Introduction to Bioinformatics

Chris Miller, Ph.D.
Washington University in St. Louis

Bioinformatics Workshop 2024-2025

(aka bfx-workshop)

Applied Bioinformatics for Genomics II

(aka BIOL.5625.01)

Bioinformatics Workshop 2024-2025

Supported by – ICTS Precision Health

- We aim to catalyze genomic research by providing grant review, development services, guidance and resources for genomic researchers and genomics education in the community.

Cite the **NIH CTSA Grant #UL1 TR002345** when research is supported by ICTS/CTSA funding or any ICTS Core Services

BFX Workshop – contact Jenny if you haven't received the following

- Slack access, welcome email, Outlook bfx-workshop-2024 group invite

Register for BFX

https://redcap.link/BFX2024

icts-precisionhealth.wustl.edu

johnegarza@wustl.edu

Applied Bioinformatics for Genomics II

Course: BIOL.5625.01

1 Credit Hour DBBS course

- **50% grade**: Attendance
  - 75% (10 lectures) must be in person
  - 3 can be viewed via recordings

- **50% grade**: Assignments
  - choose 8 of the 10 assignments
  - due by the end of the second Friday after the lecture

Register for BFX

https://redcap.link/BFX2024

icts-precisionhealth.wustl.edu

johnegarza@wustl.edu

Who we are, and why you should trust us

Chris Miller, Ph.D.

Course Director
Associate Professor
Division of Oncology

John Garza

Course Coordinator/TA
Bioinformatics/Genome Analytics
Programmer

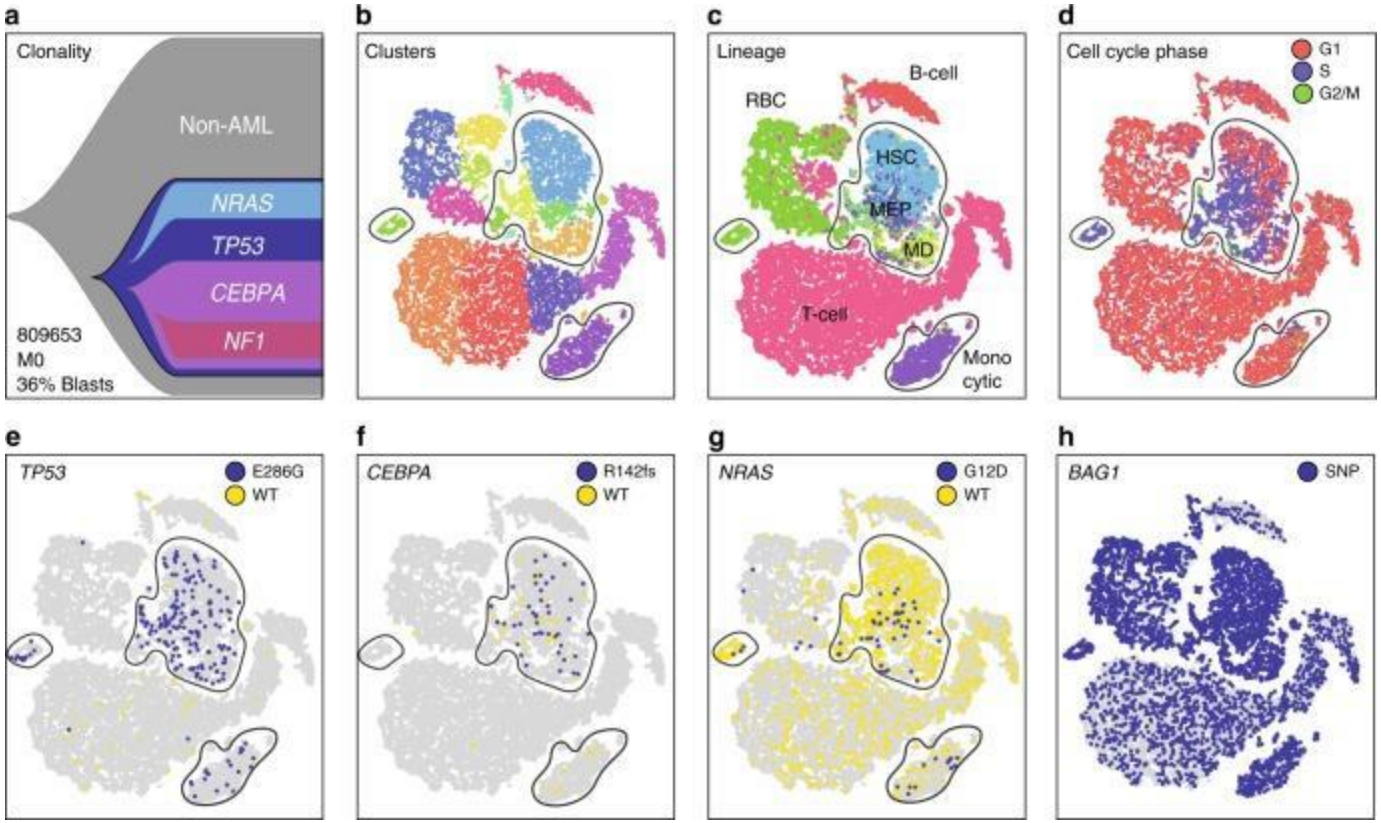
20 years of experience in
Bioinformatics and Computational Biology

Other Lecturers/Organizers include:

Jason Walker
Obi Griffith
Jennifer Foltz
Juan Macias
Brigida Rusconi

Why learn bioinformatics?

- Biology is now a quantitative discipline - especially genomics


a
Clonality
Non-AML
NRAS
TP53
CEBPA
NF1
809653
M0
36% Blasts
b
Clusters
c
Lineage
B-cell
RBC
HSC
MEP
MD
T-cell
Mono
cytic
d
Cell cycle phase
G1
S
G2/M
e
TP53
E286G
WT
f
CEBPA
R142fs
WT
g
NRAS
G12D
WT
h
BAG1
SNP

Why learn bioinformatics?

- Biology is now a quantitative discipline - especially genomics
- Skills in programming, statistics, and visualization help you get the most out of your data

People who need complex data analysis

People who know how to do complex data analysis

https://hellogiggles.com/news/how-many-people-attend-coachella/
https://www.nytimes.com/2020/07/02/theater/germany-theater-coronavirus.html

Why learn bioinformatics?

- Biology is now a quantitative discipline - especially genomics
- Skills in programming, statistics, and visualization help you get the most out of your data
- We’re aiming to teach you the theory and practice of computational biology, with a focus on genomics but lessons that apply broadly

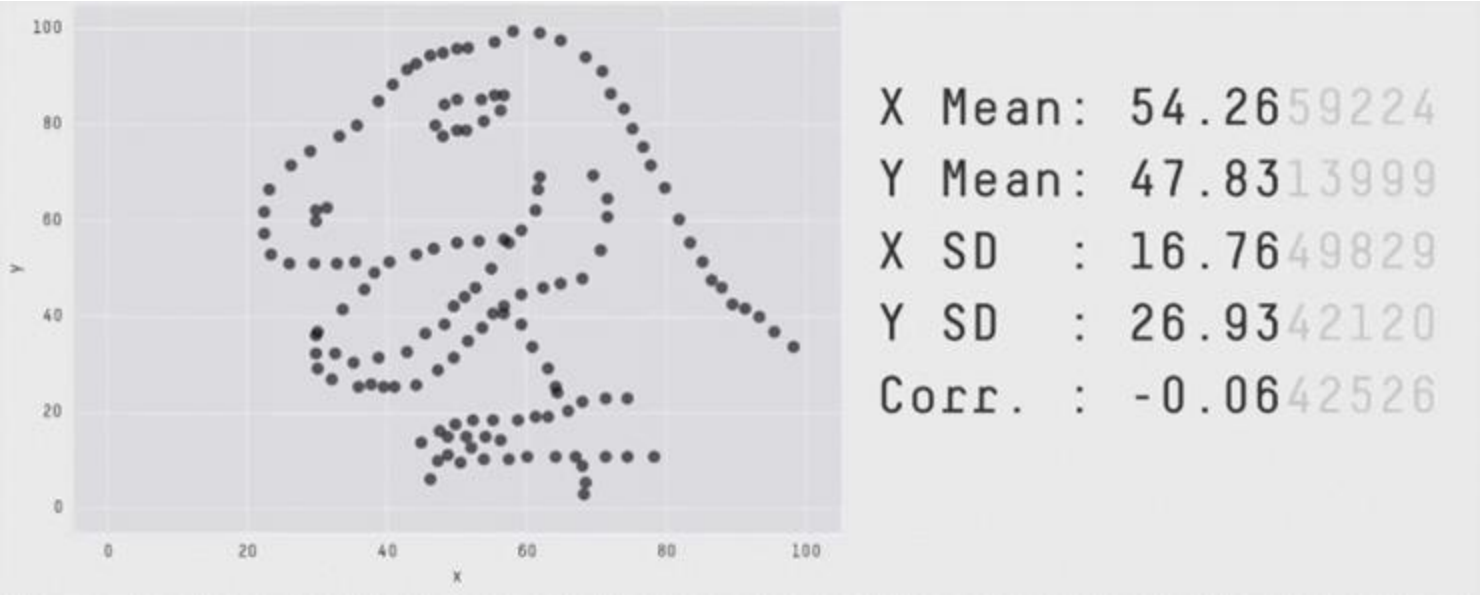
Goals:

- To empower you to improve and expedite your research
- To expose you to new ideas and techniques that may advance your research program

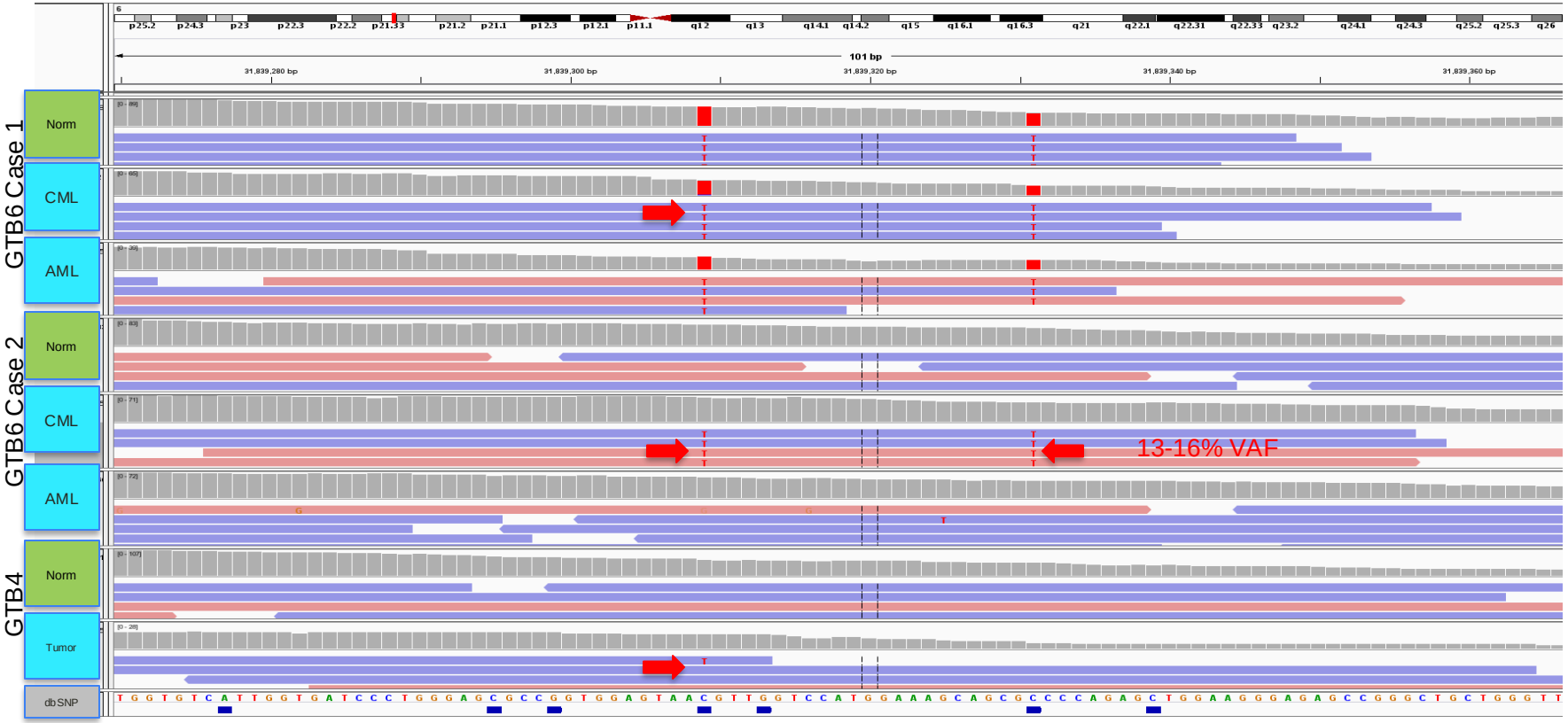
Don’t trust your data

Summary statistics are dangerous

- Visualize your data!
- A picture is worth a thousand p-values



Contamination of CML samples



Watch out!

- Computational analyses require controls too!
- Look at the data and understand it's limitations!
- Don't assume that the data is clean – prove to yourself that it is!

Expectations:

- Check the prerequisites from fall weeks 1-3. Install the software, be familiar with the unix command line, know how to use docker to launch analyses
  - https://github.com/genome/bfx-workshop/tree/master/lectures/week_01

- Most of you are new to computational analysis – *ask questions*!

- Work hard, follow along, and get your money's worth from this course

- The folks teaching and the TAs all know their stuff, *ask questions*!

Course Structure:

- Weekly lecture introducing topic
- Practical exercise allowing you to apply that knowledge
- https://github.com/genome/bfx-workshop
- ICTS Slack instance: #bfx-workshop channel
- Email list: bfx-workshop-2023@gowustl.onmicrosoft.com

Genome arithmetic and bedtools

some slides adapted from:

Aaron Quinlan's Applied Computational Genomics course
https://github.com/quinlan-lab/applied-computational-genomics

The Griffith Lab's RNAbio course:
https://rnabio.org/

A common class of problems:

I have a list of SNPs in this patient. Which ones hit known enhancers?

Did my sequencing run give good coverage over the genes I care about?

My experiment returned 300 ChIP-seq peaks. Which ones are near cancer-related genes?

What is a genome interval?

At minimum defined by **chromosome, start, and stop**

may also have strand, name, other fields

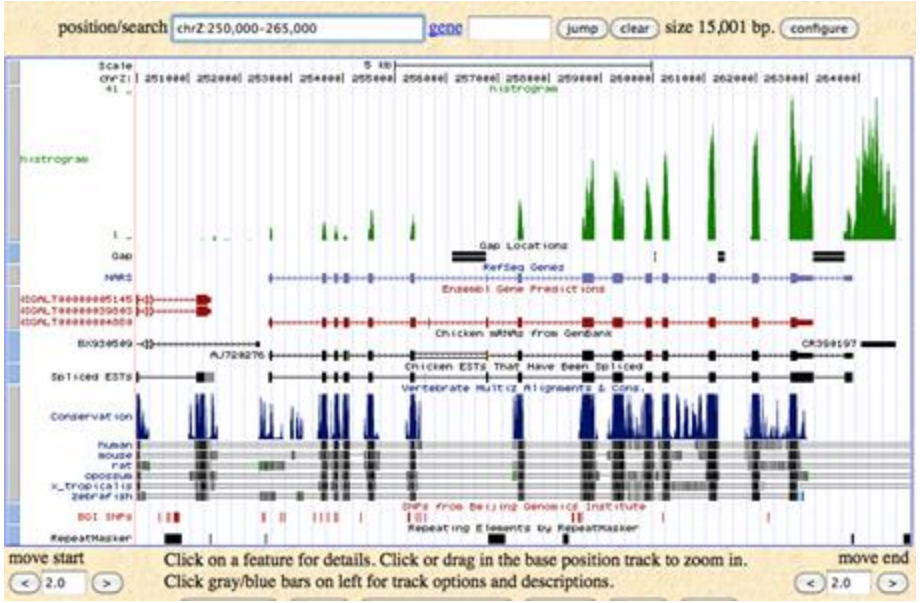
```
chr1:12345-98765

chrX    12345     98765         -

chr17   7661779   7687550   ENSG00000141510    TP53    -
```

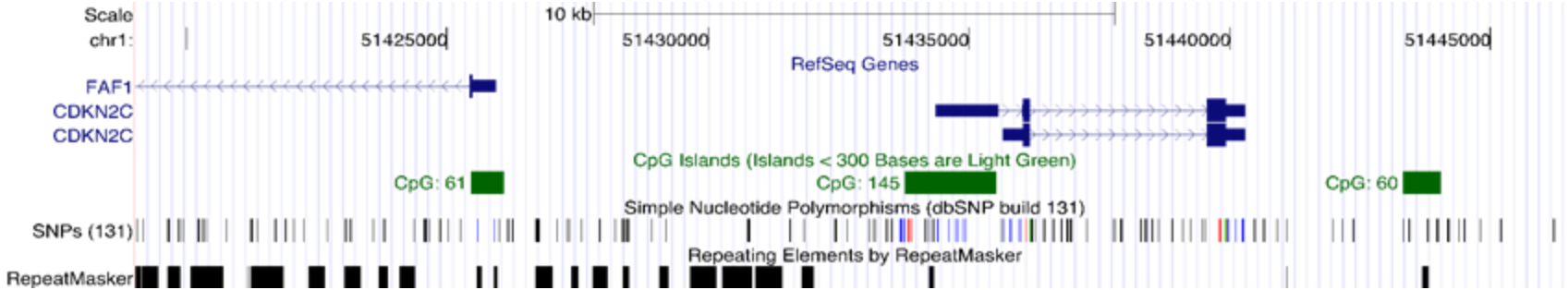
Examples of genome intervals

- Genes: exons, introns, UTRs, promoters (BED, GFF, GTF)
- Conservation (BEDGRAPH)
- Genetic variation (VCF)
- Sequence alignments (BAM)
- Transcription factor binding sites (BED, BEDGRAPH)
- CpG islands (BED)
- Segmental duplications (BED)
- Chromatin annotations (BED)
- Gene expression data (WIG, BIGWIG, BEDGRAPH)



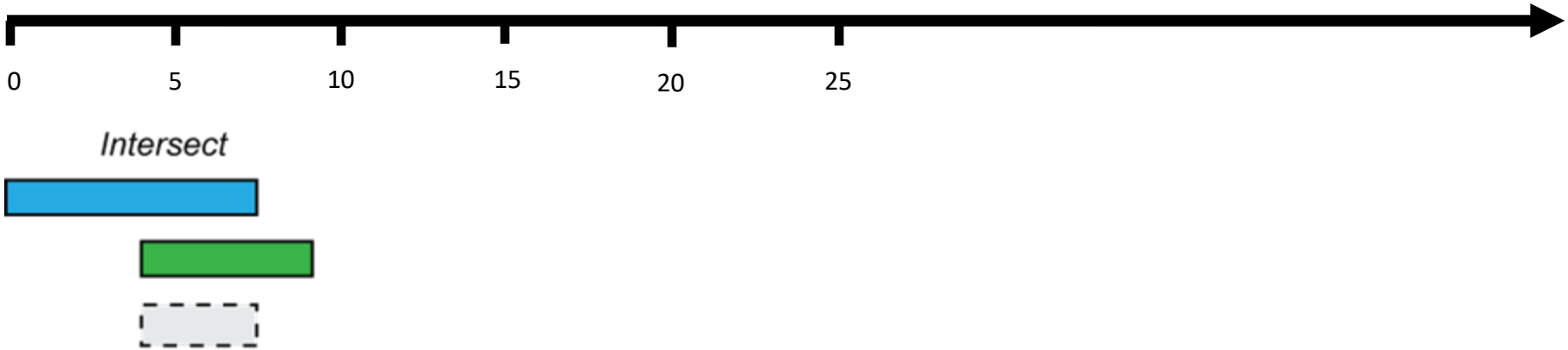
Your own observations: put them in context

Genome arithmetic

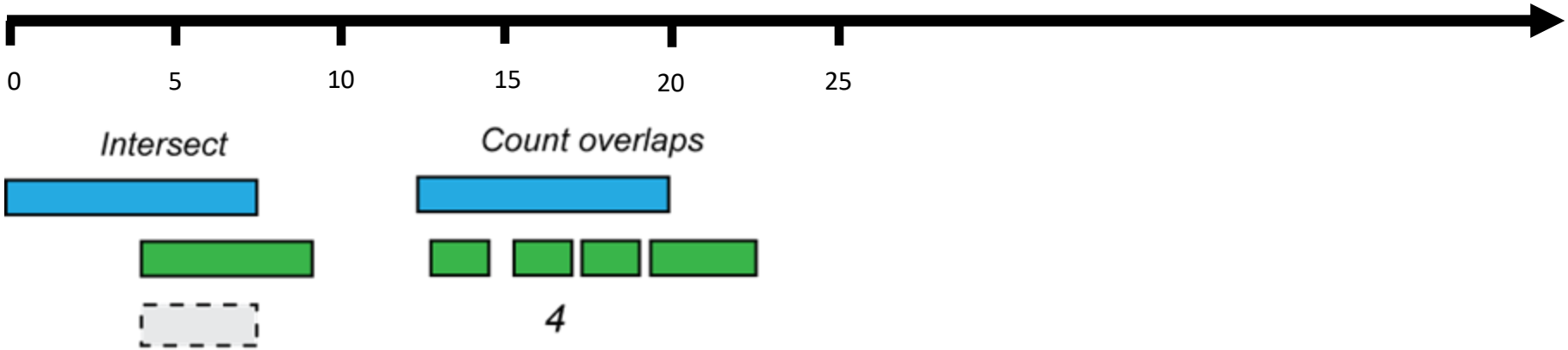


the method of comparing, contrasting,
and gaining insight using multiple genome interval files

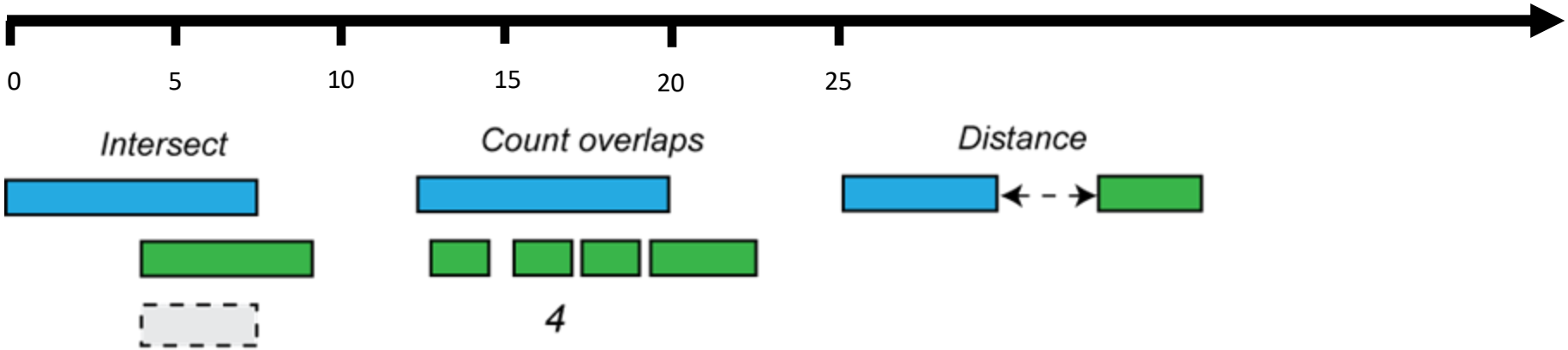
Genome arithmetic operations



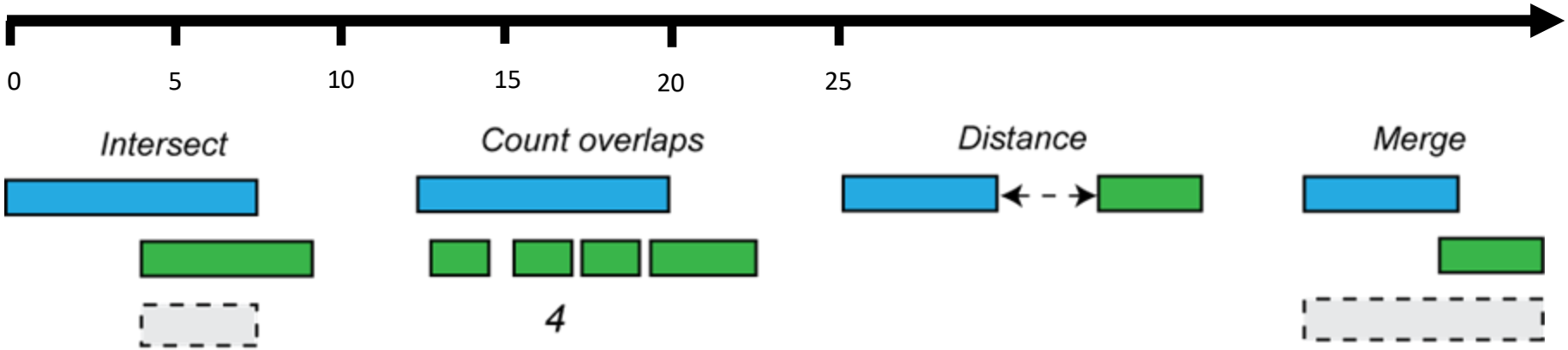
Genome arithmetic operations



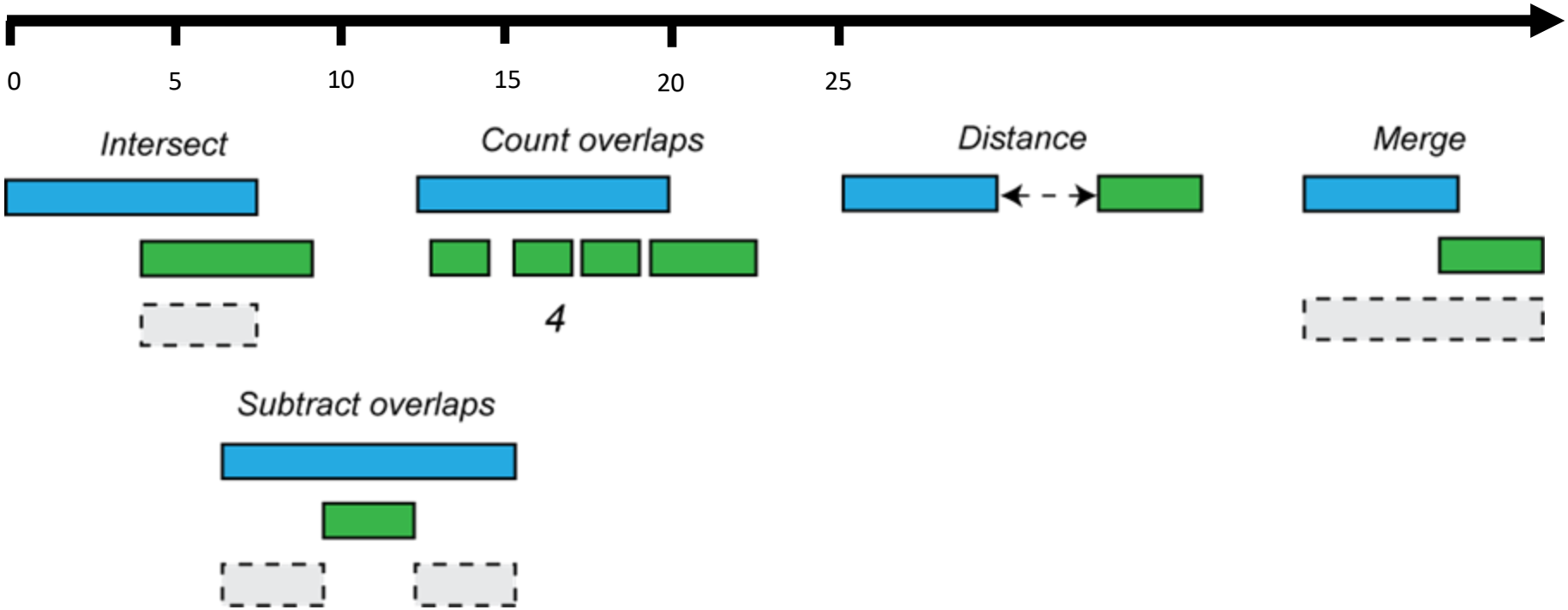
Genome arithmetic operations



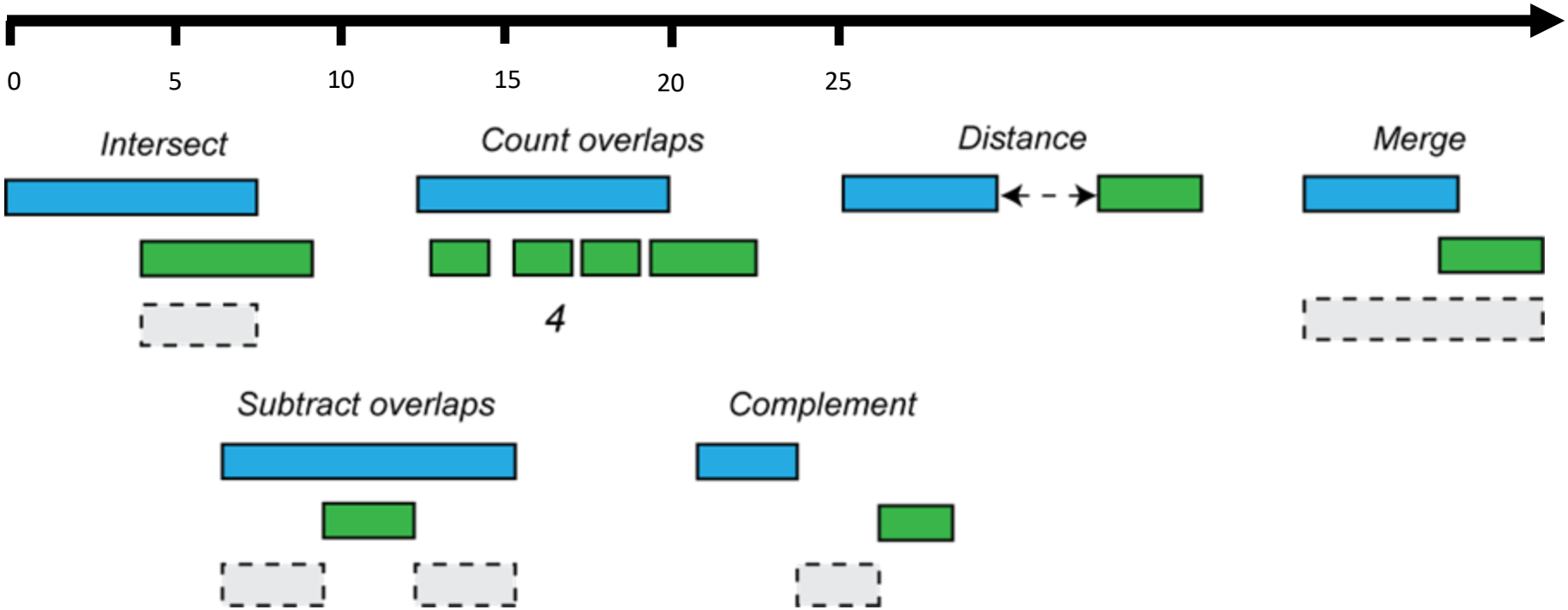
Genome arithmetic operations



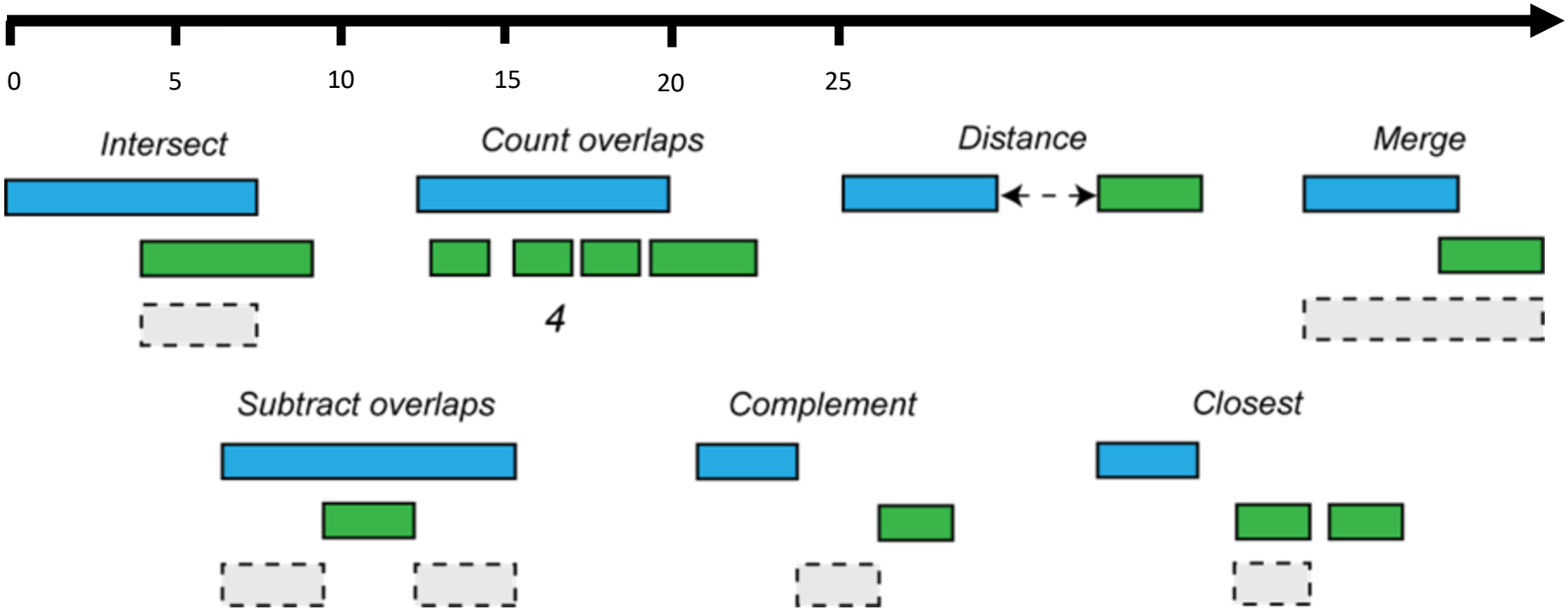
Genome arithmetic operations



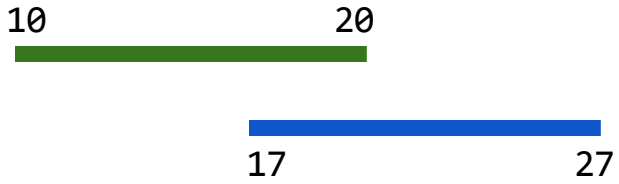
Genome arithmetic operations

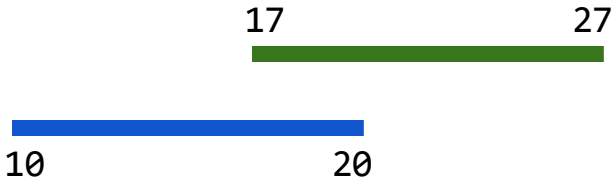


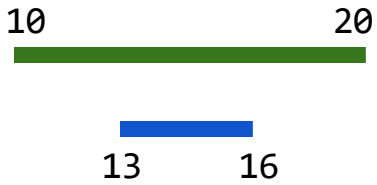
Genome arithmetic operations

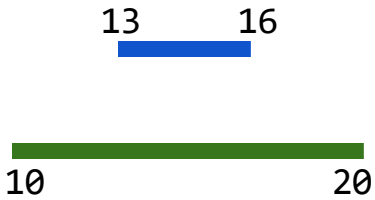


Do two intervals intersect (overlap)?



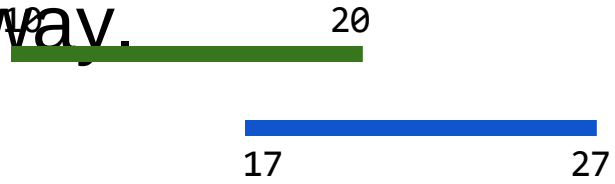


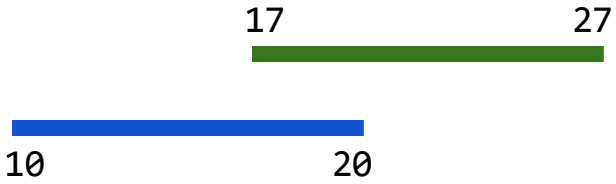


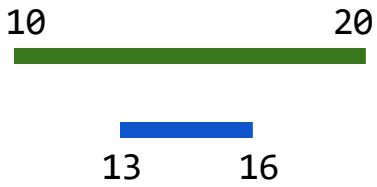


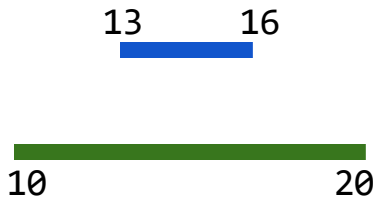

```
if ((a.start <= b.start and a.end >= b.start) or
    (b.start <= a.start and b.end >= a.start) or
    (a.start <= b.start and a.end >= b.end)   or
    (b.start <= a.start and b.end >= a.end))
{
  INTERSECTION!!!
}
else NADA!!!
```

Do two intervals intersect (overlap)? A simpler way.







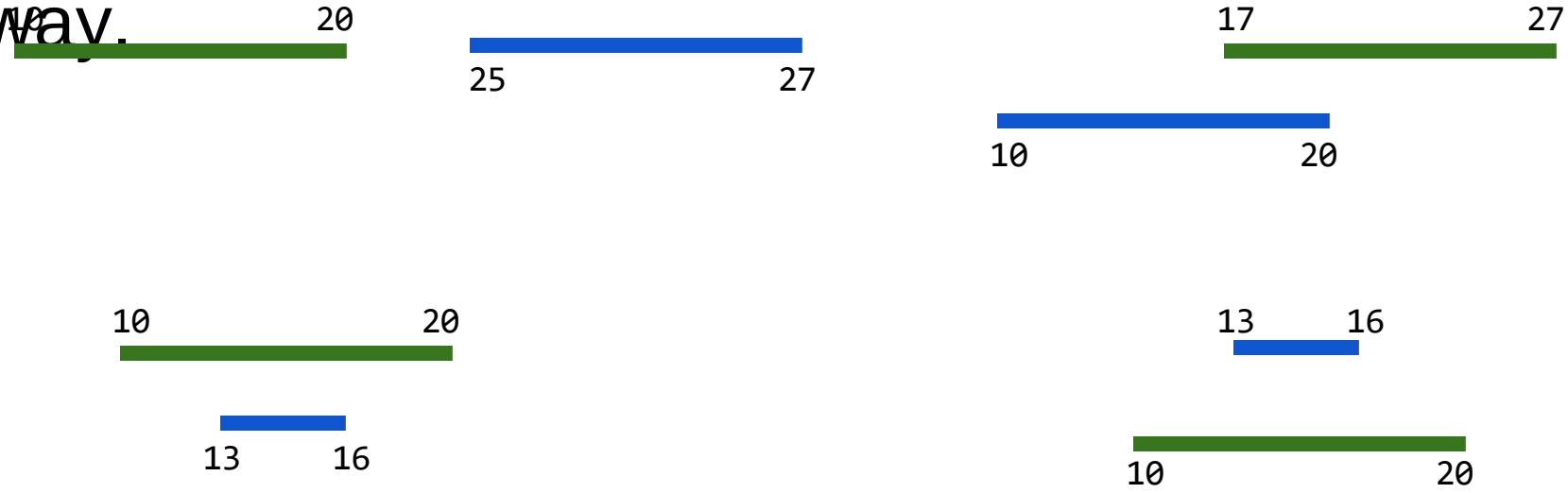



```
I = min(a.end, b.end) - max(a.start, b.start)

        if I > 0, intersection,
  if I <= 0, distance between the intervals

      = min(20, 27) - max(10, 17)
            = 20-17 = 3
```

Do two intervals intersect (overlap)? A simpler way.




```
I = min(a.end, b.end) - max(a.start, b.start)

        if I > 0, intersection,
  if I <= 0, distance between the intervals

      = min(20, 27) - max(10, 17)
            = 20-17 = 3
```

Unsorted data is slow!

```
chr8    94925972    94949411
chr2    24077433    24085861
chr20   46684365    46689779
chr20   45372563    45407889
chr20   34704290    34713439
chr17   7661779
7687550
chr17   29566052    29573157
chr15   43403061    43510728
chr1    223779899   223845972
chr11   44885903    44951306
chr11   128934731   128943399
```

```
chr11   128934731   128943399
chr20   45372563    45407889
chr1    223779899   223845972
chr15   43403061    43510728
chr20   46684365    46689779
chr2    24077433    24085861
chr20   34704290    34713439
chr17   7661779      7687550
chr17   29566052    29573157
chr8    94925972    94949411
chr11   44885903    44951306
```

Unsorted data is slow!

chr8	94925972	94949411
chr2	24077433	24085861
chr20	46684365	46689779
chr20	45372563	45407889
chr20	34704290	34713439
chr17	7661779	7687550
chr17	29566052	29573157
chr15	43403061	43510728
chr1	223779899	223845972
chr11	44885903	44951306
chr11	128934731	128943399

chr11	128934731	128943399
chr20	45372563	45407889
chr1	223779899	223845972
chr15	43403061	43510728
chr20	46684365	46689779
chr2	24077433	24085861
chr20	34704290	34713439
chr17	7661779	7687550
chr17	29566052	29573157
chr8	**94925972**	**94949411**
chr11	44885903	44951306

Unsorted data is slow!

```
chr8    94925972    94949411
chr2    24077433    24085861
chr20   46684365    46689779
chr20   45372563    45407889
chr20   34704290    34713439
chr17   7661779
7687550
chr17   29566052    29573157
chr15   43403061    43510728
chr1    223779899   223845972
chr11   44885903    44951306
chr11   128934731   128943399
```

```
chr11   128934731   128943399
chr20   45372563    45407889
chr1    223779899   223845972
chr15   43403061    43510728
chr20   46684365    46689779
chr2    24077433    24085861
chr20   34704290    34713439
chr17   7661779      7687550
chr17   29566052    29573157
chr8    94925972    94949411
chr11   44885903    44951306
```

Unsorted data is slow!

chr8	94925972	94949411
chr2	24077433	24085861
chr20	46684365	46689779
chr20	45372563	45407889
chr20	34704290	34713439
chr17	7661779	7687550
chr17	29566052	29573157
chr15	43403061	43510728
chr1	223779899	223845972
chr11	44885903	44951306
chr11	128934731	128943399

chr11	128934731	128943399
chr20	45372563	45407889
chr1	223779899	223845972
chr15	43403061	43510728
chr20	46684365	46689779
chr2	24077433	24085861
chr20	34704290	34713439
chr17	7661779	7687550
chr17	29566052	29573157
chr8	**94925972**	**94949411**
chr11	44885903	44951306

Unsorted data is slow!

```
chr8    94925972    94949411
chr2    24077433    24085861
chr20   46684365    46689779
chr20   45372563    45407889
chr20   34704290    34713439
chr17   7661779
7687550
chr17   29566052    29573157
chr15   43403061    43510728
chr1    223779899   223845972
chr11   44885903    44951306
chr11   128934731   128943399
```

```
chr11   128934731   128943399
chr20   45372563    45407889
chr1    223779899   223845972
chr15   43403061    43510728
chr20   46684365    46689779
chr2    24077433    24085861
chr20   34704290    34713439
chr17   7661779      7687550
chr17   29566052    29573157
chr8    94925972    94949411
chr11   44885903    44951306
```

Unsorted data is slow!

```
chr8   94925972   94949411
chr2   24077433   24085861
chr20  46684365   46689779
chr20  45372563   45407889
chr20  34704290   34713439
chr17  7661779
7687550
chr17  29566052   29573157
chr15  43403061   43510728
chr1   223779899  223845972
chr11  44885903   44951306
chr11  128934731  128943399
```



```
chr11  128934731  128943399
chr20  45372563   45407889
chr1   223779899  223845972
chr15  43403061   43510728
chr20  46684365   46689779
chr2   24077433   24085861
chr20  34704290   34713439
chr17  7661779     7687550
chr17  29566052   29573157
chr8   94925972   94949411
chr11  44885903   44951306
```

Unsorted data is slow!

```
chr8    94925972    94949411
chr2    24077433    24085861
chr20   46684365    46689779
chr20   45372563    45407889
chr20   34704290    34713439
chr17   7661779
7687550
chr17   29566052    29573157
chr15   43403061    43510728
chr1    223779899   223845972
chr11   44885903    44951306
chr11   128934731   128943399
```

```
chr11   128934731   128943399
chr20   45372563    45407889
chr1    223779899   223845972
chr15   43403061    43510728
chr20   46684365    46689779
chr2    24077433    24085861
chr20   34704290    34713439
chr17   7661779      7687550
chr17   29566052    29573157
chr8    94925972    94949411
chr11   44885903    44951306
```

Unsorted data is slow!

```
chr8    94925972    94949411
chr2    24077433    24085861
chr20   46684365    46689779
chr20   45372563    45407889
chr20   34704290    34713439
chr17   7661779
7687550
chr17   29566052    29573157
chr15   43403061    43510728
chr1    223779899   223845972
chr11   44885903    44951306
chr11   128934731   128943399
```



```
chr11   128934731   128943399
chr20   45372563    45407889
chr1    223779899   223845972
chr15   43403061    43510728
chr20   46684365    46689779
chr2    24077433    24085861
chr20   34704290    34713439
chr17   7661779      7687550
chr17   29566052    29573157
chr8    94925972    94949411
chr11   44885903    44951306
```

Sorting the files helps!

```
chr1    223779899   223845972
chr11   44885903    44951306
chr11   128934731   128943399
chr15   43403061    43510728
chr17   7661779
7687550
chr17   29566052    29573157
chr2    24077433    24085861
chr20   34704290    34713439
chr20   45372563    45407889
chr20   46684365    46689779
chr8    94925972    94949411
```

```
chr1    223779899   223845972
chr11   44885903    44951306
chr11   128934731   128943399
chr15   43403061    43510728
chr17   7661779
7687550
chr17   29566052    29573157
chr2    24077433    24085861
chr20   34704290    34713439
chr20   45372563    45407889
chr20   46684365    46689779
chr8    94925972    94949411
```

Sorting the files helps!

```
chr1    223779899   223845972
chr11   44885903    44951306
chr11   128934731   128943399
chr15   43403061    43510728
chr17   7661779
7687550
chr17   29566052    29573157
chr2    24077433    24085861
chr20   34704290    34713439
chr20   45372563    45407889
chr20   46684365    46689779
chr8    94925972    94949411
```

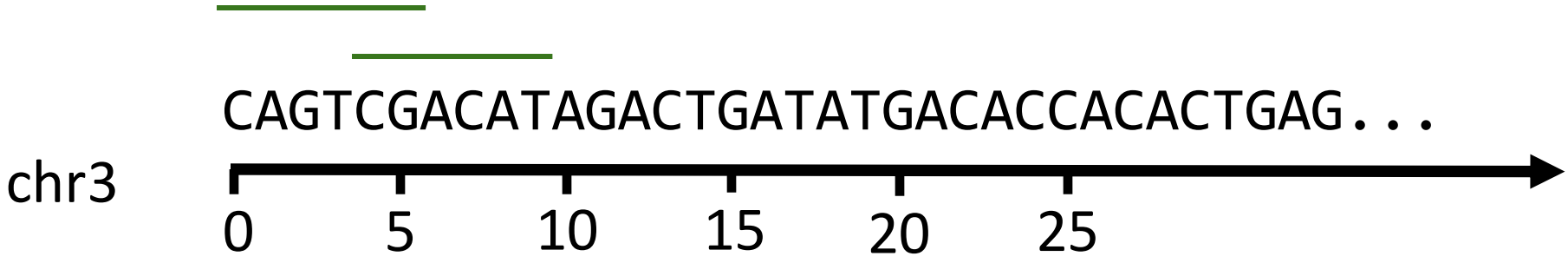
```
chr1    223779899   223845972
chr11   44885903    44951306
chr11   128934731   128943399
chr15   43403061    43510728
chr17   7661779
7687550
chr17   29566052    29573157
chr2    24077433    24085861
chr20   34704290    34713439
chr20   45372563    45407889
chr20   46684365    46689779
chr8    94925972    94949411
```

Sorting the files helps!

```
chr1    223779899   223845972
chr11   44885903    44951306
chr11   128934731   128943399
chr15   43403061    43510728
chr17   7661779
7687550
chr17   29566052    29573157
chr2    24077433    24085861
chr20   34704290    34713439
chr20   45372563    45407889
chr20   46684365    46689779
chr8    94925972    94949411
```

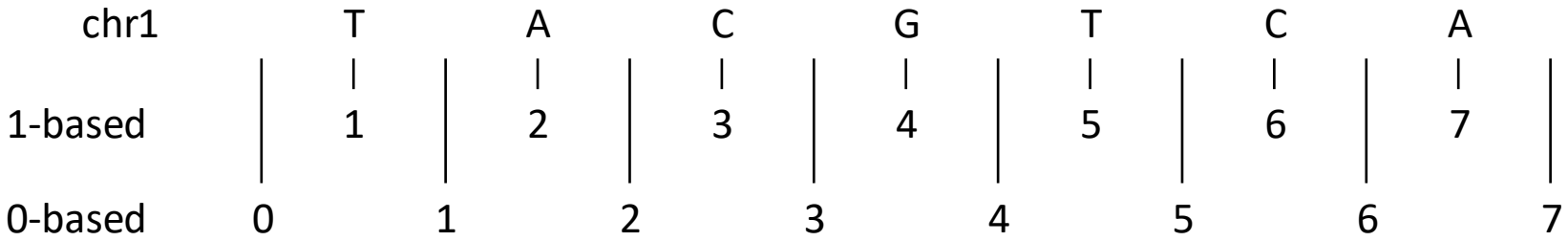
```
chr1    223779899   223845972
chr11   44885903    44951306
chr11   128934731   128943399
chr15   43403061    43510728
chr17   7661779
7687550
chr17   29566052    29573157
chr2    24077433    24085861
chr20   34704290    34713439
chr20   45372563    45407889
chr20   46684365    46689779
chr8    94925972    94949411
```

Genome arithmetic depends upon the genome coordinate system




```
chr3    0    7
chr3    4    10
```

Genomic coordinates – 1 vs 0 based



	1-based	0-based
Indicate a single nucleotide	chr1:4-4 G	chr1:3-4 G
Indicate a range of nucleotides	chr1:2-4 ACG	chr1:1-4 ACG
Indicate a single nucleotide variant	chr1:5-5 T/A	chr1:4-5 T/A

- 1-based : Single nucleotides, variant positions, or ranges are specified directly by their corresponding nucleotide numbers
  - GFF, SAM, VCF, Ensembl browser, …
- 0-based: Single nucleotides, variant positions, or ranges are specified by the coordinates that flank them
  - BED, BAM, UCSC browser, …

Genome builds

Reference Genome builds

Current human: GRCh38, hg38, b38

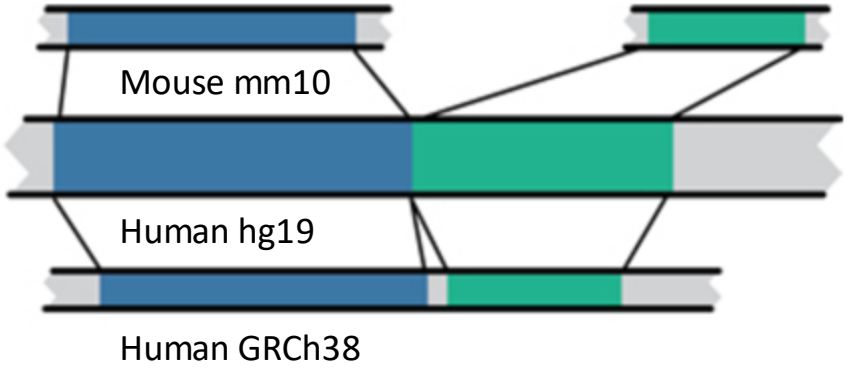
alternates: GRCh38v2_ccdg,
GRCh38_full_analysis_set_plus_decoy_hla

Previous human: GRCh37, hg19, b37

Current mouse: GRCm39, mm39
Previous mouse: GRCm38, mm10

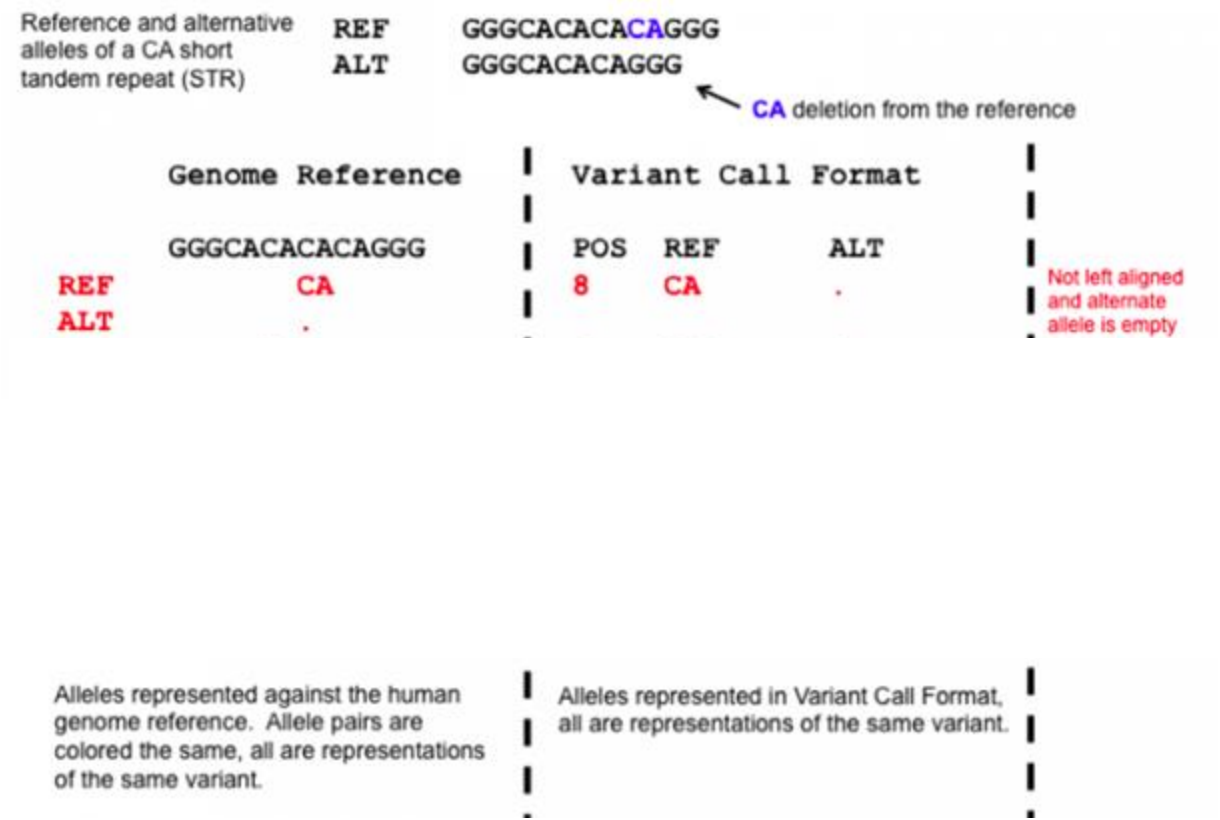
New human assembly: T2T-CHM13, pan-genomes

Lift-over




```
17  7661779 7687550 TP53   build 38
17  7565256 7590863 TP53   build 37
```

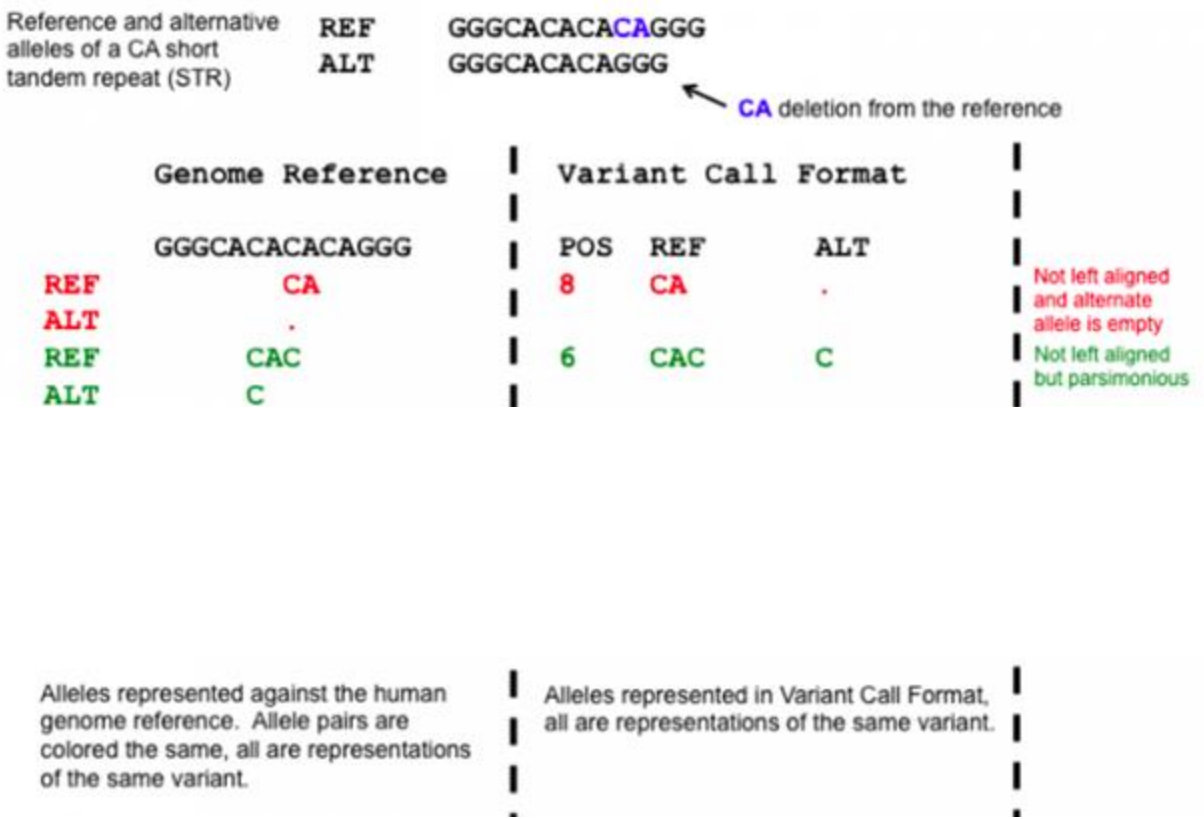
Variant shifting (alignment) and parsimony/trimming



Parsimony: representing variant in as few nucleotides as possible without reducing the length of any allele to 0

Left (right) aligning = shifting the start position of a variant as far to the left (right) as possible

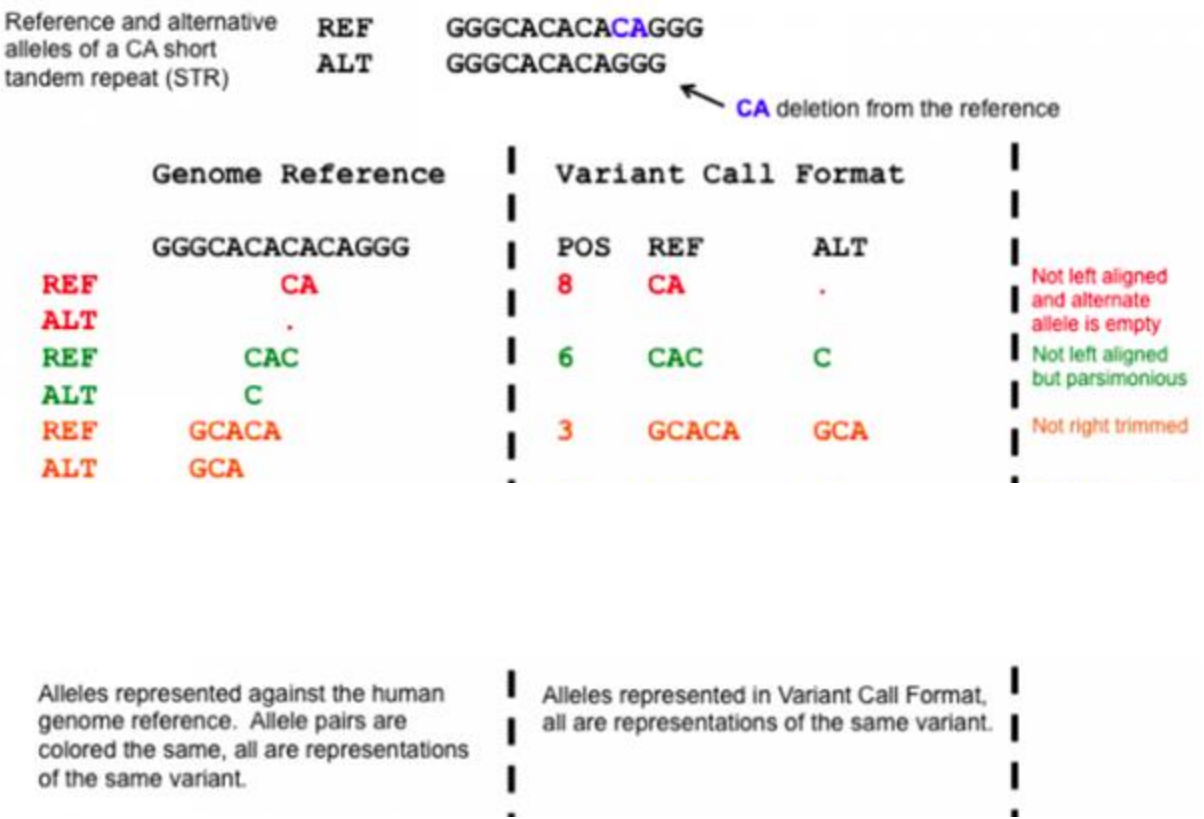
Variant shifting (alignment) and parsimony/trimming



Parsimony: representing variant in as few nucleotides as possible without reducing the length of any allele to 0

Left (right) aligning = shifting the start position of a variant as far to the left (right) as possible

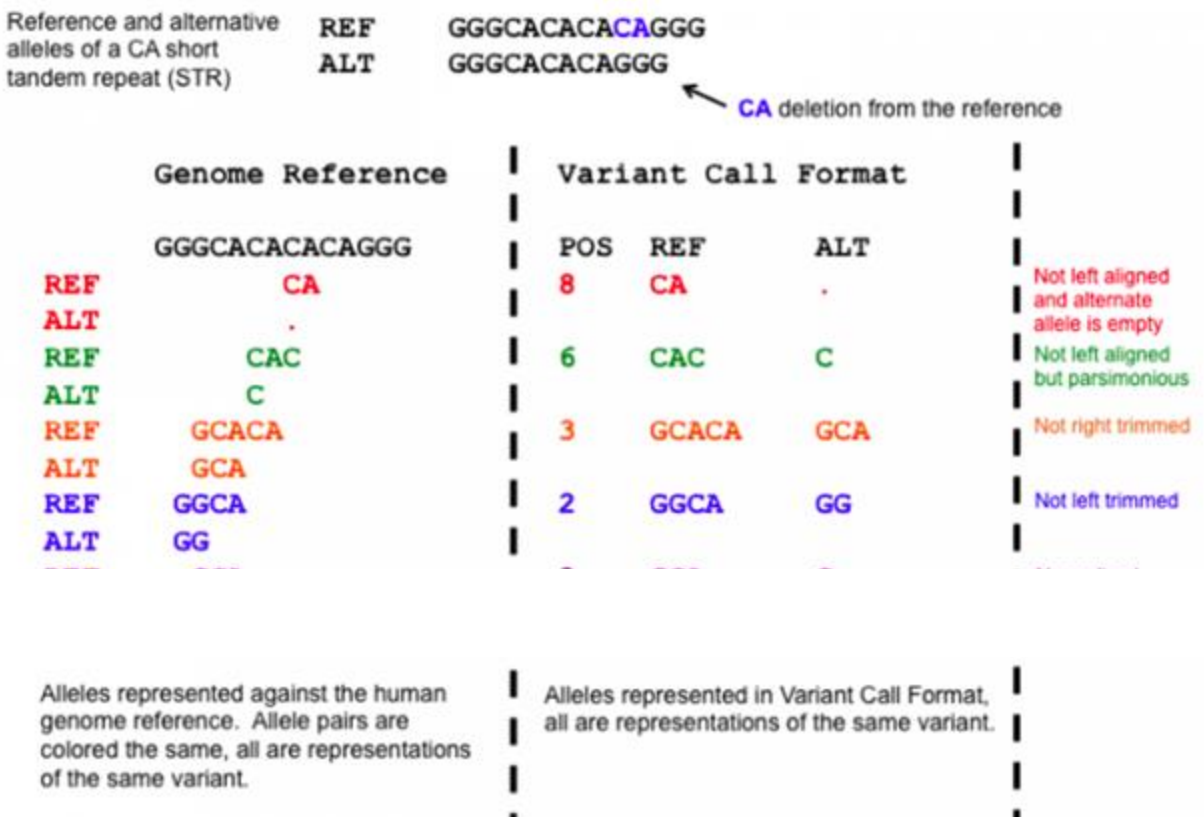
Variant shifting (alignment) and parsimony/trimming



Parsimony: representing variant in as few nucleotides as possible without reducing the length of any allele to 0

Left (right) aligning = shifting the start position of a variant as far to the left (right) as possible

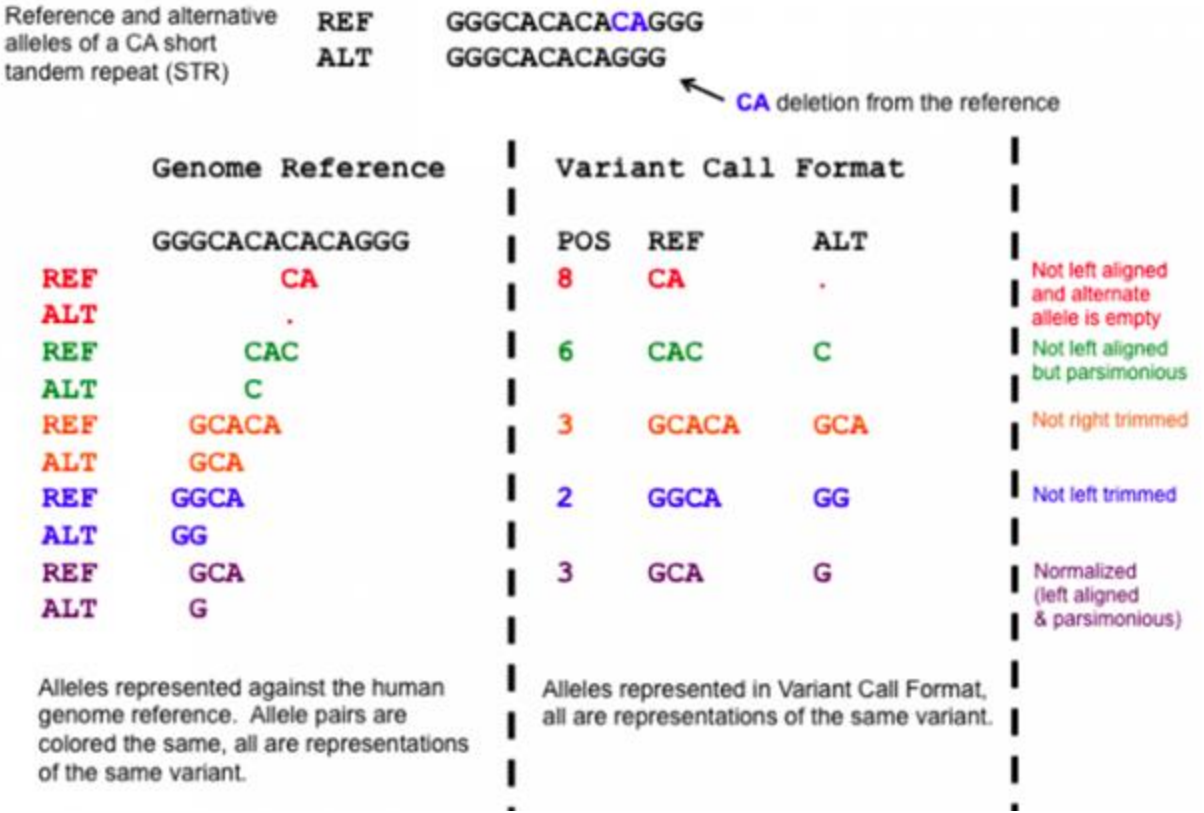
Variant shifting (alignment) and parsimony/trimming



Parsimony: representing variant in as few nucleotides as possible without reducing the length of any allele to 0

Left (right) aligning = shifting the start position of a variant as far to the left (right) as possible

Variant shifting (alignment) and parsimony/trimming



Parsimony: representing variant in as few nucleotides as possible without reducing the length of any allele to 0

Left (right) aligning = shifting the start position of a variant as far to the left (right) as possible

Intervals are often represented in the BED format

- There are several flavors of BED format: BED3, BED4, BED6, BED8, etc
- First 3 fields always required: **chr, start, stop**
- Followed by up to 9 additional optional fields: name, score, strand, thickStart, thickEnd, itemRGB, blockCount, blockSizes, blockStarts

```
chr7    127471196  127472363  Pos1  0  +
chr7    127472363  127473530  Pos2  0  +
chr7    127473530  127474697  Pos3  0  +
chr7    127474697  127475864  Pos4  0  +
chr7    127475864  127477031  Neg1  0  -
chr7    127477031  127478198  Neg2  0  -
chr7    127478198  127479365  Neg3  0  -
chr7    127479365  127480532  Pos5  0  +
chr7    127480532  127481699  Neg4  0  -
```

Manipulation of SAM/BAM and BED files

- Several tools are used ubiquitously in sequence analysis to manipulate these files

- SAM/BAM files
  - samtools
  - bamtools
  - Picard

- BED files
  - bedtools
  - bedops

Bedtools: a swiss army knife for genome analysis

BEDTools: a flexible suite of utilities for comparing genomic features

Aaron R. Quinlan; Ira M. Hall

Bioinformatics (2010) 26 (6): 841-842.
DOI: https://doi.org/10.1093/bioinformatics/btq033
Published: 28 January 2010 **Article history**

Abstract

Motivation: Testing for correlations between different sets of genomic features is a fundamental task in genomics research. However, searching for overlaps between features with existing web-based methods is complicated by the massive datasets that are routinely produced with current sequencing technologies. Fast and flexible tools are therefore required to ask complex questions of these data in an efficient manner.

Results: This article introduces a new software suite for the comparison, manipulation and annotation of genomic features in Browser Extensible Data (BED) and General Feature Format (GFF) format. BEDTools also supports the comparison of sequence alignments in BAM format to both BED and GFF features. The tools are extremely efficient and allow the user to compare large datasets (e.g. next-generation sequencing data) with both public and custom genome annotation tracks. BEDTools can be combined with one another as well as with standard UNIX commands, thus facilitating routine genomics tasks as well as pipelines that can quickly answer intricate questions of large genomic datasets.

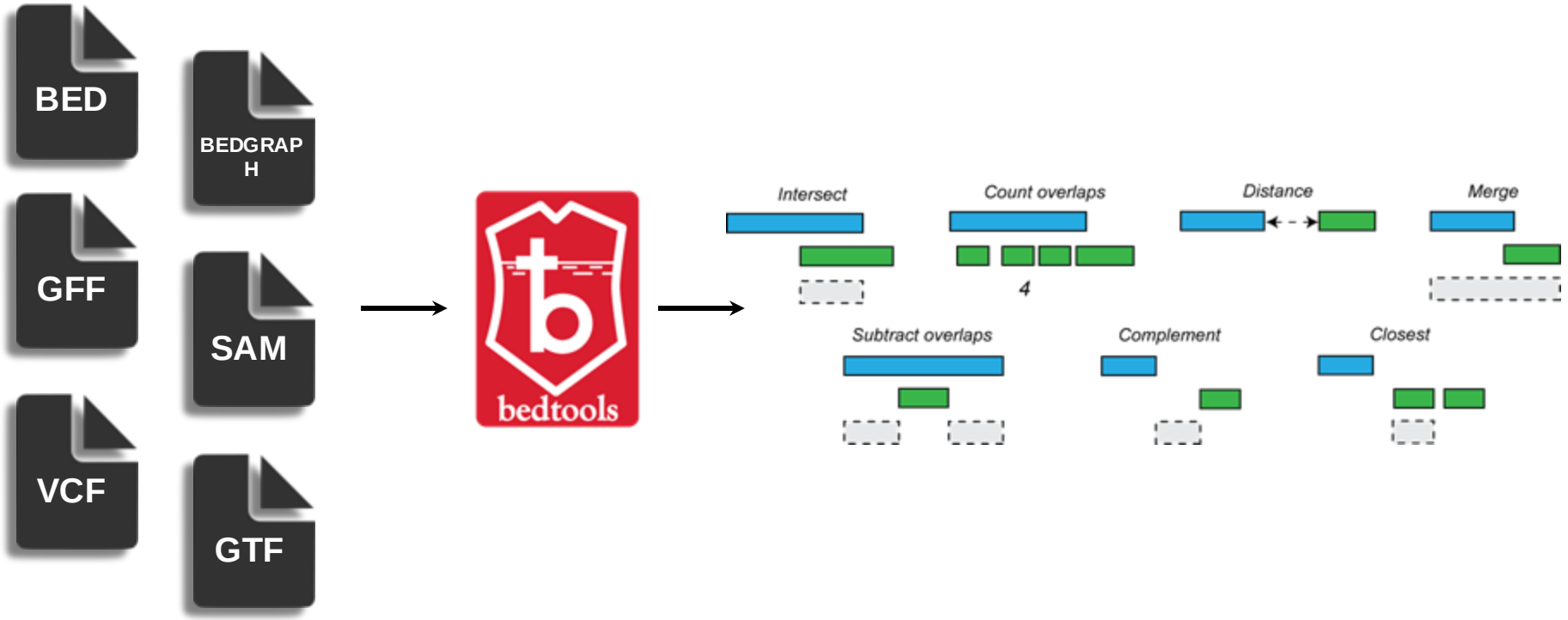
Papers:

https://doi.org/10.1093/bioinformatics/btq033
DOI: 10.1002/0471250953.bi1112s47

Documentation:

http://bedtools.readthedocs.io/en/latest/

Supports most interval formats & handles diff. coordinate systems



http://bedtools.readthedocs.io/en/latest/

Bedtools: example analyses

- Closest gene to a ChIP-seq peak.
- Is my latest discovery novel?
- Is there strand bias in my data?
- How many genes does this mutation affect?
- Where did I fail to collect sequence coverage?
- Is my favorite feature significantly correlated with some other feature?
- What is the density of variants in "windows" along the genome?

http://bedtools.readthedocs.io/en/latest/

Assignment: work through the bedtools tutorial.

https://sandbox.bio/tutorials/bedtools-intro

Non-interactive version: http://quinlanlab.org/tutorials/bedtools.html