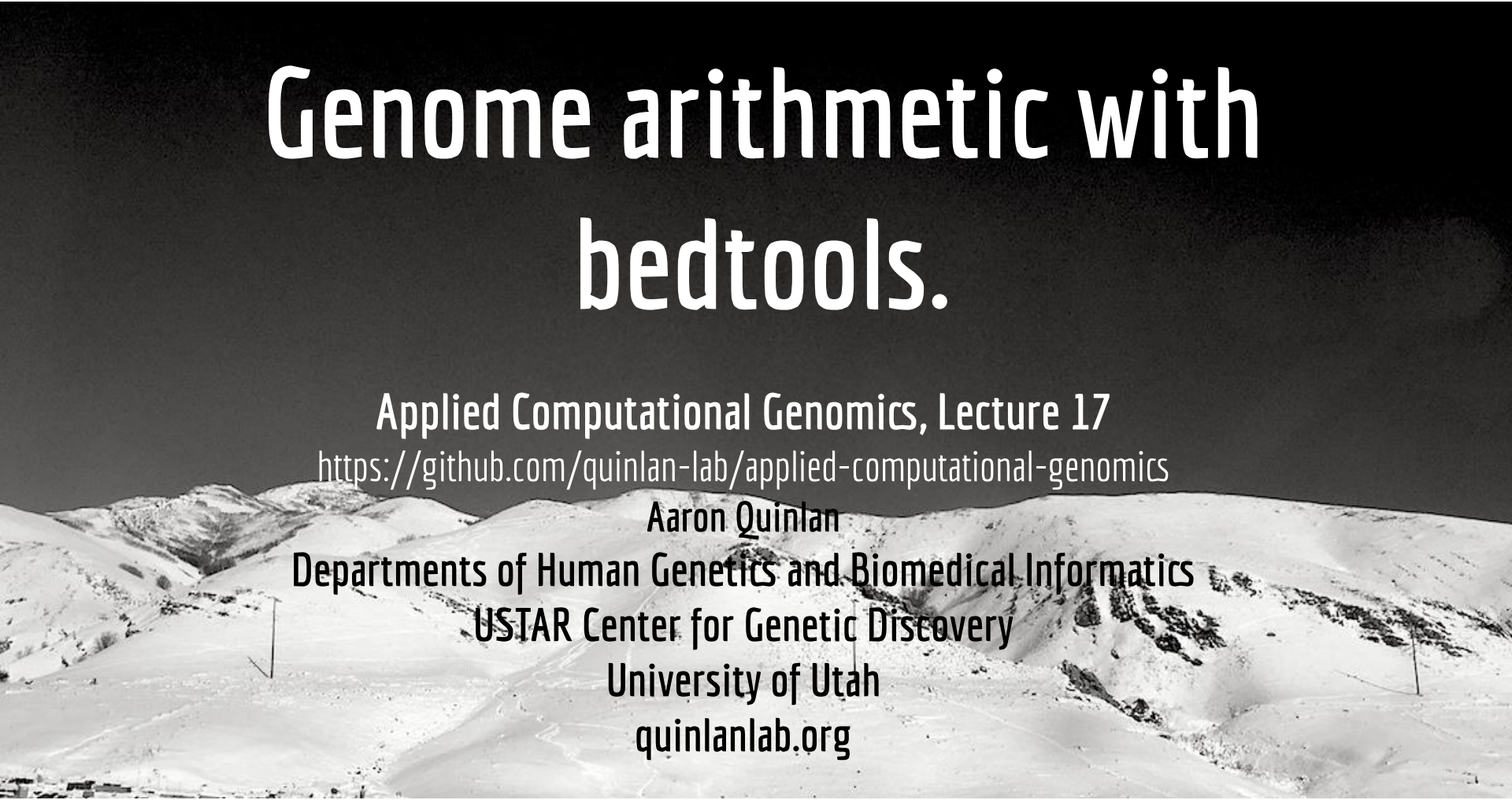
Genome arithmetic with bedtools.

Applied Computational Genomics, Lecture 17

https://github.com/quinlan-lab/applied-computational-genomics

Aaron Quinlan
Departments of Human Genetics and Biomedical Informatics
USTAR Center for Genetic Discovery
University of Utah
quinlanlab.org

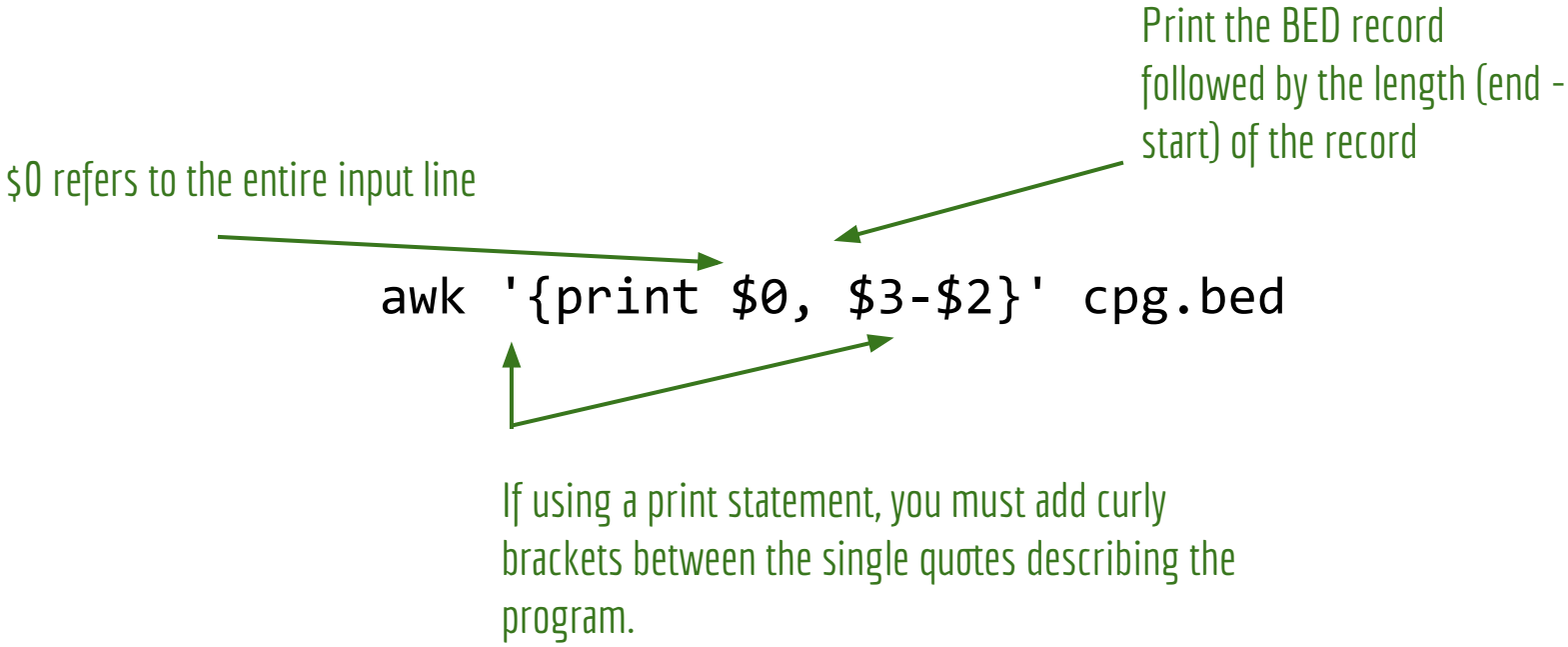


Getting a bit more sophisticated. Mini programming!!!

Report lines if they are the 100th through the 200th lines in the file OR (||) they are from chr22

```
awk '(NR>=100 && NR <= 200) || ($1 == "chr22")' cpg.bed
```

Do some computation and report the results



By default, output is separated by a space. Prefer tabs

BEGIN: before anything else happens, execute what is in the BEGIN statement. Then start processing the input.

Print the BED record followed by the length (end - start) of the record. Separated by a TAB, the OFS (output field separator)

```
awk 'BEGIN{OFS="\t"}{print $0, $3-$2}' cpg.bed
or
awk -v OFS="\t" '{print $0, $3-$2}' cpg.bed
or
awk '{len=($3-$2); print $0"\t"len}' cpg.bed
or
awk -v OFS="\t" '{len=($3-$2); print $0, len}'
cpg.bed
```

Compute the total number of base pairs represented by CpG islands

Create a variable named "sum" whose value starts at 0, but is increased by the length ($3-$2) of each CpG island.

END: after all the processing of each line in the file occurs, print the final value of sum.

```
awk '{sum += $3-$2}END{print sum}' cpg.bed
```

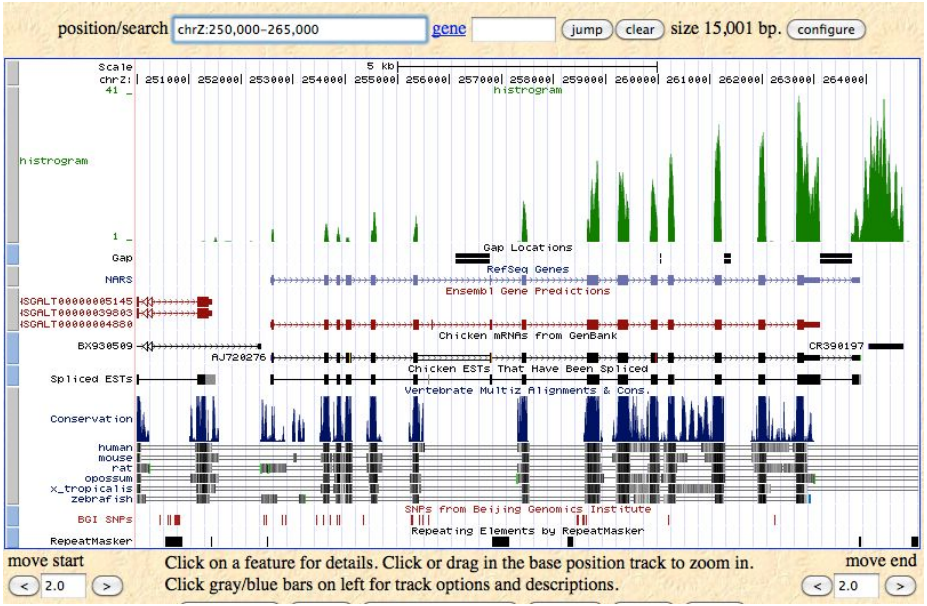
How many (whitespace-separated) columns are on each line?

NF: The number of "fields" (that is, the number of whitespace-separated values) detected for the line

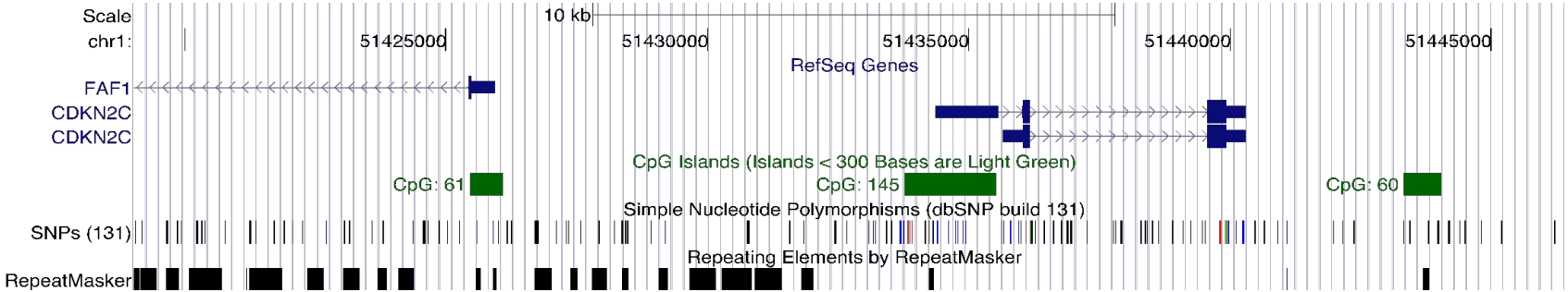
```
awk '{print NF}' cpg.bed
```

What is a genome interval?

- 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**

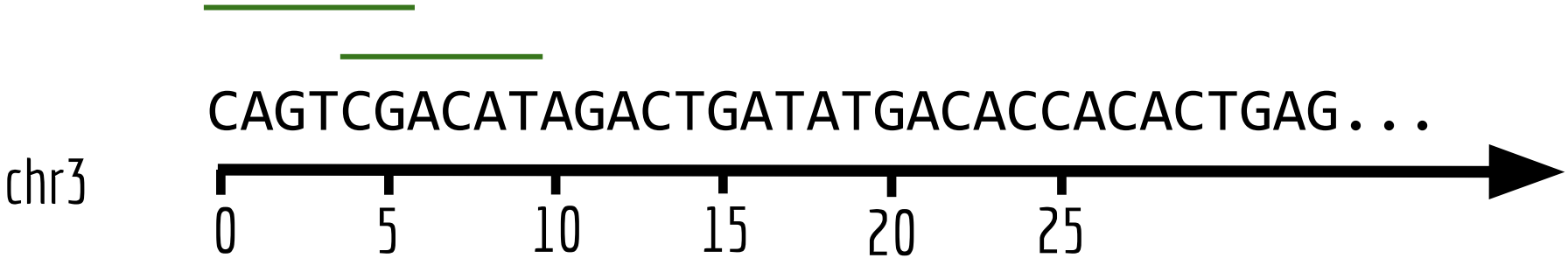


Genome intervals



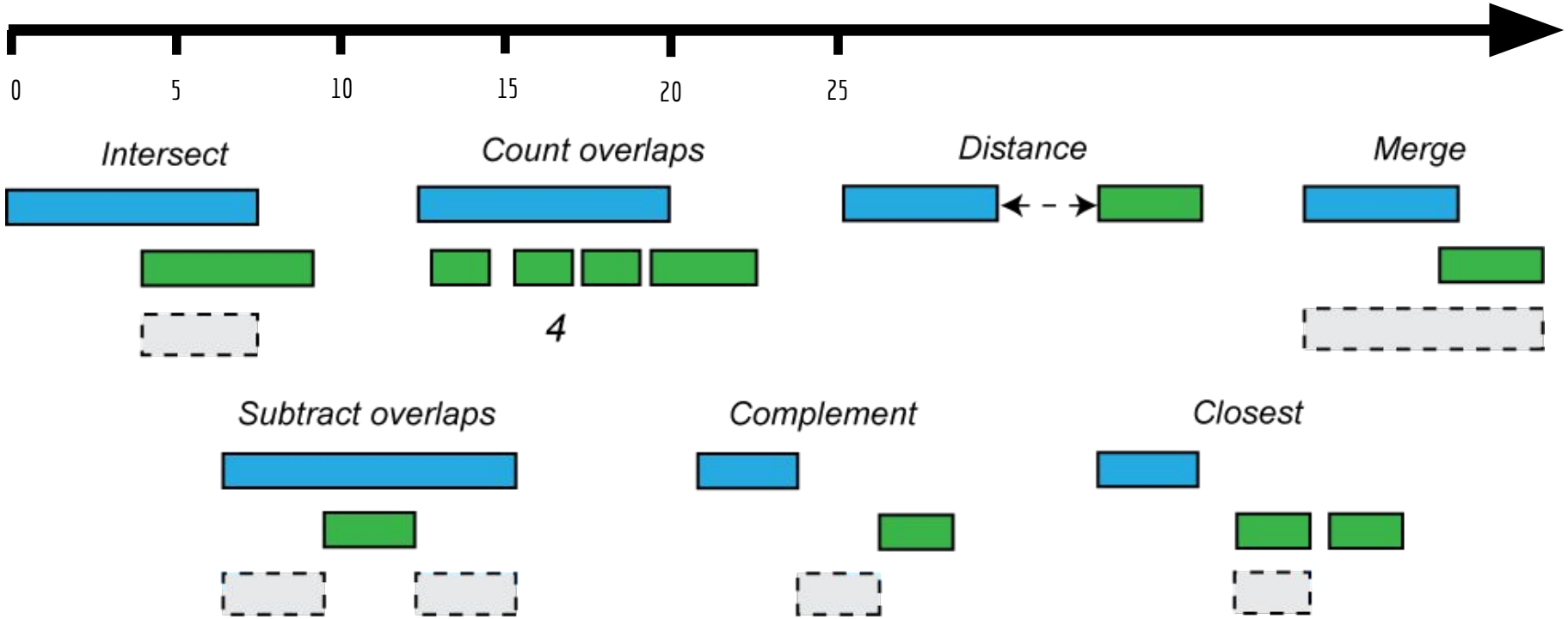
Genome arithmetic: the method of comparing, contrast and gain insight among multiple genome interval files

Genome arithmetic depends upon the genome coordinate system

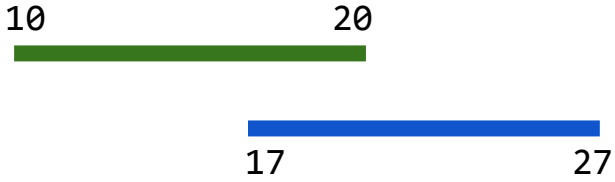


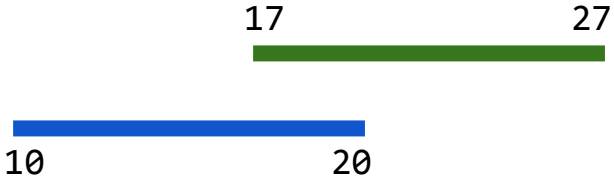

```
chr3    0    7
chr3    4    10
```

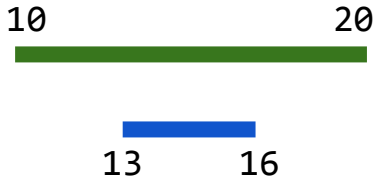
Genome arithmetic operations

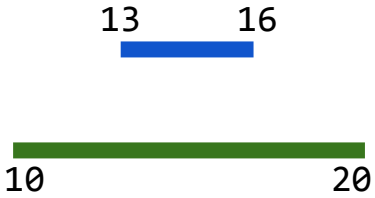


Do two intervals intersect (overlap)?



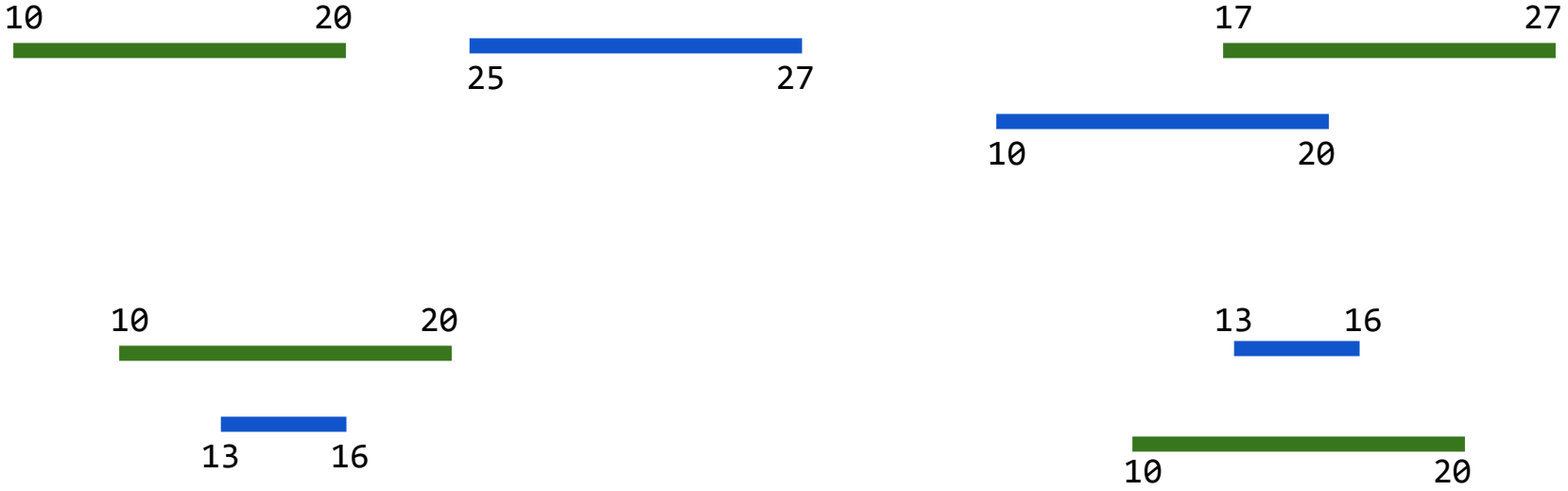







```
if ((a.start <= b.start and a.end >= b.start) or
    (b.start <= a.start and b.end >= a.start) or
    (a.start <= b.start and a.end >= b.end)   or
    (b.start <= a.start and b.end >= a.end))
{
  INTERSECTION!!!
}
else NADA!!!
```

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
```

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

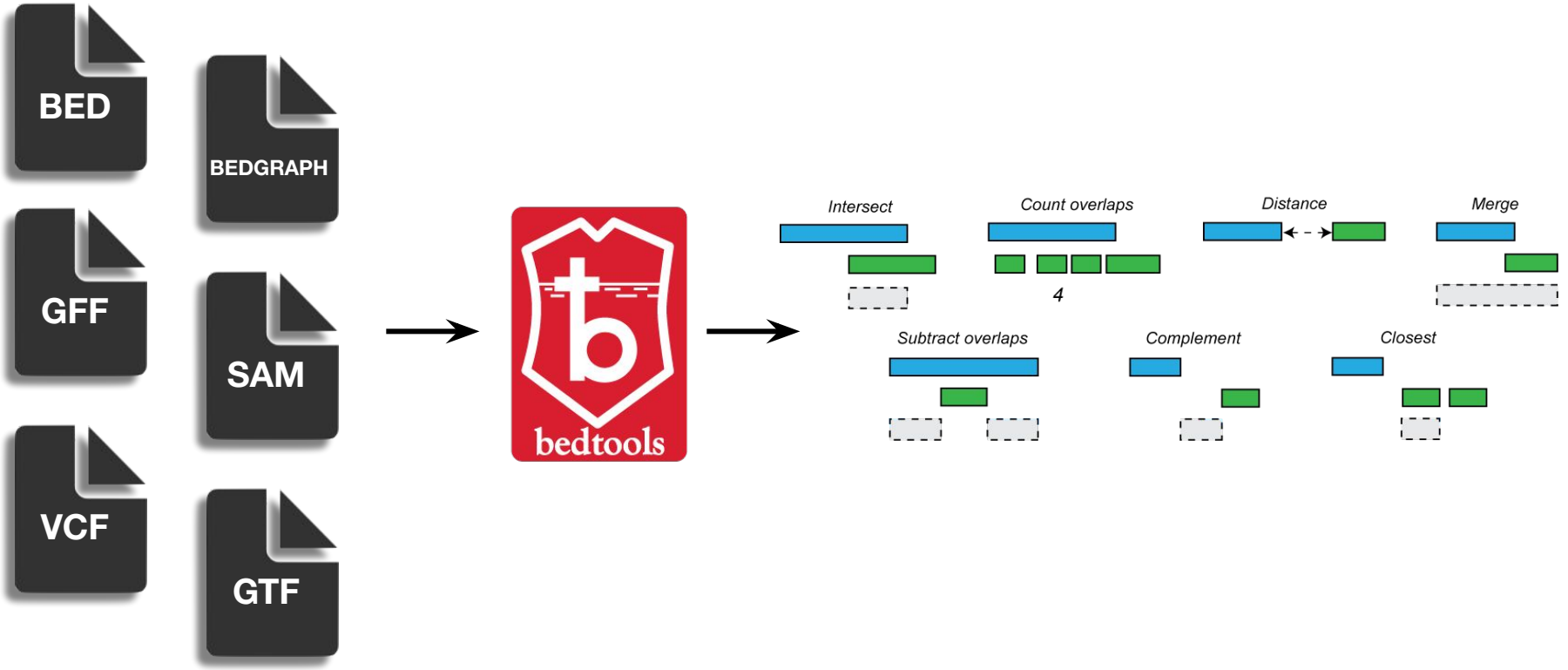
Documentation:

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

Code:

https://github.com/arq5x/bedtools2

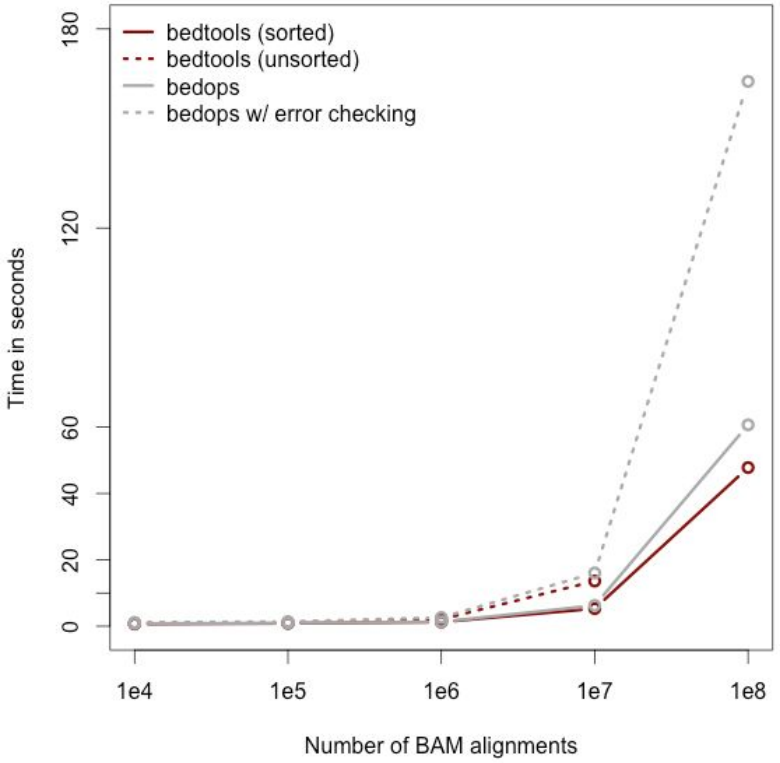
Supports most interval formats & handles diff. coordinate systems



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?

Bedtools is fairly fast.



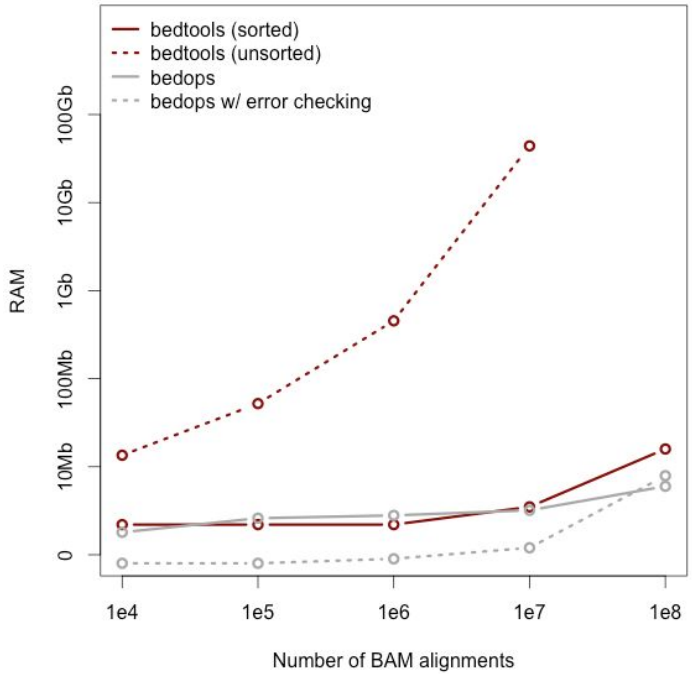

```
# bedtools sorted
$ bedtools intersect \
          -a ccds.exons.bed -b aln.bam.bed \
          -c \
          -sorted

# bedtools unsorted
$ bedtools intersect \
          -a ccds.exons.bed -b aln.bam.bed \
          -c

# bedmap (without error checking)
$ bedmap --echo --count --bp-ovr 1 \
         ccds.exons.bed aln.bam.bed

# bedmap (no error checking)
$ bedmap --ec --echo --count --bp-ovr 1 \
         ccds.exons.bed aln.bam.bed
```

And doesn't use (too) much memory when files are "genome sorted".




```
# bedtools sorted
$ bedtools intersect \
          -a ccds.exons.bed -b aln.bam.bed \
          -c \
          -sorted

# bedtools unsorted
$ bedtools intersect \
          -a ccds.exons.bed -b aln.bam.bed \
          -c

# bedmap (without error checking)
$ bedmap --echo --count --bp-ovr 1 \
         ccds.exons.bed aln.bam.bed

# bedmap (no error checking)
$ bedmap --ec --echo --count --bp-ovr 1 \
         ccds.exons.bed aln.bam.bed
```

Sort chromosomes lexicographically.

Then sort numerically by start coordinate

```
sort -k1,1 -k2,2n myfile.bed > myfile.sorted.bed
```

The "chromsweep" algorithm

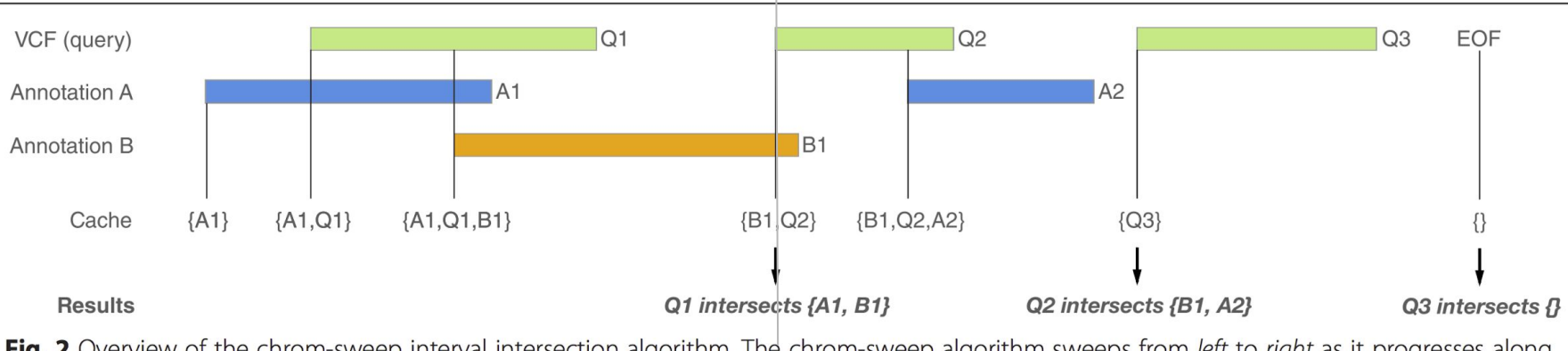
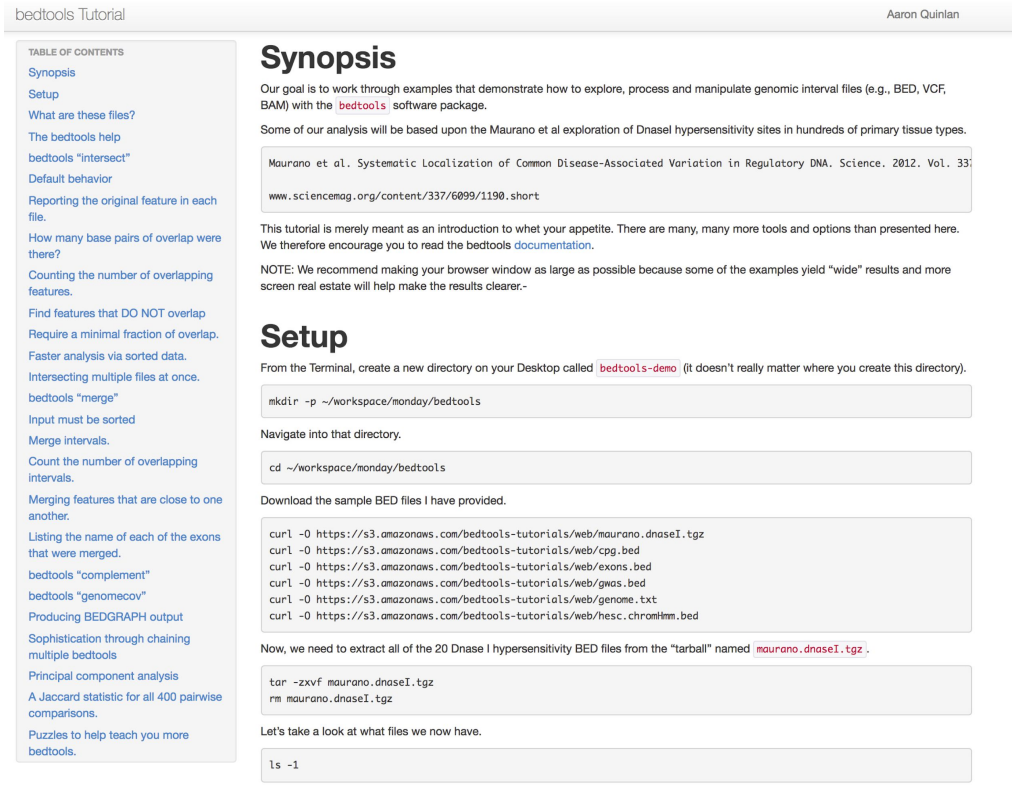


Fig. 2 Overview of the chrom-sweep interval intersection algorithm. The chrom-sweep algorithm sweeps from *left* to *right* as it progresses along each chromosome. *Green* intervals from the query VCF in the *first row* are annotated by annotation files A (*blue*) and B (*orange*) in the *second* and *third rows*, respectively. The cache row indicates which intervals are currently in the cache at each point in the progression of the sweeping algorithm. Intervals enter the cache in order of their chromosomal start position. First *A1* enters the cache followed by *Q1*. Since *Q1* intersects *A1*, they are associated, as are *Q1* and *B1* when *B1* enters the cache. Each time a new query interval enters the cache, any interval it does not intersect is ejected. Therefore, when *Q2* enters the cache, *Q1* and *A1* are ejected. Since *Q1* is a query interval, it is sent to be reported as output. Proceeding to the right, *A2* and then *Q3* enter the cache; the latter is a query interval and so the intervals that do not overlap it—*B1*, *Q2*, and *A2*—are ejected from the cache with the query interval, *Q2*, which is sent to the caller. Finally, as we reach the end of the incoming intervals, we clear out the final *Q3* interval and finalize the output for this chromosome. EOF: End of File

Let's work through the bedtools tutorial.



bedtools Tutorial — Aaron Quinlan

TABLE OF CONTENTS

Synopsis

Our goal is to work through examples that demonstrate how to explore, process and manipulate genomic interval files (e.g., BED, VCF, BAM) with the `bedtools` software package.

Some of our analysis will be based upon the Maurano et al exploration of DnaseI hypersensitivity sites in hundreds of primary tissue types.

```
Maurano et al. Systematic Localization of Common Disease-Associated Variation in Regulatory DNA. Science. 2012. Vol. 33

www.sciencemag.org/content/337/6099/1190.short
```

This tutorial is merely meant as an introduction to whet your appetite. There are many, many more tools and options than presented here. We therefore encourage you to read the bedtools documentation.

NOTE: We recommend making your browser window as large as possible because some of the examples yield "wide" results and more screen real estate will help make the results clearer.-

Setup

From the Terminal, create a new directory on your Desktop called `bedtools-demo` (it doesn't really matter where you create this directory).

```
mkdir -p ~/workspace/monday/bedtools
```

Navigate into that directory.

```
cd ~/workspace/monday/bedtools
```

Download the sample BED files I have provided.

```
curl -O https://s3.amazonaws.com/bedtools-tutorials/web/maurano.dnaseI.tgz
curl -O https://s3.amazonaws.com/bedtools-tutorials/web/cpg.bed
curl -O https://s3.amazonaws.com/bedtools-tutorials/web/exons.bed
curl -O https://s3.amazonaws.com/bedtools-tutorials/web/gwas.bed
curl -O https://s3.amazonaws.com/bedtools-tutorials/web/genome.txt
curl -O https://s3.amazonaws.com/bedtools-tutorials/web/hesc.chromHmm.bed
```

Now, we need to extract all of the 20 Dnase I hypersensitivity BED files from the "tarball" named `maurano.dnaseI.tgz`.

```
tar -zxvf maurano.dnaseI.tgz
rm maurano.dnaseI.tgz
```

Let's take a look at what files we now have.

```
ls -1
```

Connect to malibu.

```
mkdir bedtools-tutorial
cd bedtools-tutorial
```

http://quinlanlab.org/tutorials/bedtools/bedtools.html