

BEDTools– Genome Arithmetic

Biol4230

Thurs, March 22, 2018

Bill Pearson wrp@virginia.edu 4-2818 Pinn 6-057

- Borrowed from Aaron Quinlan, BEDTools author
- Genome file types/formats
 - format types
 - BED files
- Overview of genome features
- Genome arithmetic
 - approaches
 - UCSC "binning" algorithm
- Introduction to BEDtools
 - intersection / window / many more

fasta.bioch.virginia.edu/biol4230

1

To learn more:

File formats:

1. <http://genome.ucsc.edu/FAQ/FAQformat>
2. Mills (2014) *Curr. Protoc. Bioinform.* 45:A.1B.1-A.1B.18

BEDTools:

1. <http://bedtools.readthedocs.org>
2. Quinlan, A. R. BEDTools: The Swiss-Army Tool for Genome Feature Analysis. *Curr Protoc Bioinformatics* **47**, 11.12.1–11.12.34 (2014).
3. Galaxy Screencasts: <http://main.g2.bx.psu.edu/>
4. Download genome features from UCSC and use BEDTools on the command line

fasta.bioch.virginia.edu/biol4230

2

Analysis of transcription regulation

1. Affy-chip/RNA-seq for RNA expression levels
2. Identify differentially expressed genes (edgeR, DESeq2)
 - (possibly) identify subsets of genes associated with different pathways
3. Isolate regions of DNA around (pathway-specific) coordinately expressed (induced) genes
 - BEDtools
4. Take collections of candidate regulatory regions and look for common DNA motif
 - meme, HOMER

fasta.bioch.virginia.edu/biol4230

3

Genome file formats:

Data	Alignment	Feature
GenBank	SAM/BAM	GFF2/GTF
EMBL	Axt	GFF3
FASTA	MAF	VCF
FASTQ	Chain	BED/bigBED
	Stockholm	bedGraph
		WIG/bigWIG

Mills (2014) Current Prot. in Bioinfo.
Appendix A.1B.1-A.1B.18

fasta.bioch.virginia.edu/biol4230

4

Genome data file formats:

- FASTA:

```
>sp|P09488|GSTM1_HUMAN
MPMILGYWDIRGLAHAIRLLLEYTDSSYEKKYTMGDAPDYDRSQWLNEKFKLGLDFPNLPYLI
DGAHKITQSNAILCYIARKHNLCGETEEKIRVDILENQTMMDNHMQLGMICYNPEF
```

- FASTQ (FASTA with quality scores)

```
@UNC10-SN254_238:4:1101:1351:51174/1
CTTCGCGGTAGCTGGGACCGCCGTTCACTCGCNAATANGCNGCTCTNTGT
+
B@CFFFFFDFHHHJJJJJJIIJJ@FHHJGIJJ#####
@UNC10-SN254_238:4:1101:1351:7840/1
ATTTTTCACATGGAGCAGGAACGGAGTAAATGCAANACNGTGTNTAA
+
CCCCFFFFHHHHJJIIJJIIIGIJJHIGHHIIHIG#####
```

Mills (2014) Current Prot. in Bioinfo.
Appendix A.1B.1-A.1B.18

fasta.bioch.virginia.edu/biol4230

5

Genome feature file formats:

- GFF/GTF/GFF2:

```
##gff-version 2
##Type Protein

##sequence-region P09488 1 218
P09488 UniProtKB mature_protein_region 2 218 . . . Note "Glutathione S-trans. Mu 1"
P09488 UniProtKB polypeptide_domain 2 88 . . . Note "GST N-terminal"
P09488 UniProtKB polypeptide_domain 90 208 . . . Note "GST C-terminal"
P09488 UniProtKB polypeptide_region 7 8 . . . Note "Glutathione binding"
P09488 UniProtKB polypeptide_region 46 50 . . . Note "Glutathione binding"
P09488 UniProtKB polypeptide_region 59 60 . . . Note "Glutathione binding"
P09488 UniProtKB polypeptide_region 72 73 . . . Note "Glutathione binding"
P09488 UniProtKB binding_motif 116 116 . . . Note "Substrate"
P09488 UniProtKB alt_seq_site 153 189 . . . Note "UniProtKB FT ID: VSP_036618"
P09488 UniProtKB nat_variant_site 173 173 . . . Note "K -> N"
P09488 UniProtKB nat_variant_site 210 210 . . . Note "S -> T" ; ...

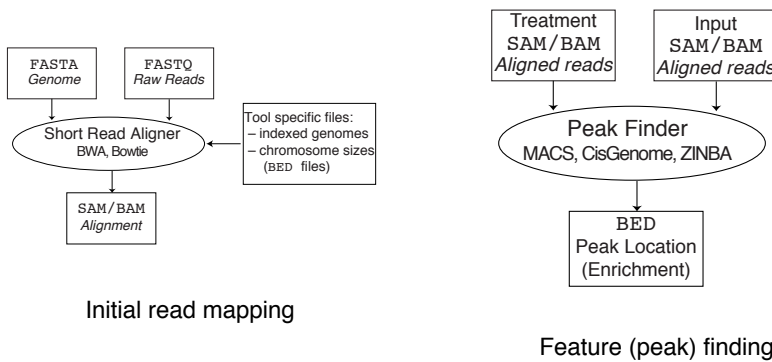
Note field[8] is can be very long, separated by ';'
Note "S -> T" ; Note "UniProtKB FT ID: VAR_014497" ; Note "dbSNP:rs449856" ; Link
"http://www.ncbi.nlm.nih.gov/SNP/snp_ref.cgi?type=rs&rs=449856" ; Link
"http://www.ensembl.org/Homo_sapiens/Variation/Explore?v=rs449856"
```

Mills (2014) Current Prot. in Bioinfo.
Appendix A.1B.1-A.1B.18

fasta.bioch.virginia.edu/biol4230

6

Genome analysis pipelines



Mills (2014) Current Prot. in Bioinfo.
Appendix A.1B.1-A.1B.18

fasta.bioch.virginia.edu/biol4230

9

Genome data (UCSC table browser)

Table Browser

Use this program to retrieve the data associated with a track in text format, to calculate intersections between tracks, and to retrieve DNA sequence covered by a track. For help in using this application see [Using the Table Browser](#) for a description of the controls in this form, the [User's Guide](#) for general information and sample queries, and the [OpenHelix Table Browser tutorial](#) for a narrated presentation of the software features and usage. For more complex queries, you may want to use [Galaxy](#) or our [public MySQL server](#). To examine the biological function of your set through annotation enrichments, send the data to [GREAT](#). Send data to [GenomeSpace](#) for use with diverse computational tools. Refer to the [Credits](#) page for the list of contributors and usage restrictions associated with these data. All tables can be downloaded in their entirety from the [Sequence and Annotation Downloads](#) page.

clade: genome: assembly:

group: track: [add custom tracks](#) [track hubs](#)

table: [describe table schema](#)

region: ☐ genome ☒ position [lookup](#) [define regions](#)

identifiers (names/accessions): [paste list](#) [upload list](#)

filter: [create](#)

intersection: [create](#)

correlation: [create](#)

output format: Send output to ☐ Galaxy ☐ GREAT ☐ GenomeSpace

output file: (leave blank to keep output in browser)

file type returned: ☒ plain text ☐ gzip compressed

[get output](#) [summary/statistics](#)

To reset all user cart settings (including custom tracks), [click here](#).

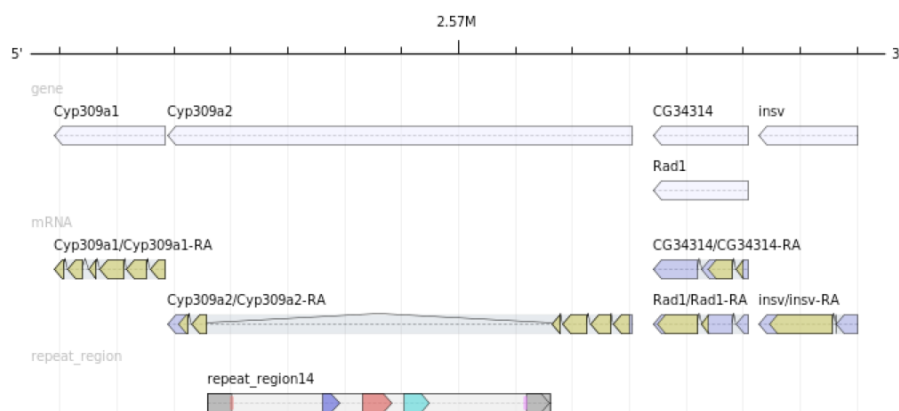
fasta.bioch.virginia.edu/biol4230

10

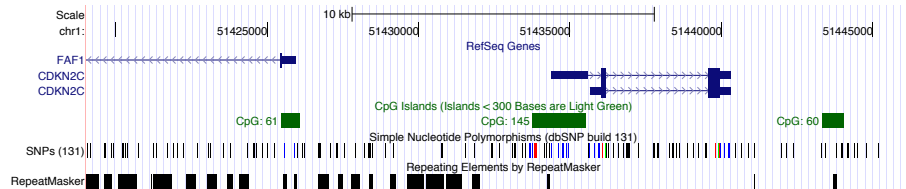
What is a genome “feature”?

- Genes: exons, introns, UTRs, promoters
- Conservation
- Genetic variation
- Transposons
- Origins of replication
- TF binding sites
- CpG islands
- Segmental duplications
- Sequence alignments
- Chromatin annotations
- Gene expression data
- ...
- Your own observations: put them in context

Genome “features”

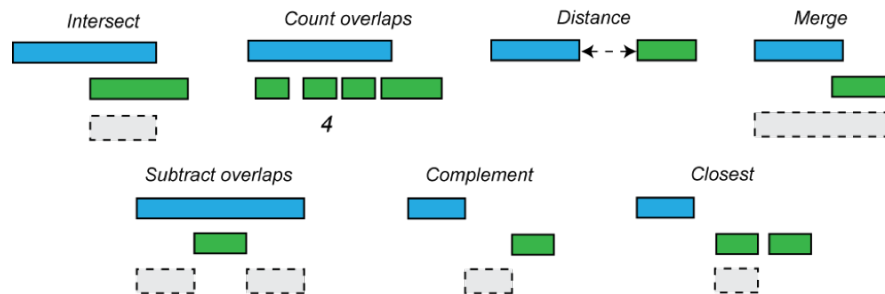


Overlaps, closest, merge...



What is genome arithmetic?

"Set theory on the genome"



Answerable questions

- 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?

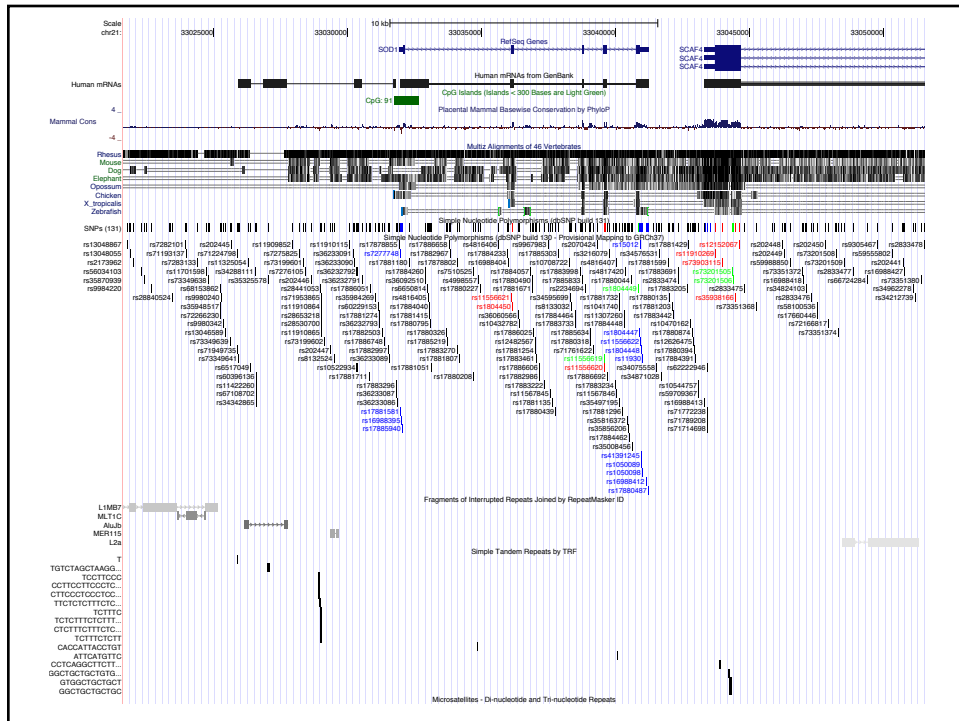
An annoying wealth of formats

- BED
- BEDGRAPH
- WIG
- GFF
- VCF
- SAM/BAM
- ...

Standard attributes

- chrom
- start position
- end position
- name / label
- score / value
- strand
- other

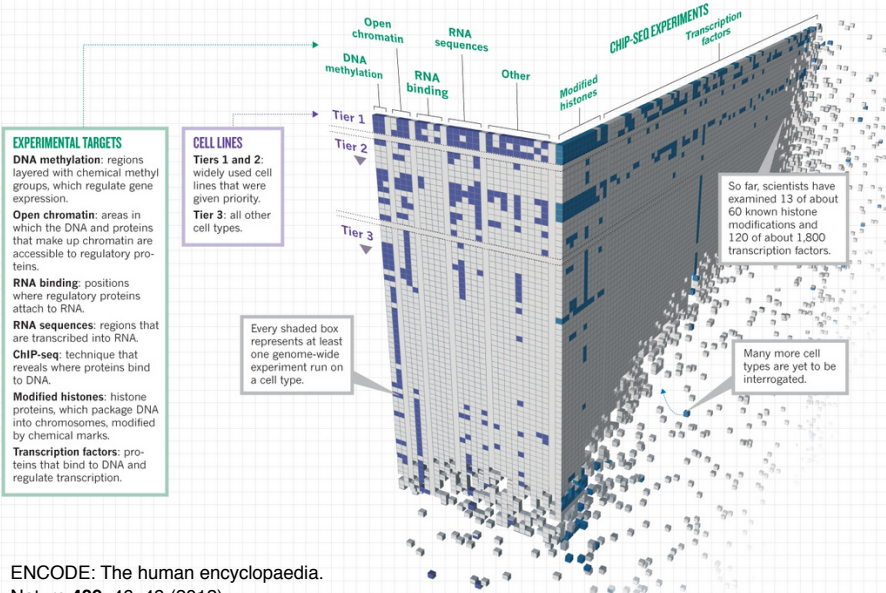
No standards: Some formats use 1-based coordinates, while others use 0-based



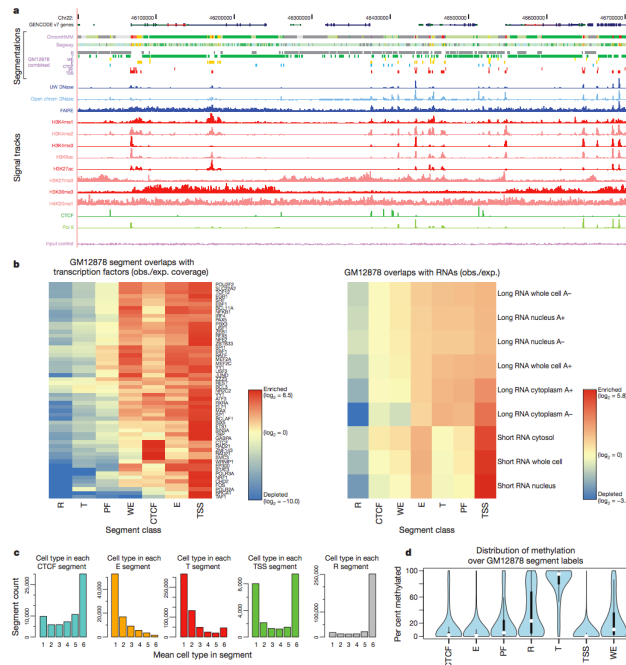
Why?

MAKING A GENOME MANUAL

Scientists in the Encyclopedia of DNA Elements Consortium have applied 24 experiment types (across) to more than 150 cell lines (down) to assign functions to as many DNA regions as possible — but the project is still far from complete.



ENCODE: The human encyclopaedia.
Nature **489**, 46–48 (2012).



<http://www.nature.com/nature/journal/v489/n7414/pdf/nature11247.pdf>

Searching for overlaps

- Brute force: loop over all N features
 - Can be tragically slow when N is large
- Build a “tree”
 - R-tree
 - NCList
 - UCSC “binning” algorithm
- Extend the sort & “merge” algorithm
 - Widely used to “join” database table

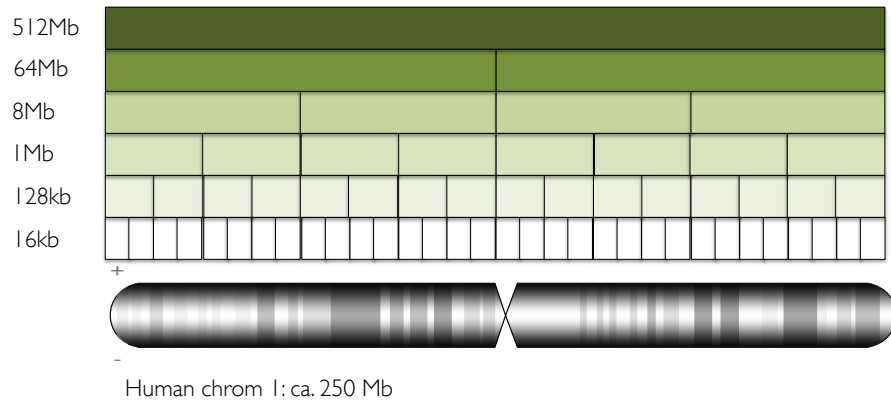
Brute force: sloooooow

```
foreach fA in file A
{
    foreach fB in file B
    {
        does fA overlap fB?*
    }
}
```

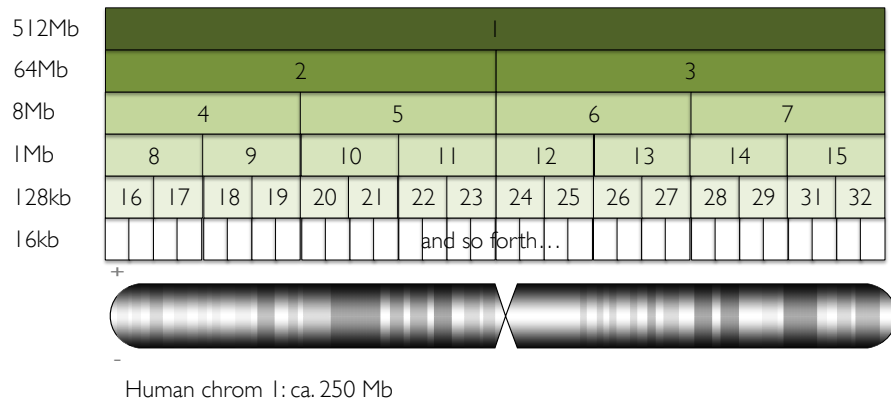


* When A and B contain many features, the answer will be NO the vast majority of the time. This means most of the processing is spent NOT finding features that overlap.

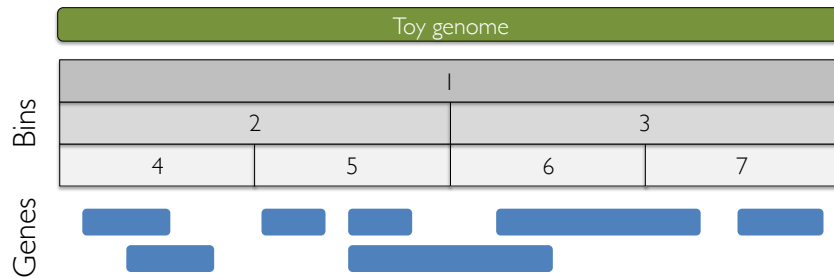
Narrow the search space w/ “binning” the UCSC genome browser approach



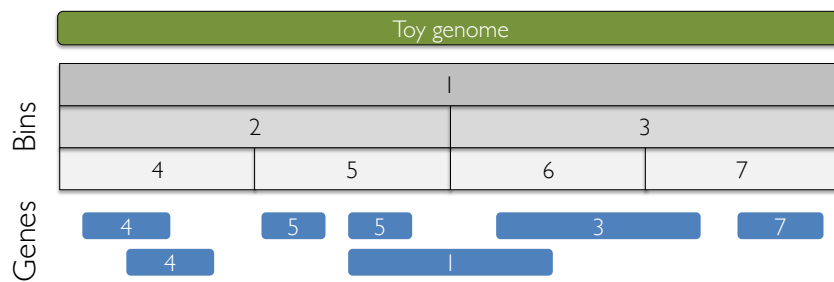
Narrow the search space w/ “binning” the UCSC genome browser approach



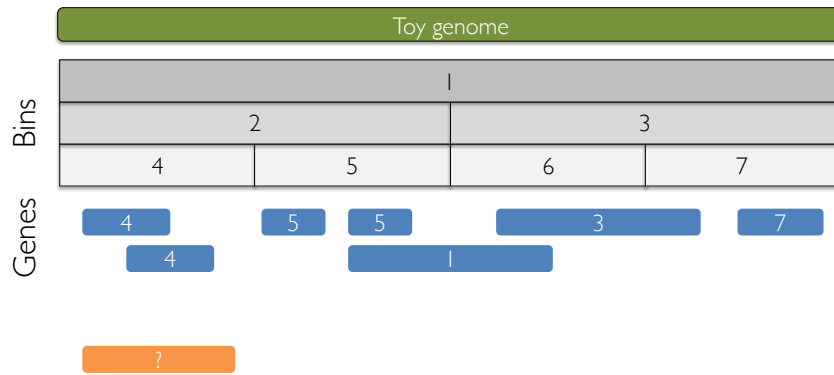
A toy example



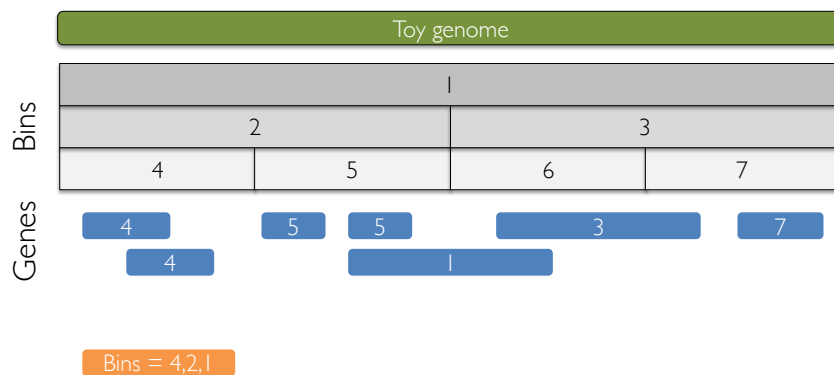
A toy example



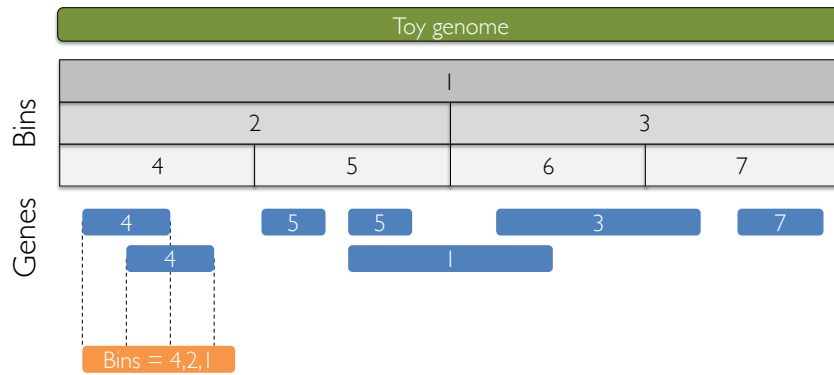
A toy example



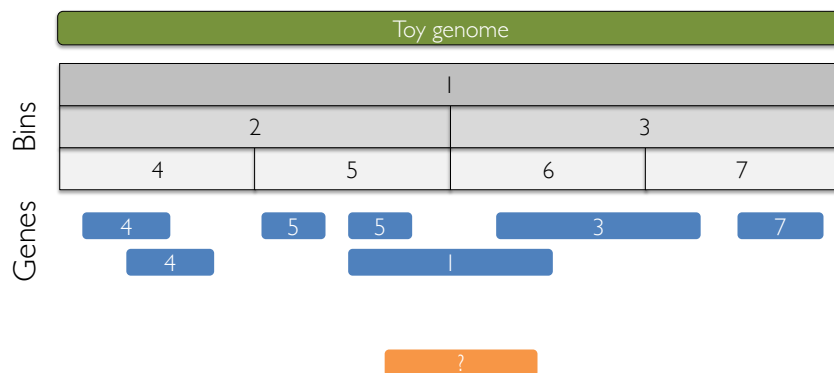
A toy example



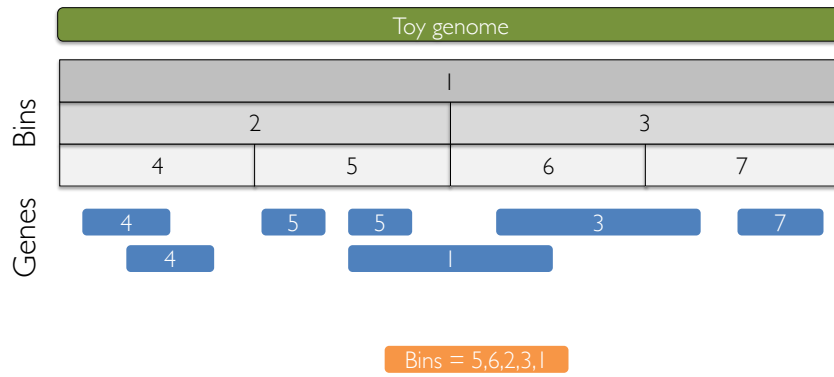
A toy example



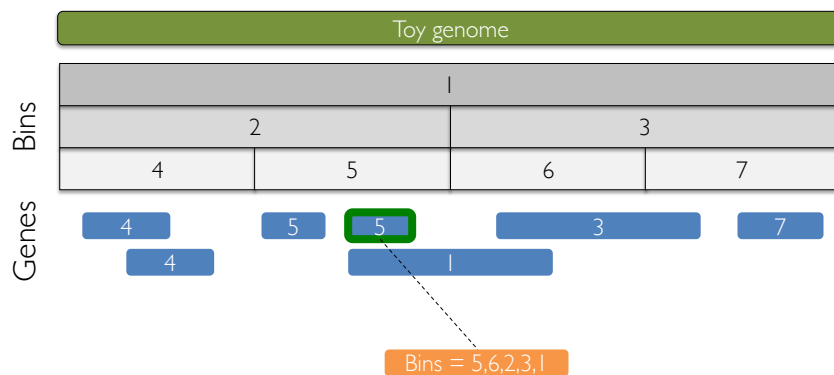
A toy example



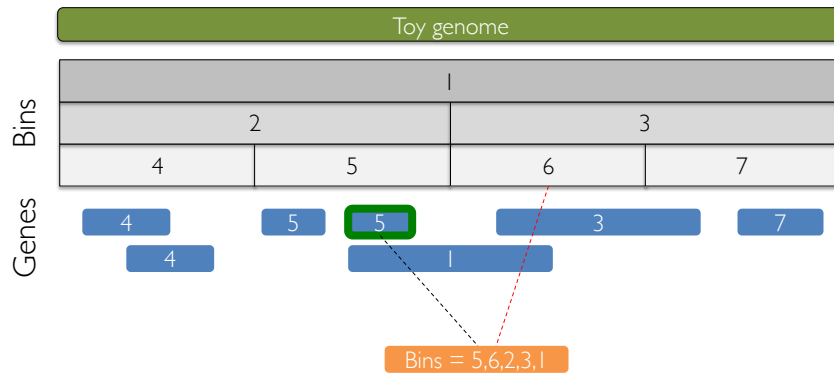
A toy example



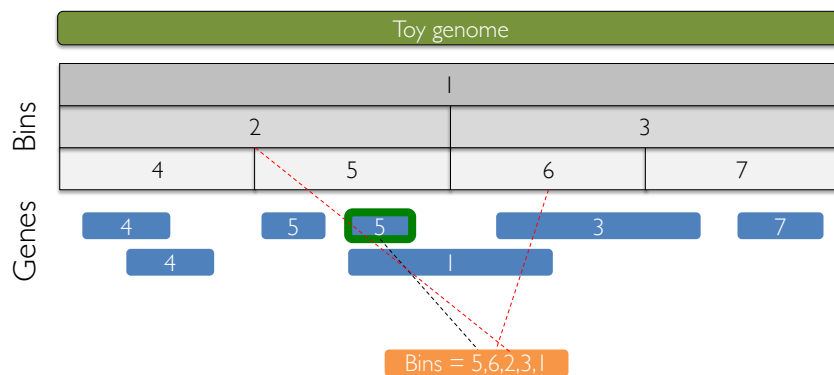
A toy example



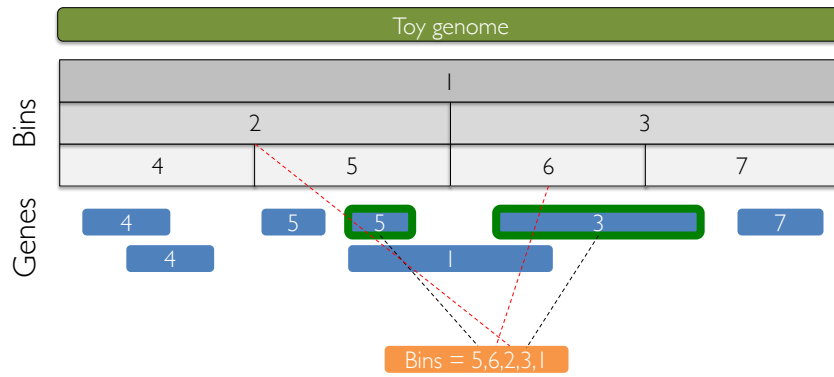
A toy example



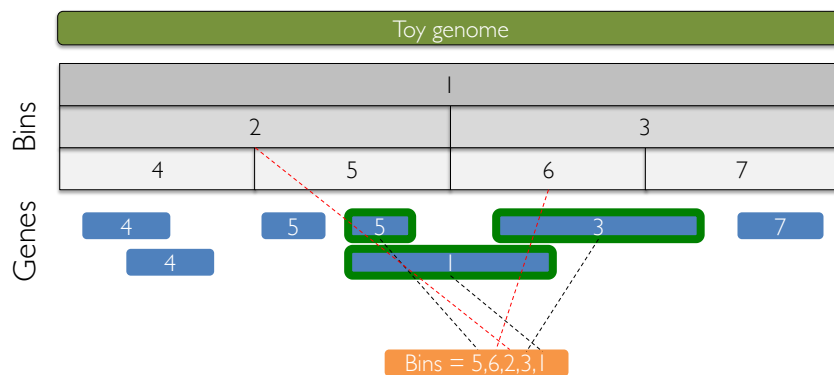
A toy example



A toy example



A toy example



Available tools for GA

UCSC Genome Browser

Galaxy

[BEDTools](#)

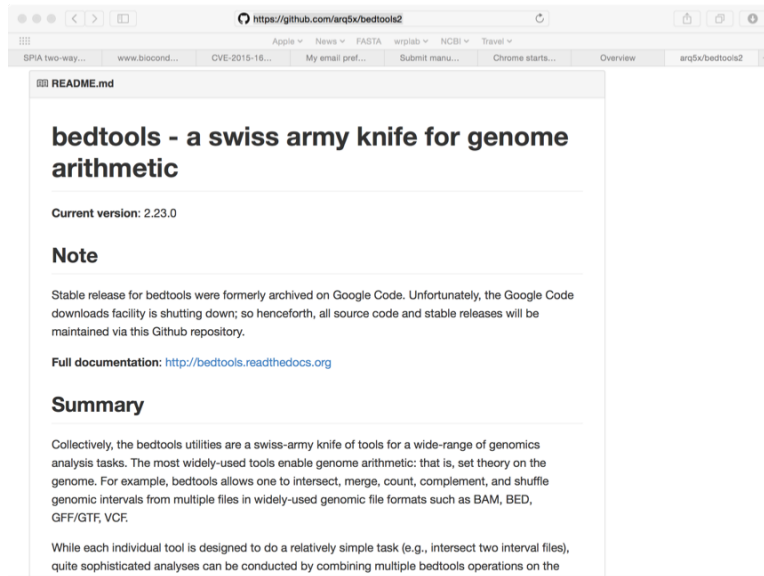
UCSC Genome Browser

<http://genome.ucsc.edu/cgi-bin/hgTables?command=start>

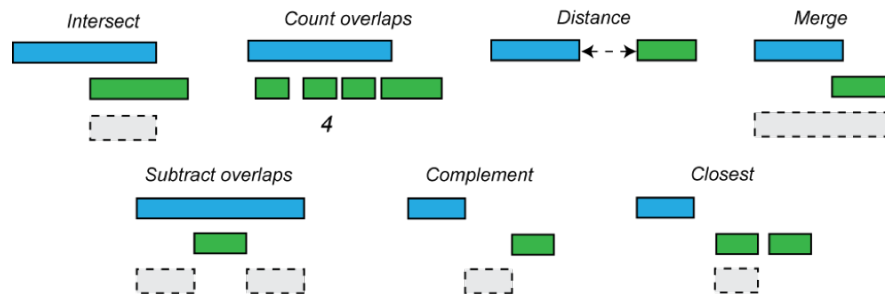
The screenshot shows the UCSC Genome Browser Table Browser interface. The browser window has a title bar and a navigation bar with links: Home, Genomes, Genome Browser, Blast, Tables, Gene Sorter, PCR, Session, FAQ, Help. The main content area is titled "Table Browser" and contains a form for querying genomic data. The form includes fields for "clade" (set to "Human"), "genome" (set to "Human"), "assembly" (set to "Feb. 2009 (GRCh37)"), "group" (set to "Comparative Genomics"), "track" (set to "Comparative Genomics"), "table" (set to "Private Cons (phyloP/cons/PhastCons)"), "region" (set to "genome"), "filter" (set to "create"), "subtract merge" (set to "create"), "intersection" (set to "create"), "correlation" (set to "create"), "output format" (set to "Send output to"), "output file" (set to "leave blank to keep output in browser"), "file type returned" (set to "plain text"), and "Note" (set to "return more than 100,000 lines, change the filter setting (above). The entire data set may be available for download as a very large file that contains the original data values (not compressed into the wiggle format) -- see the Download page."). Below the form is a section titled "Using the Table Browser" with a list of bullet points explaining the form fields: "clade: Specifies which clade the organism is in.", "genome: Specifies which organism data to use.", "assembly: Specifies which version of the organism's genome sequence to use.", "group: Selects the type of tracks to be displayed in the track list. The options correspond to the track groupings shown in the Genome Browser. Select 'All Tracks' for an alphabetical list of all available tracks in all groups. Select 'All Tables' to see all tables including those not associated with a track.", "database: (with 'All Tables' group option) Determines which database should be used for options in table menu.", "track: Selects the annotation track data to work with. This list displays all tracks belonging to the group specified in the group list.", "table: Selects the SQL table data to use. This list shows all tables associated with the track specified in the track list.", "describe table schema: Displays schema information for the tables associated with the selected track.", "region: Restricts the query to a particular chromosome or region. Select 'genome' to apply the query to the entire genome or 'ENCODE' to examine only the ENCODE regions. To limit the query to a specific position, type a

BEDTools

<https://github.com/arq5x/bedtools2>



BEDTools supports common GA operations and common formats



BEDTools + UNIX

Real world usage

Find SNPs that have the potential to
alter gene expression regulation by
affecting methylation at CpG islands.

(restrict to chrom 1)

Step 1: Get annotations

- Single-nucleotide polymorphisms (SNPs)
- CpG islands
- Genes

SNPs from dbSNP via UCSC

The workflow consists of the following steps:

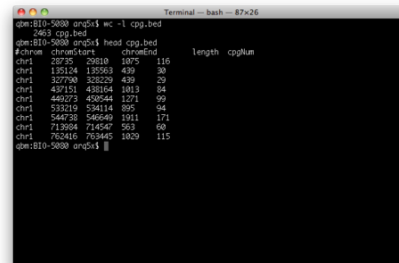
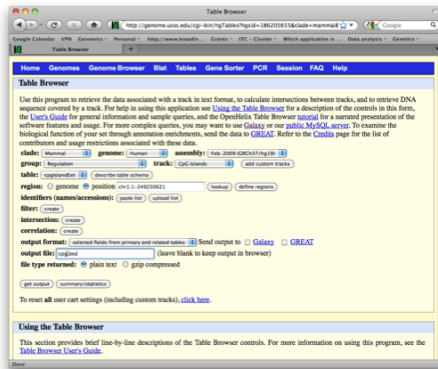
- Table Browser:** The UCSC Table Browser interface is shown with the 'Table Browser' tab selected. The 'Track' dropdown is set to 'SNPs from dbSNP'. The 'Table' dropdown is set to 'rs'. The 'Output format' is set to 'Table'. The 'Output file' is set to 'snps.bed'.
- Select Fields:** The 'Select Fields from hg19.snp131' window is shown. The 'Fields' list includes 'chr', 'pos', 'ref', 'alt', 'rsid', 'name', 'score', 'strand', 'observed', and 'func'. The 'Fields' list is checked, and the 'OK' button is clicked.
- Table:** The resulting table of SNP data is shown. The table has columns for 'chr', 'pos', 'ref', 'alt', 'rsid', 'name', 'score', 'strand', 'observed', and 'func'. The data is sorted by 'pos'.
- Terminal:** A terminal window shows the command `gzip -d snps.bed` and the output of the command, which is a list of SNPs with their coordinates and annotations.

The output of the terminal command is as follows:

```
gzip -d snps.bed
2064272 snps.bed
#chrom chromstart chromend name score strand observed func
chr1 104118 104122 rs55229806 0 + C/T near-gene-5
chr1 10491 10492 rs55998931 0 + C/T near-gene-5
chr1 105118 105119 rs52639580 0 + C/G near-gene-5
chr1 105125 10513 rs52181548 0 + A/G near-gene-5
chr1 10527 10528 rs18219492 0 + A/G near-gene-5
chr1 10580 10584 rs18219412 0 + A/G near-gene-5
chr1 10595 10627 rs18218527 0 + A/G near-gene-5
chr1 10937 10938 rs12857367 0 + A/G near-gene-5
chr1 1100 1100 rs76373904 0 + A/C near-gene-5
gzip -d snps.bed
```

"snps.bed"
N = 2,064,872

CpG islands via UCSC



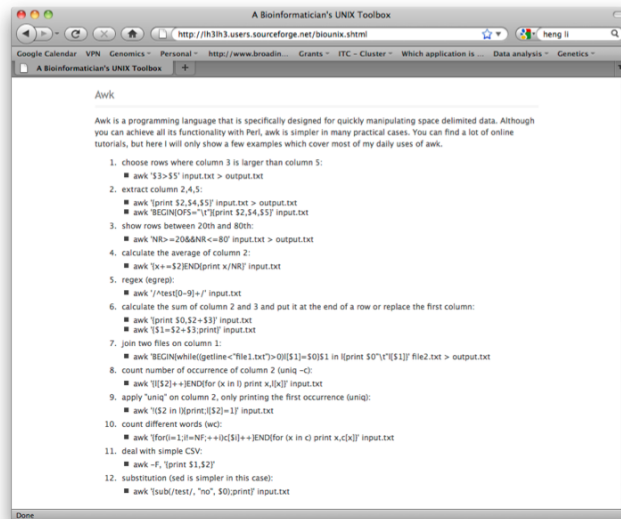
"cpg.bed"
N = 2,463

Problem: No name and no strand in the output. Let's fix it..

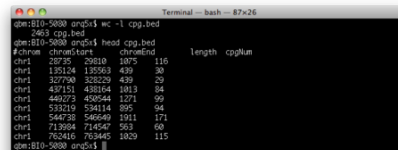
The awesome power of *awk*

- written in 1977 by Alfred Aho, Peter Weinberger, and Brian Kernighan
- Similar constructs as Perl (\$1, \$2, \$3, etc.)
- Typically used for filtering, summarizing, or reorganizing files.
- "One liners", used on files or "streams"
- `awk '{print "Hello World!\n"}'` (Print "Hello World!")
- `awk '$2 >= 30' foo.txt` (Get lines in foo where 2nd col. is g.t.e. 30)
- `awk '{print $3,$2,$7,$1}' foo.txt > foo.reordered.txt`

The awesome power of *awk*



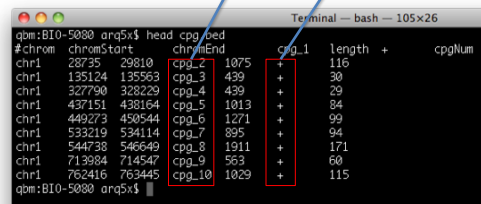
Use awk to fix cpg.bed



Problem: No name and no strand in the output. Let's fix it..

```
awk '{OFS="\t"; print $1,$2,$3,"cpg_"NR,$4,"+", $5}' cpg.bed > cpg.full.bed  
mv cpg.full.bed cpg.bed
```

OFS — output field separator; NR — record number



[illegible]

```

[1] chr10:5038 chr5:5 uc - 1 genes-bed
[2] bedcov gene-bed
chr10:5038 chr5:5 head genes-bed
chr1 1481 2070 MM.A00640 0 0 29270 29270 0 11 469,69,152,159,166,136,137,147,20,154,50, 0,600,1434,2245,2466,2871,3244,3553,3966,10376,14971,
chr1 34610 36041 MM.A02011 0 0 36041 36041 0 3 564,335,381, 0,666,1110,
chr1 34610 36041 MM.A02019 0 0 36041 36041 0 3 564,335,381, 0,666,1110,
chr1 69040 70086 MM.N01915404 0 0 69040 70086 0 1 911, 0,
chr1 323891 323891 MM.A03820 0 0 323891 323891 0 4 107,50,250,1560,396,547,3144,
chr1 323891 323891 MM.A03822 0 0 323891 323891 0 3 169,58,4143, 0,396,547,
chr1 323891 323891 MM.A03225 0 0 323891 323891 0 3 169,58,4143, 0,396,547,
chr1 367658 367658 MM.A0195777 0 0 367658 367658 0 1 793, 0,
chr1 367658 367658 MM.A0195224 0 0 367658 367658 0 3 190, 0,
chr1 367658 367658 MM.A0195221 0 0 367658 367658 0 1 930, 0,
chr10:5038 chr5:5

```

Let's put it all together by using BEDTools

Step 1: Find SNPs in CpG islands

```
# How many SNPs are there on chrom 1?
wc -l snps.bed
2064872
# Find SNPs that overlap CpG islands using intersectBed
bedtools intersect -a snps.bed -b cpG.bed -wa >
snps.cpg.bed
# How many SNPs overlapped a CpG island?
wc -l snps.cpg.bed
14974
```

BEDTools typically puts the second file in memory, and reads the first file line-by-line, so put the smaller file second.

-wa report the intersecting -a region, normally, only the intersection is reported

So, 0.7% of the SNPs were in CpG islands. Is this sensible?

Step 1: Find SNPs in CpG islands

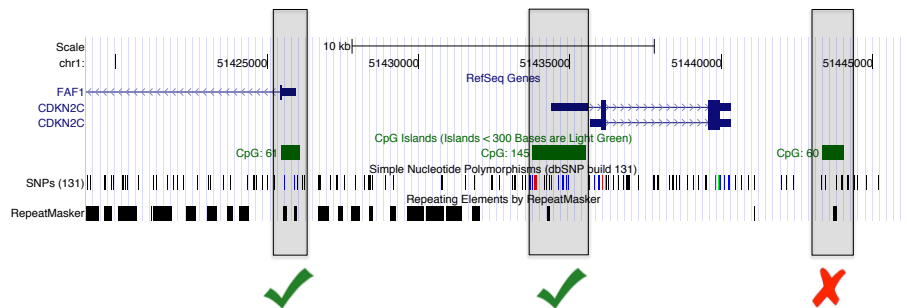
```
# How many SNPs are there on chrom 1?
wc -l snps.bed
2064872
# Find SNPs that overlap CpG islands using intersectBed
bedtools intersect -a snps.bed -b cpG.bed -wa > snps.cpg.bed
# How many SNPs overlapped a CpG island?
wc -l snps.cpg.bed
14974
```

So, 0.7% of the SNPs were in CpG islands. Is this sensible?

```
# How much of chrom1 do the CpG islands occupy?
awk '{sum+=$7} END{print sum}' cpG.bed
1881629
```

$1,881,629 / 249,250,621 = 0.75\%$

Step 2: Find TSS/promoter CpG islands



Step 2: Find TSS/promoter CpG islands

use “window” from BEDTools

By default, bedtools window adds 1000 bp upstream and downstream of each A feature and searches for features in B that overlap this “window”. If an overlap is found in B, both the *original* A feature and the *original* B feature are reported (expands window for intersection, produces more columns):

```
$ cat A.bed
chr1 100 200
$ cat B.bed
chr1 500 1000
chr1 1300 2000
$ bedtools window -a A.bed -b B.bed
chr1 100 200 chr1 500 1000
```

```
bedtools window -l 10000 -r 0 -sw -a genes.bed -b cpG.bed | \
cut -f 13-19 | \
sort | \
uniq > cpG.gene_impact.bed
```

-l (add to left coordinate)
-r (add to right coordinate)
-sw (add/subtract based on strand)

Step 3: Find SNPs in TSS/promoter CpG islands

use intersectBed from BEDTools

```
bedtools intersect -a snps.cpg.bed -b  
cpg.gene_impact.bed > snps.cpg.gene_impact.bed
```

Step 3: Find SNPs in TSS/promoter CpG islands

use intersectBed from BEDTools

```
bedtools intersect -a snps.cpg.bed -b  
cpg.gene_impact.bed > snps.cpg.gene_impact.bed
```

What types of SNPs are they? Eg., A→G, G→T

```
grep -v "\-" snps.cpg.gene_impact.bed | \  
cut -f 7 | \  
sort | \  
uniq -c
```

BEDTools

- Genome file types/formats
 - format types
 - BED files
- Overview of genome features
- Genome arithmetic
 - approaches
 - UCSC "binning" algorithm
- Introduction to BEDtools
 - intersection / window / many more

Analysis of transcription regulation

1. Affy-chip/RNA-seq for RNA expression levels
2. Identify differentially expressed genes (edgeR, DESeq2)
 - (possibly) identify subsets of genes associated with different pathways
3. Isolate regions of DNA around (pathway-specific) coordinately expressed (induced) genes
 - BEDtools
4. Take collections of candidate regulatory regions and look for common DNA motif
 - meme, HOMER