Week 12, Lecture 23

István Albert

Biochemistry and Molecular Biology
and Bioinformatics Consulting Center

Penn State

A suite of tools to address common questions raised in genomic studies — mostly with regard to overlap and proximity relationships between data sets — BEDOPS aims to be scalable and flexible, facilitating the efficient and accurate analysis and management of large-scale genomic data.

downloads

Current download: v1.2.5 / October 13, 2012 (changes)

A companion set of documentation, conversion scripts and sample input files is also available for offline browsing.

It offers a complementary tools and approaches – very nice examples and tutorials that demonstrate the thought process that goes with interval analysis.
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

(It would take substantial work to write a better intersect than what bedtools has, many tools are variants of intersect: window, closest etc)
BedTools intersect

$ ~/bin/bedtools intersect

Tool: bedtools intersect (aka intersectBed)
Version: v2.17.0
Summary: Report overlaps between two feature files.

Usage: bedtools intersect [OPTIONS] -a <bed/gff/vcf> -b <bed/gff/vcf>

Options:
- -abam The A input file is in BAM format. Output will be BAM as well.
- -ubam Write uncompressed BAM output. Default writes compressed BAM.
- -bed When using BAM input (-abam), write output as BED. The default is to write output in BAM when using -abam.
- -wa Write the original entry in A for each overlap.
- -wb Write the original entry in B for each overlap.
  - Useful for knowing _what_ A overlaps. Restricted by -f and -r.
Using the genes feature file

```bash
$ (head -1 sc.gff & cat sc.gff | awk ' $3="gene" { split($9,x,";"); $9=x[1]; print $0 }') > genes.gff

$ head genes.gff
##gff-version 3
chrI SGD gene 335 649 . + . ID=YAL069W
chrI SGD gene 538 792 . + . ID=YAL068W-A
chrI SGD gene 1807 2169 . - . ID=YAL068C
chrI SGD gene 2480 2707 . + . ID=YAL067W-A
chrI SGD gene 7235 9016 . - . ID=YAL067C
chrI SGD gene 10091 10399 . + . ID=YAL066W
chrI SGD gene 11565 11951 . - . ID=YAL065C
chrI SGD gene 12046 12426 . + . ID=YAL064W-B
chrI SGD gene 13363 13743 . - . ID=YAL064C-A

$ See the code repository for step by step instructions
Running the intersection

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

Investigate what the –v flag does

```
ialbert@porthos ~/work/lec23
$ ~/bin/bedtools intersect -abam r1.fq.bam -b genes.gff > genomic.bam
ialbert@porthos ~/work/lec23
$ ~/bin/samtools index genomic.bam
ialbert@porthos ~/work/lec23
$ 
```
Turn the alignment into BED file

```
ialbert@portos ~/work/lec23
$ ~/bin/bedtools bamtobed -h
Tool:     bedtools bamtobed (aka bamToBed)
Version:  v2.17.0
Summary:  Converts BAM alignments to BED6 or BEDPE format.
Usage:    bedtools bamtobed [OPTIONS] -i <bam>
Options:
         -bedpe
```

```
ialbert@portos ~/work/lec23
$ ~/bin/bedtools bamtobed -i r1.fq.bam | head
chrI   61    131    chrI_62_609_1:0:0_3:0:0_270/2  60   +
chrI   359   429    chrI_360_831_2:0:0_2:0:0_5fe/2  60   +
chrI   363   433    chrI_364_929_1:0:0_1:0:0_332/2  60   +
chrI   446   516    chrI_447_929_3:0:0_5:0:0_c9/1   29   +
chrI   478   548    chrI_479_938_0:0:0_1:0:0_80/1   60   +
chrI   528   598    chrI_529_1015_4:0:0_1:0:0_5cd/1  60   +
chrI   539   609    chrI_62_609_1:0:0_3:0:0_270/1  60   -
chrI   541   611    chrI_542_1102_0:0:0_0:0_0:2e/1  60   +
chrI   635   705    chrI_636_1155_4:0:0:0:0:0:83/2  29   +
chrI   761   831    chrI_762_1298_0:0:0:4:0:0:43b/2  29   +
```
Operations may turn BAM files into BED

<table>
<thead>
<tr>
<th>chrI</th>
<th>359</th>
<th>429</th>
<th>chrI_360_831_2:0:0_2:0:0_5fe/2</th>
<th>60</th>
<th>+</th>
<th>359</th>
<th>429</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>,0,0</td>
<td>1</td>
<td>70,</td>
<td>0, chrI SGD gene</td>
<td>335</td>
<td></td>
<td>649</td>
<td>.</td>
<td>+</td>
</tr>
<tr>
<td>D=YAL069W</td>
<td>70</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>chrI</td>
<td>363</td>
<td>433</td>
<td>chrI_364_929_1:0:0_1:0:0_332/2</td>
<td>60</td>
<td>+</td>
<td>363</td>
<td>433</td>
<td>0</td>
</tr>
<tr>
<td>,0,0</td>
<td>1</td>
<td>70,</td>
<td>0, chrI SGD gene</td>
<td>335</td>
<td></td>
<td>649</td>
<td>.</td>
<td>+</td>
</tr>
<tr>
<td>D=YAL069W</td>
<td>70</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>chrI</td>
<td>446</td>
<td>516</td>
<td>chrI_447_929_3:0:0_5:0:0_c9/1</td>
<td>29</td>
<td>+</td>
<td>446</td>
<td>516</td>
<td>0</td>
</tr>
<tr>
<td>,0,0</td>
<td>1</td>
<td>70,</td>
<td>0, chrI SGD gene</td>
<td>335</td>
<td></td>
<td>649</td>
<td>.</td>
<td>+</td>
</tr>
<tr>
<td>D=YAL069W</td>
<td>70</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

- **c**
  - 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.

- **v**
  - Only report those entries in A that have _no overlaps_ with B.
  - Similar to "grep -v" (an homage).

- **f**
  - Minimum overlap required as a fraction of A.
  - Default is 1E-9 (i.e., 1bp).
  - FLOAT (e.g. 0.50)

- **r**
  - Require that the fraction overlap be reciprocal for A and B.
  - In other words, if -f is 0.90 and -r is used, this requires that B overlap 90% of A and A _also_ overlaps 90% of B.

- **s**
  - 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.

- **S**
  - 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**

  Examines 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

```
ialbert@porthos ~/work/lec20
$ ~/bin/bedtools getfasta

Tool: bedtools getfasta (aka fastaFromBed)
Version: v2.16.2
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 field for the FASTA header
- **-tab** Write output in TAB delimited format.
  - Default is FASTA format.
- **-s** Force strandedness. If the feature occupies the antisense, strand, the sequence will be reverse complemented.
  - By default, strand information is ignored.
```
Get the sequence to every feature annotated as `binding_site`

```
ialbert@porthos ~/work/lec20
$ cat simple.gff | awk ' $3=="binding_site" { print $0 } ' > binding.gff

ialbert@porthos ~/work/lec20
$ head -3 binding.gff
chrI SGD binding_site 532 544 . - . .
chrI SGD binding_site 229670 229682 . + . .
chrII SGD binding_site 6339 6351 . - . .

ialbert@porthos ~/work/lec20
$ ~/bin/bedtools getfasta -fi refs/sc.fa -bed binding.gff -fo binding.fa

ialbert@porthos ~/work/lec20
$ head -4 binding.fa
>chrI:531-544
ATCATTATGCACG
>chrI:229669-229682
CGTGTATGGTGAT
```

Note the coordinate system change!
Tool: bedtools nuc (aka nucBed)
Version: v2.16.2
Summary: Profiles 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
- **-s**  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 case when matching -pattern. By default, case matters.

Output format:
The following information will be reported after each BED entry:
1) %AT content
bedtools groupby

Very potent tools once we understand what it does. Groups on a column and applies an operator to a different column.
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
ialbert@porhos ~/work/lec23
$ cat features.gff | awk '{ $6=$5-$4+1; print $0 } ' > temp.gff

ialbert@porhos ~/work/lec23
$ sort -k 3 temp.gff > sizes.gff

ialbert@porhos ~/work/lec23
$ ~/bin/bedtools groupby -i sizes.gff -g 3 -c 6 -ops mean | head
ARS    372.50445103857566644
CDS    1284.17101531480420817
LTR_retrotransposon 5837.02000000000043656
X_element   448.65625
X_element_combinatorial_repeat 275.821428571428555188
Y_prime_element 6057.89473684210497595
binding_site 13
centromere 117.6875
centromere_DNA_Element_I 10
centromere_DNA_Element_II 82.6875

ialbert@porhos ~/work/lec23
$  
```
Generate a GFF file from the mutations (homework 20)

```
ialbert@porthos ~/work/lec23
$ cat mutations.txt | awk '{ print $1, "wgsim", "variant", $2, $2, ".", $5, ".", "ref="$3 ";alt="$4 " }' > mutations.gff

ialbert@porthos ~/work/lec23
$ ~/bin/bedtools intersect -a mutations.gff -b snps.vcf | head -5
chrI  wgsim  variant  4748    4748    .    +    .    ref=T;alt=K
chrI  wgsim  variant  28903  28903  .    +    .    ref=-;alt=C
chrI  wgsim  variant  33773  33773  .    +    .    ref=G;alt=S
chrI  wgsim  variant  39716  39716  .    +    .    ref=G;alt=K
chrI  wgsim  variant  47844  47844  .    -    .    ref=-;alt=C
```

Find mutations that you can recover via snp calling
Homework 23

Refine homework 20. Use the VCF file and mutations file from that homework. Transform the mutations.txt file into a GFF or VCF file.

Using bedtools commands show the code and report the numbers for

- **TP (True Positives):** number of expected SNPs that you have found
- **FP (False Positives)** – SNPs that you found but are not in the mutation file
- **FN (False Negatives)** – SNPs that you should have found but did not

Look at the FP and FN list, in one paragraph discuss of some common aspects of the data that ended up on the FP and FN list.