Back to BedTools – most important tool: intersect

- Some functionality may be easier to recreate: *slop, flank, complement* (although don’t underestimate them)

- *Intersect* is the a complex algorithm that would be difficult to improve on

(If would take substantial work to write a better intersect than what *bedtools* has, many tools are variants of intersect: *window, closest* etc)
Using the genes feature file

```bash
$ (head -i sc.gff & cat oc.gff) | awk 'BEGIN { split($0, x, "\t"); $0="gene"} { print x[1];}' > genes.gff

$ head genes.gff
#gff-version 3
chr1 SGD gene 335 649 . + ID=YAL056W
chr1 SGD gene 538 792 . + ID=YAL069W-A
chr1 SGD gene 1307 2149 . - ID=YAL069C
chr1 SGD gene 2480 2707 . + ID=YAL067W-A
chr1 SGD gene 7235 9016 . - ID=YAL067C
chr1 SGD gene 10091 10399 . + ID=YAL066W
chr1 SGD gene 11565 11851 . + ID=YAL056C
chr1 SGD gene 12046 12426 . + ID=YAL064B
chr1 SGD gene 13363 13743 . - ID=YAL064C-A
```

See the code repository for step by step instructions

## Running the intersection

```
$ /bin/bedtools intersect -a bw.bed -b genes.gff | head -3
```

```
chr1 273 343
chr1.274.796.2105.2106.4097/2 60 + chr1. SGD q
chr1 298 368
chr1.299.779.1111.1140.6421/1 60 + chr1. SGD q
chr1 335 649
chr1.337.649.1100.1120.24 60 + chr1. SGD q
chr1 310 380
chr1.311.918.4110.4147/4 60 + chr1. SGD q
chr1 335 649
```

-wo flag writes entry A then entry B followed by the overlap

## Intersect alignments

```
$ /bin/bedtools intersect -asam r1.fq.bam -b genes.gff > genomic.bam
```

```
$ /bin/bamtools index genomic.bam
```

```
```

Investigate what the -v flag does

## Turn the alignment into BED file

```
$ /bin/bamtools bamtobed -h
```

```
Tool: bamtools bamtobed [xs] bamtoBed
Version: v2.17.0
Summary: Converts BAM alignments to BED or BEDPE format.
Usage: bamtools bamtobed [OPTIONS] -i <bam>
```
Operations may turn BAM files into BED

The `--wo` option turns the BAM file into BED as it now needs to be displayed as text.

Warning: the file has mixed representation left side BED right side GFF

Awesome features – strand, overlap fraction and others

- For each entry in A, report the number of overlaps with B.
  - Reports 0 for A entries that have no overlap with B.
  - Overlaps restricted by `-f` and `-r`.

- Only report those entries in A that have no overlap with B.
  - Similar to `--no-overlap` (as homespans).

- Minimum overlap required as a fraction of A.
  - Default is 10% (i.e., 1/10).
  - `FRACTION` (e.g., 0.50)

- Require that the fraction overlap be reciprocal for A and B.
  - In other words, if `-f` is 0.99 and `-r` is used, this requires
    that B overlap 99% of A and A overlap 99% of B.

- Require same strandedness. That is, only report hits in B
  that overlap A on the same strand.
  - By default, overlaps are reported without respect to strand.

- Require different strandedness. That is, only report hits in B
  that overlap A on the opposite strand.
  - By default, overlaps are reported without respect to strand.

Study and learn the intersect flags

- These have been added because they support very common use cases

- Create toy examples and investigate the results

- Bedtools **intersect** can solve the vast majority of questions that deal with positioning

Intersect variants: window, closest

- **bedtools window**
  Examine a "window" around each feature in A and reports all features in B that overlap the window.

- **bedtools closest**
  For each feature in A, finds the closest feature (upstream or downstream) in B.
  
  **Window**: extend or flank the original intervals then uses intersect.
  
  **Closest**: intersect intervals then sorts by the overlap.
Other helpful bedtool commands:

**getfasta** — extracts sequences

```
$ ./bedtools getfasta
```

- **Tool:** bedtools getfasta (aka fastaFromBed)
- **Version:** V2.26.2
- **Summary:** Extract DNA sequences into a fasta file based on feature coordinates.
- **Usage:** bedtools getfasta [OPTIONS] -fi <fasta> -bed <bed/gff/vcf> --fo <fasta>

**Options:**
- **-fi** Input FASTA file
- **-bed** BED/GFF/VCF file of ranges to extract from -fi
- **-fo** Output file (can be FASTA or TAB-delimited)
- **-name** Use the name yield for the FASTA header
- **-tab** Write output in TAB delimited format.
  - Default is FASTA format.
- **-s** Force strandness. If the feature occupies the antisense strand, the sequence will be reverse complemented.
  - By default, strand information is ignored.

```
$ ./bedtools getfasta -fi human_gencode_v19.sorted.gff3 -bed human_promoters.bed -fo human_promoters.fna
```

**get the sequence to every feature annotated as binding_site**

```
$ cat simple.gff | awk '{OFS="\t";print $0}' > binding.gff
```

```
$ head -3 binding.gff
chr1   SGD   binding_site   1066   2306   +
chr1   SGD   binding_site   2306   5286   +
chr1   SGD   binding_site   5286   1238   +
```

```
$ bedtools getfasta -fi refs/na12878/gencode_v28.annotation.gff3 -bed binding.gff -fo binding.fa
```

```
$ head -4 binding.fa
CTATATTGAT
GATATTGAT
ATATGATAT
TATATGATAT
```

**bedtools nuc**

```
$ ./bedtools nuc
```

- **Tool:** bedtools nuc (aka nucbed)
- **Version:** V2.26.2
- **Summary:** Profile the nucleotide content of intervals in a fasta file.
- **Usage:** bedtools nuc [OPTIONS] -fi <fasta> -bed <bed/gff/vcf>

**Options:**
- **-fi** Input FASTA file
- **-bed** BED/GFF/VCF file of ranges to extract from -fi
- **-n** Profile the sequence according to strand.
- **-seq** Print the extracted sequence
- **-pattern** Report the number of times a user-defined sequence is observed (case-sensitive).
- **-c** Ignore cases when matching patterns. By default, case matters.

```
$ ./bedtools nuc -fi human_gencode_v19.sorted.gff3 -bed human_promoters.bed
```

**bedtools groupby**

```
$ ./bedtools groupby
```

- **Tool:** bedtools groupby
- **Version:** V2.26.2
- **Summary:** Summarize a dataset column based upon common genomic groups. Also to the -g option list method.
- **Usage:** bedtools groupby -g [group_columns] -o [group_mode]...

**Options:**
- **-g** Specify the column (1-based) for the grouping.
- **-o** Specify the output mode (1-based) that should be summarized.
- **-n** Not required.
- **-c** Specify the operation that should be applied to the columns.

```
$ cat groupby.gff | awk '{OFS="\t";print $0}' > groupby.gff
```
More on groupby

- Data must be sorted on the grouping column

- Valid operators: sum, mean, median, mode, min, max (see help for more)

- Example: Find the average length of each feature in the `features.gff` file

Using bedtools groupby

```bash
# Examples of using bedtools groupby
```