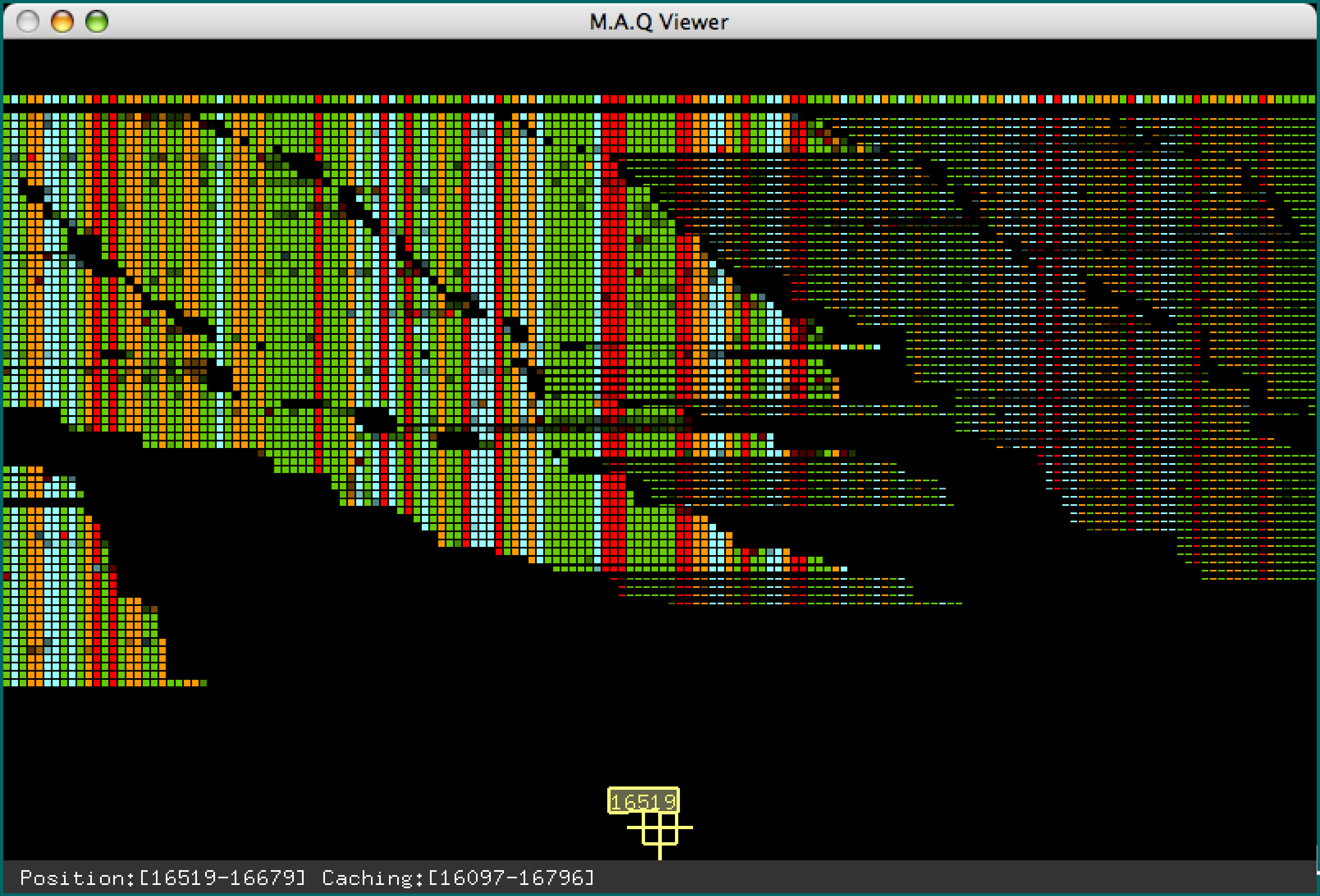
Screenshot of SAMtools tview



```
421    490431    490441    490451    490461    490471    490481    490491
GTTTGTTAGCCTGCTTTGTTGCTTACTTAGCTGCGTGCTTGTTCAAGAAGAAACTGACAGAAACTCAATTGACAA
...................................A.......................................
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,  a,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,, ...
...................................A  ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,a       ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
...................................A          .............................
  .................................A..         ,,,,,,,,,,,,,,,,,,,,,,,,,,,,
   ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,a,,,         ,,,,,,,,,,,,,,,,,,,,,,,,,,,
,,     ............................A.......     ,,,,,,,,,,,,,,,,,,,,,,,,,,,
...     ,,,,,,,,,,,,,,,,,,,,,,,,,,,a,,,,,,,,     ,,,,,,,,,,,,,,,,,,,,,,,,,,
,,,,,,   ..........................A.........     .........................
,,,,,,,,,,,,  .....................A..............  ,,,,,,,,,,,,,,,,,,,,,,,
,,,,,,,,,,,,   ....................A...............    ....................
,,,,,,,,,,,,,   ...................A................      .................
................   ................A...................      ..............
,,,,,,,,,,,,,,,,    ...............A....................     ..............
,,,,,,,,,,,,,,,,,,,    ............A.......................    ............
,,,,,,,,,,,,,,,,,,,      ,,,,,,,,,,a,,,,,,,,,,,,,,,,,,,,,,,,,     ,,,,,,,,,
,,,,,,,,,,,,,,,,,,,,      ,,,,,,,,,a,,,,,,,,,,,,,,,,,,,,,,,,,,    ,,,,,,,,,
,,,,,,,,,,,,,,,,,,,,       ........A.........................G.      ......
.....................       .......A............................     ......
```

EMBL

MaqView: Another alignment viewer



Position:[16519-16679] Caching:[16097-16796]

SAMtools pileup output

```
I  25514  G  G  42  0   25  5   ....^:.                   CCCCC
I  25515  T  T  42  0   25  5   .....                     CC?CC
I  25516  A  G  48  48  25  7   GGGGG^:G^:g               CCCCCC5
I  25517  G  G  51  0   25  8   ......,^:,                CCCCCC1?
I  25518  T  T  60  0   25  11  ......,,^:.^:,^:,         CCCCCC3A<:;
I  25519  T  T  60  0   25  11  ......,,.,,               CCCCCC>A@AA
I  25520  G  G  60  0   25  11  ......,,.,,               CCCACC>A@<A
I  25521  T  T  60  0   25  11  ......,,.,,               CCCCCC?ACAA
I  25522  A  A  60  0   25  11  ......,,.,,               CCCCCC>ACAA
I  25523  A  A  72  0   25  15  ......,,.,,^:.^:,^:,^:.   CCCCCC;ACAAC??C
I  25524  C  C  72  0   25  15  ......,,.,,.,,.           CCCCCC6<<A?C=9C
I  25525  C  C  56  0   24  18  ......,,.,,.,,.^:,^!.^:T CCCCCC>ACA?C=AC<CC
I  25526  A  A  81  0   24  18  ......,,.,,.,,.,..       CCCCCC>ACAACAACACC
I  25527  A  A  56  0   24  18  ......,,.,,.,,.,.G       CCCCCC?ACAA@A?CACC
```

Fields: chromosome, position, reference base, consensus base, consensus quality, SNP quality, maximum mapping quality, coverage, base pile-up, base quality pile-up

Coverage vectors

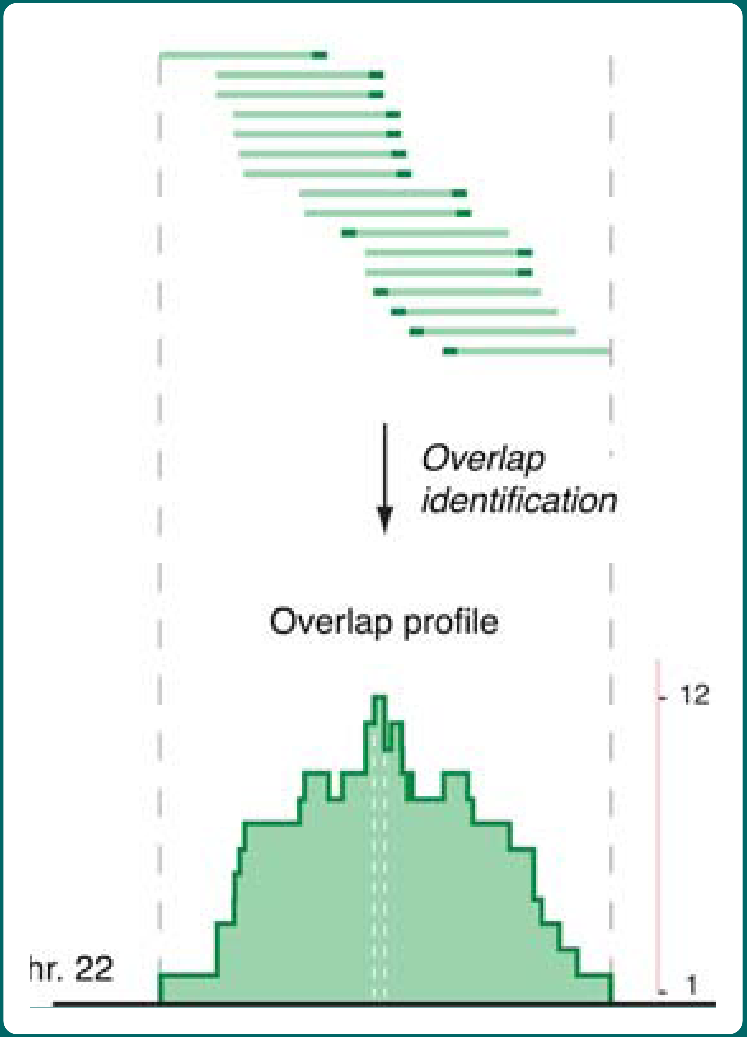
EMBL

Coverage

- In resequencing, we hope to sequence uniformly, i.e., see each part of the genome represented by the same amount of reads.
- Due to the random nature of shotgun sequencing, we need to "cover the genome several times" in order to see each position at least once.

- In other techniques (ChIP-Seq, RNA-Seq, Tag-Seq, CNV-Seq, etc.), the local coverage is what we are interested in.

Coverage vectors



<-- Solexa reads,
aligned to genome

<-- coverage vector

Figure taken from Zhang et al., PLoS Comp. Biol. 2008

EMBL-EBI

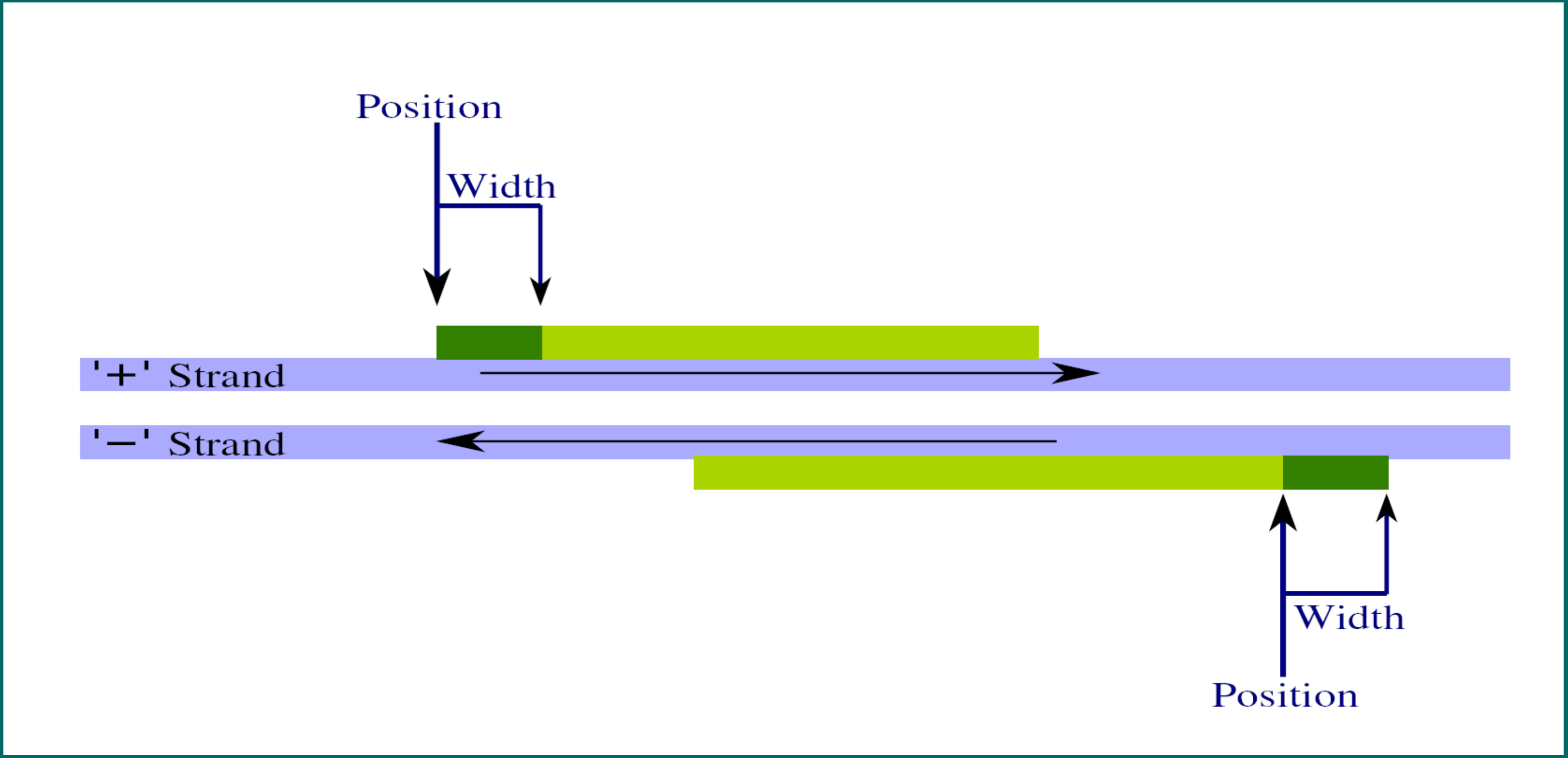
Coverage vectors

- A coverage (or: “pile-up”) vector is an integer vector with on element per base pair in a chromosome, tallying the number of reads (or fragments) mapping onto each base pair.
- It is the essential intermediate data type in assays like ChIP-Seq or RNA-Seq

- One may ever count the coverage by the reads themselves, or extend to the length of the fragments

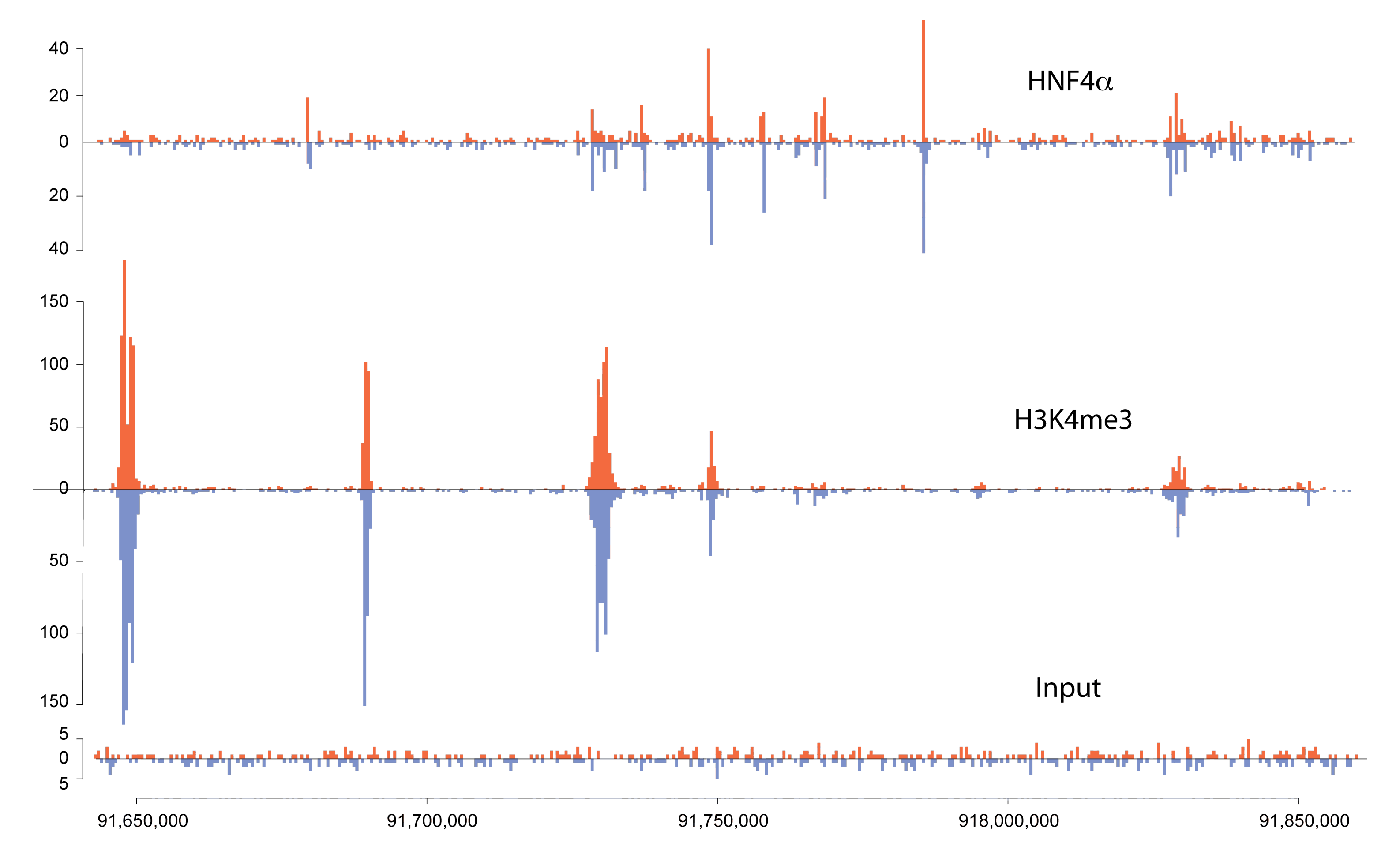
Calculating coverage vectors

Extending reads to fragments:



EMBL

Chip-Seq coverage: examples



EMBL

Figure courtesy of Christiana Spyrou (CR UK)

The issue with multiple reads

If one finds several reads with the exact same sequenche, does this mean

- that many fragments from this locus were precipitated and often got got cut at the exact same place, or
- that there was only a single fragment, but it was amplified more efficiently than fragments from other loci in the PCR (or more efficiently transcribed to cDNA)?
  - If you consider the latter more likely, you should count these reads only once. However, this dramatically compresses your dynamic range.

Ambiguous matches and mappability

- If a read matches at several places in the reference, the best match should be used.
- If there are several equally good matches, an aligner may
  - chose an alignment at random
  - discard the read
  - report all alignments and delay the choice to downstream analysis
- It is useful to know which regions in the genome are repetitive on the scale of the read length and hence give rise to alignment ambiguities.

EMBL

*

EMBL