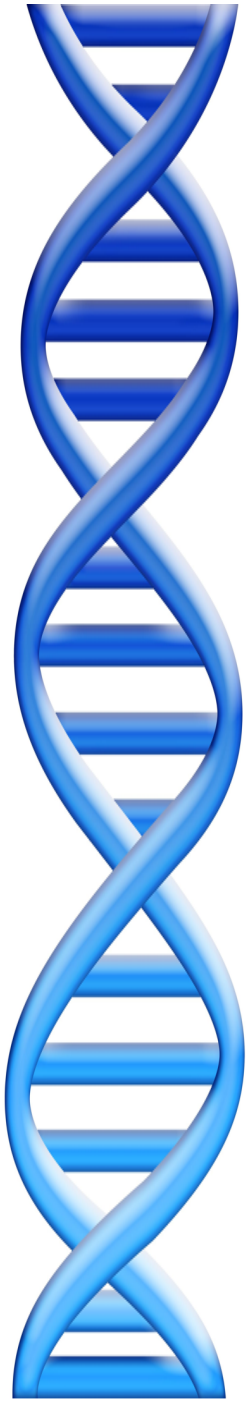
Computational Analysis Primer

Michael Schatz & Justin Kinney

Nov 8, 2011
QB Lecture 2



Outline

Part 1: Overview & Fundamentals

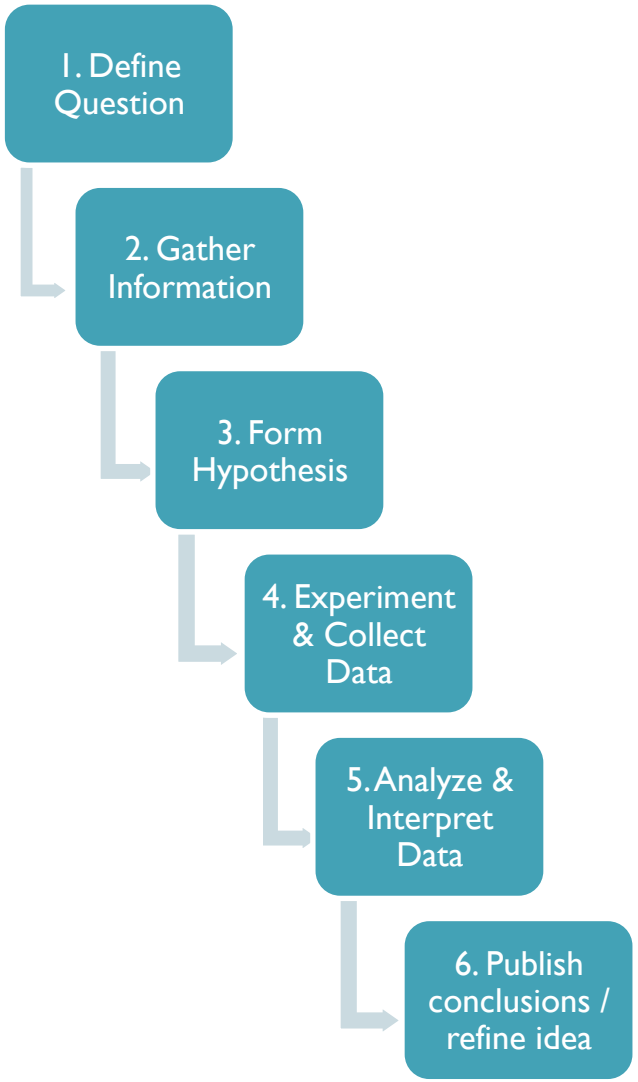
- Why Computers?
- Overview of Computation Systems
- Unix and Scripting Primer

Part 2: Example Analysis

Scientific Method

What is analysis?

- Experimental design
  - Frame the question so that it can be quantitatively answered
- Assay design
  - Statistical, mathematical, computational methods to improve the sensors
- Drawing conclusions
  - Identify trends, patterns, correlations, and causal links



How do we analyze?

- Paradigms of science:
  1. Make observations
  2. Formulate mathematical models
  3. Simulate processes
  4. Data-intensive discovery

How do we draw conclusions?

- Comparison & Triangulation: How does X compare to Y?

X	Y
Exomes of kids with autism	Exomes of kids that do not
Genomes of Europeans	Genomes of non-Europeans, mammals, …
Gene expression in mutants	Gene expression in wild type
Firing patterns of mutant fly neurons	Firing patterns of wild type

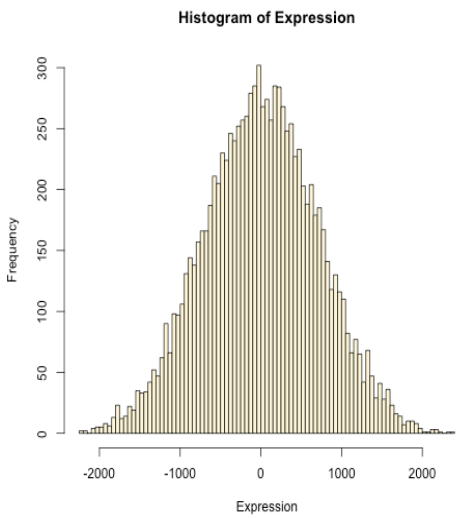
- Modeling & Predictions: How will X respond to Y?

X	Y
Mutant tomatoes	Increased temperatures
Human Microbiome	Probiotic treatments
Gene expression in mice	Knockout of transcription factor
Firing rate in flies	Decreased sodium levels

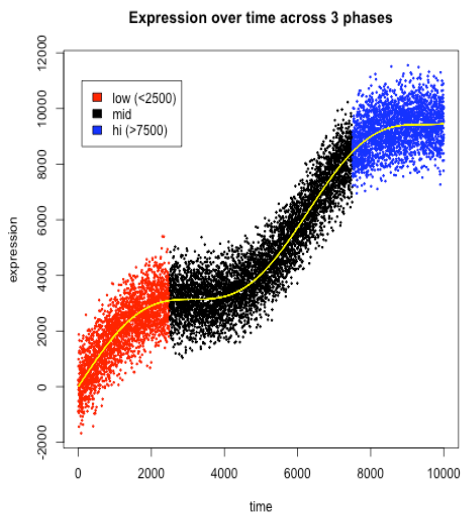
How do we DRAW conclusions?

```
-902.473
 242.817
-872.453
 73.9297
 236.169
 46.7525
 975.014
 716.563
-533.971
-120.282
  725.12
 -736.76
 176.156
 189.224
 1847.46
-159.099
-56.4754
-973.626
  1181.9
-315.455
-1480.43
 215.293
-747.505
 682.577
    …
```

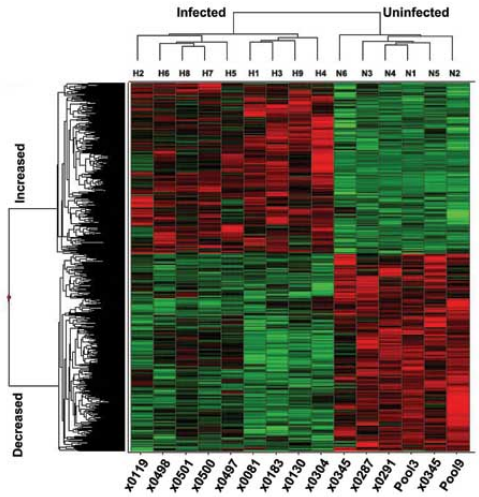
Histogram



Scatterplot



Heatmap



Data and data transformations are ubiquitous in science

Data are too numerous and transformations are too complex to do by hand

==> Mendel: 100k observations, 10 years

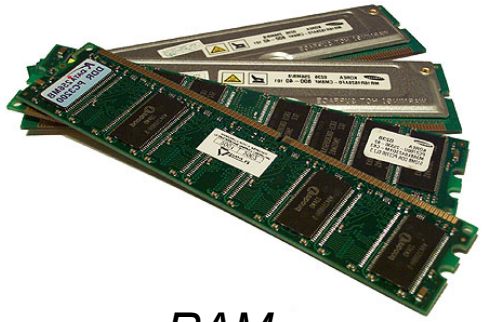
==> HiSeq 2000: 600B observations, 10 days

==> Make friends with your computational tools

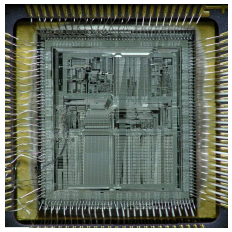
What is a computer?

[hardware]

Hard Drive
Permanent Storage – 1TB
(big, slow, cheap)



RAM
Working Storage – 8 GB
(small, fast, expensive)



Processor
Arithmetic, logic
cores, clock speed

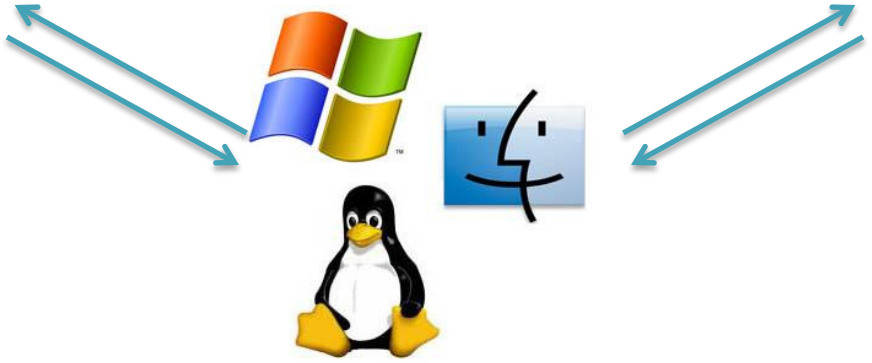
Display
Human Interface

Network
Computer Interface
Home: 10Mb/s, CSHL: 1Gb/s

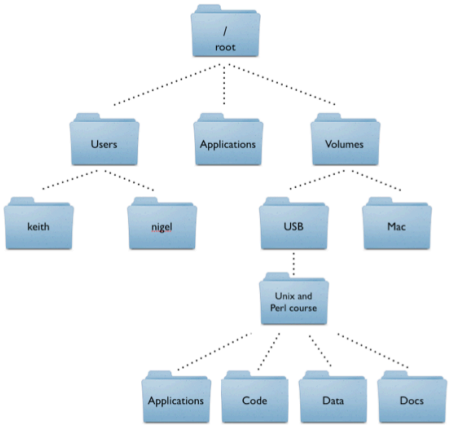
What is a computer?

[software]

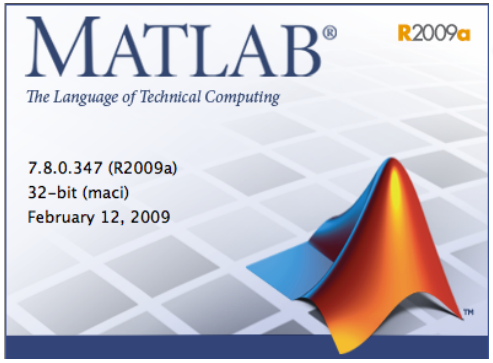
Office Applications
Presentations, Documents
Simple statistics and plots



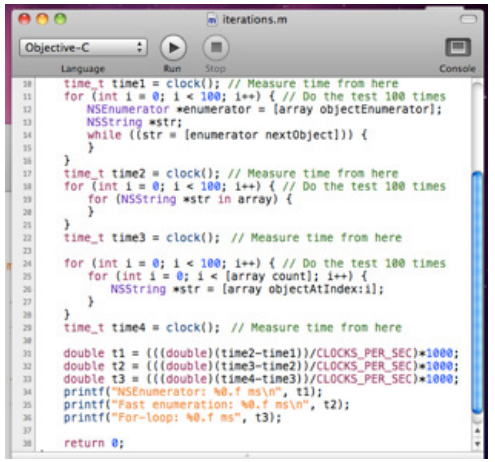
Operating System
Mission Control
Windows, Mac, Unix, iOS



Files / Data
Papers, sequences,
measurements



Scientific Applications
Specialized Analysis
Commercial

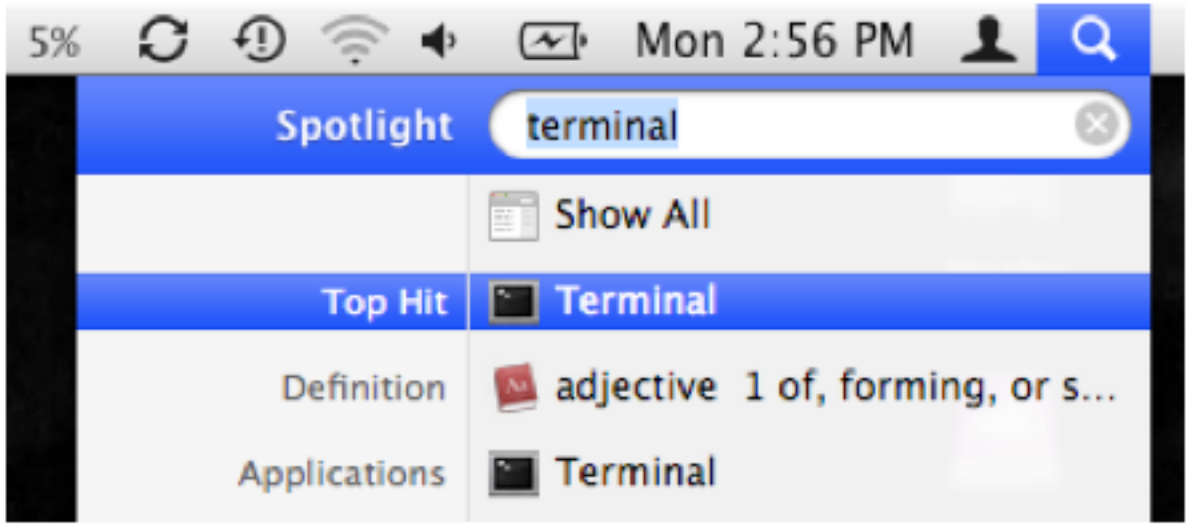


Code / Scripts
Research Applications
Academic

How does (scientific) software operate?

- The software we need to run is very specialized, there is no 'align genomes' button in Excel
  - Data files are huge, so probably wouldn't want one anyways
- It takes a lot of work (and time/money) to create a graphical interface to software, so most scientific software uses a 'command line' interface
  - Important to become comfortable using command line tools
- Scientific analyses tend to use workflows consisting of several applications where the output of one phase becomes the input to the next
  - Develop a workflow for dataset X, apply again to dataset Y

Where is the command line?



- Your Mac has a very powerful command line interface hidden just beneath the graphical environment
  - This command line interface is (basically) the same as that used by our scientific cluster BlueHelix
  - Big data files are stored on our central storage system BlueArc

- This environment has a universe of programs you can use to manipulate files and data in novel ways
  - Learning to use this environment is a lot like learning a new language
  - http://korflab.ucdavis.edu/Unix_and_Perl/index.html

Hola, como estas?

Command	Output
man	Look up something in the manual (also try Google)
ls	List the files in the current directory
cd	Change to a different directory
pwd	Print the working directory
mv, cp, rm	Move, copy, remove files
mkdir, rmdir	Make or remove directories
cat, less, head, tail, cat	Display (parts) of a text file
echo	Print a string
sort, uniq	Sort a file, get the unique lines
find, grep	Find files named X, or containing X
chmod	Change permissions on a file
wc	Count lines in a file
jot / seq	Output numbers from 1 to X (on Linux use seq)
| (pipe), > (redirect)	Send output to a different program, different file

File Hierarchy

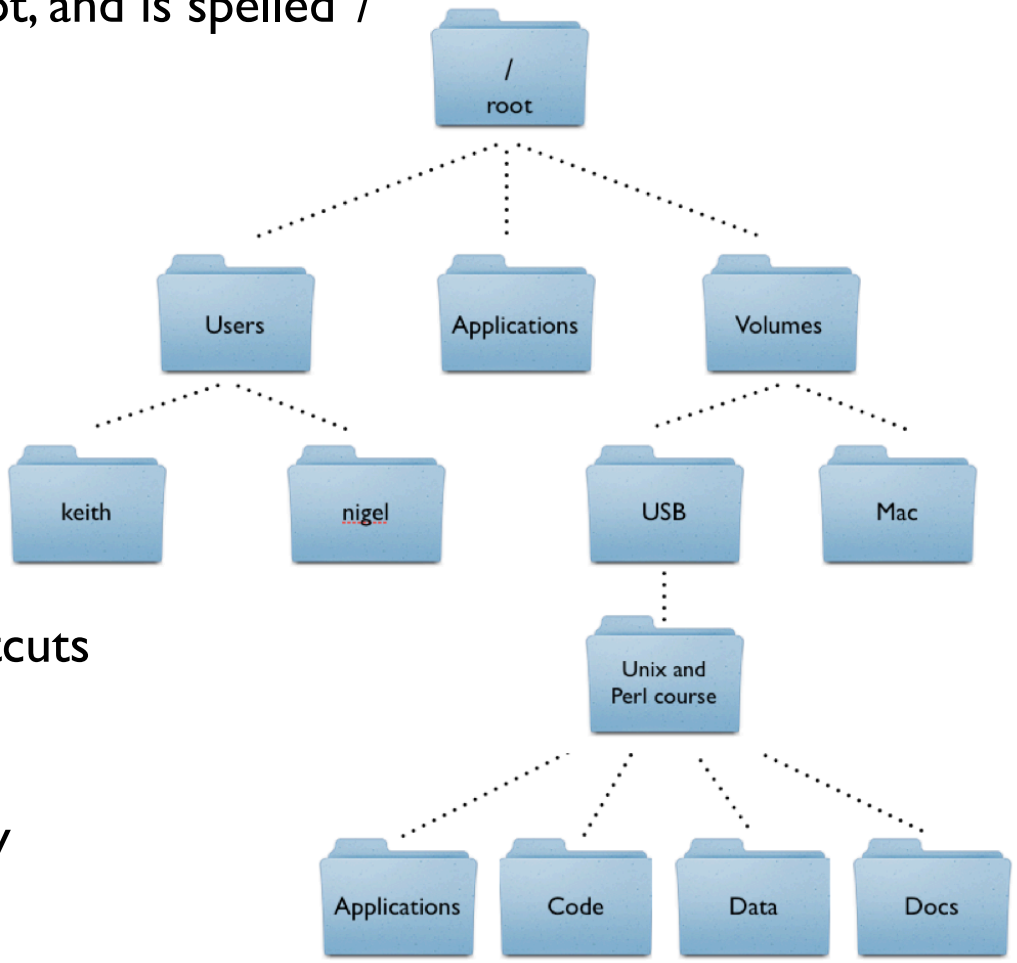
Files are stored in nested directories (folders) that form a tree

- The top of the tree is called the root, and is spelled '/'

- Your home directory (on mac) is at
  /Users/username

- Command line tools are at
  /bin/
  /usr/bin/
  /usr/local/bin/

- A few special directories have shortcuts
  ~ = home directory
  ~bob= bob's home directory
  . = current working directory
  .. = parent directory
  - = last working directory



Working with the shell

- The shell is interactive and will attempt to complete your command as soon as you press enter

```
$ pwd
/Users/mschatz

$ echo “Hello, World”
Hello, World
```

- Here are a few tips that will make your life easier

Command	Effect
Left/Right arrow	Edit your current command
Up/Down arrow	Scroll back and forth through your command history
Control-r	Search backwards through your command history
history	What commands did I just run?
Control-c	Cancel the command
Control-u	Clear the current line
Control-a, Control-e	Jump to the beginning and end of the line

Files and permissions

- Every file has an owner and a group, you can only read/write to a file if you have permission to do so

```
$ pwd
/Users/mschatz/Desktop/Unix_and_Perl_course/Data/Arabidopsis

$ ls -l
total 193976
-rw-r--r--@ 1 mschatz  staff  39322356 Jul  9  2009 At_genes.gff
-rw-r--r--@ 1 mschatz  staff  17836225 Oct  9  2008 At_proteins.fasta
-rw-r--r--@ 1 mschatz  staff  30817851 May  7  2008 chr1.fasta
-rw-r--r--@ 1 mschatz  staff  11330285 Jul 10  2009 intron_IME_data.fasta
```

- These files can be read by anyone, but only written by me
  - Change permissions with 'chmod'

```
$ chmod g+w At_*
$ man chmod
```

- Programs and scripts have the execute bit set

```
$ ls -l /bin/ls
-r-xr-xr-x  1 root  wheel  80688 Feb 11  2010 /bin/ls*
```

Working with files and directories

- Create directories and copies of the working files

```
$ mkdir myfiles
$ cd myfiles/
$ cp ../At_* .
$ ls -l
total 111648
-rw-r--r--@ 1 mschatz  staff  39322356 Nov  8 01:37 At_genes.gff
-rw-r--r--@ 1 mschatz  staff  17836225 Nov  8 01:37 At_proteins.fasta
```

- Rename files

```
$ mv At_genes.gff Arabidopsis_genes.gff
```

- See how long the files are

```
$ wc -l *
  531497 Arabidopsis_genes.gff
  214021 At_proteins.fasta
  745518 total
```

- Clean up

```
$ cd ..
$ rm -rf myfiles/
```

[WARNING!!! Double check rm]

Working with text files

- Display the first few lines of a file

```
$ head -5 At_proteins.fasta
>AT1G51370.2 | Symbols:  | F-box family protein | chr1:19049283-19050416 FORWARD
MVGGKKKTKICDKVSHEEDRISQLPEPLISEILFHLSTKDSVRTSALSTKWRYLWQSVPGLDLDPYASSNTNTIVSFVES
FFDSHRDSWIRKLRLDLGYHHDKYDLMSWIDAATTRRIQHLDVHCFHDNKIPLSIYTCTTLVHLRLRWAVLTNPEFVSLP
CLKIMHFENVSYPNETTLQKLISGSPVLEELILFSTMYPKGNVLQLRSDTLKRLDINEFIDVVIYAPLLQCLRAKMYSTK
NFQIISSGFPAKLDIDFVNTGGRYQKKKVIEDILIDISRVRDLVISSNTWKEFFLYSKSRPLLQFRYISHLNARFYISDL
```

- Show the first few proteins names in the file

```
$ grep '>' At_proteins.fasta | head -5
>AT1G51370.2 | Symbols:  | F-box family protein | chr1:19049283-19050416 FORWARD
>AT1G50920.1 | Symbols:  | GTP-binding protein-related | chr1:18874223-18876238 FORW
>AT1G36960.1 | Symbols:  | similar to unknown protein [Arabidopsis thaliana] (TAIR:A
>AT1G44020.1 | Symbols:  | DC1 domain-containing protein | chr1:16719132-16721096 RE
>AT1G15970.1 | Symbols:  | methyladenine glycosylase family protein | chr1:5486538-5
```

- Count how many proteins are present, excluding hypothetical proteins

```
$ grep '>' At_proteins.fasta | wc -l
   32825
$ grep '>' At_proteins.fasta | grep -v 'hypothetical' | wc -l
   31267
```

Working with text files 2

- Create a file of just hypothetical proteins

```
$ grep '>' At_proteins.fasta | grep 'hypothetical' > hypotheticals
$ wc -l hypotheticals
    1558 hypotheticals
```

- Count hypotheticals per chromosome

```
$ cut -f4 -d'|' hypotheticals | head -3
 chr1:11437249-11439801 FORWARD
 chr1:5167349-5168146 REVERSE
 chr1:16717096-16717944 FORWARD
$ cut -f4 -d'|' hypotheticals | cut -f1 -d':' | head -3
 chr1
 chr1
 chr1
$ cut -f4 -d'|' hypotheticals | cut -f1 -d':' | sort | uniq -c
 382  chr1
 234  chr2
 260  chr3
 204  chr4
 384  chr5
   9  chrC
  84  chrM
   1 CAB12631.1 (PTHR11061
```

[What happened here?]

Scripting basics

- A bash script is just a list of commands

```
$ cat simple_script.sh
#!/bin/sh

echo "Hello, World”
echo "Shall we play a game?”

$ chmod +x simple_script.sh
$ ./simple_script.sh
```

[What does this do?]

- Things get interesting when we add variables and loops

```
$ cat loop_script.sh
#!/bin/sh

for name in "Mike" "Justin" "Mickey"
do
  echo "Hello, $name" >> people.txt
  everyone="$name $everyone"
done
echo "Hello: $everyone" >> people.txt

$ chmod +x loop_script.sh
$ ./loop_script.sh
$ ./loop_script.sh
$ ./loop_script.sh
```

[What does this do?]

Scripting basics 2

- Conditionals and loops let us work over any number and type of file

```
$ cat conditional_script.sh
#!/bin/sh

for filename in `/bin/ls *`
do
  type=`echo $filename | cut -f2 -d'.'`
  echo "Processing $filename, type is $type"
  echo "=================="

  if [[ $type == "fasta" ]]
  then
    protein_count=`grep -c '>' $filename`
    hypo_count=`grep -c hypothetical $filename`
    echo "$filename has $protein_count total proteins, $hypo_count are hypothetical"
  elif [[ $type == "gff" ]]
  then
    echo "$filename stats"
    cut -f3 $filename | sort | uniq -c
  else
    echo "Unknown file type"
  fi

  echo "=================="
  echo
done
```

[What does this do?]

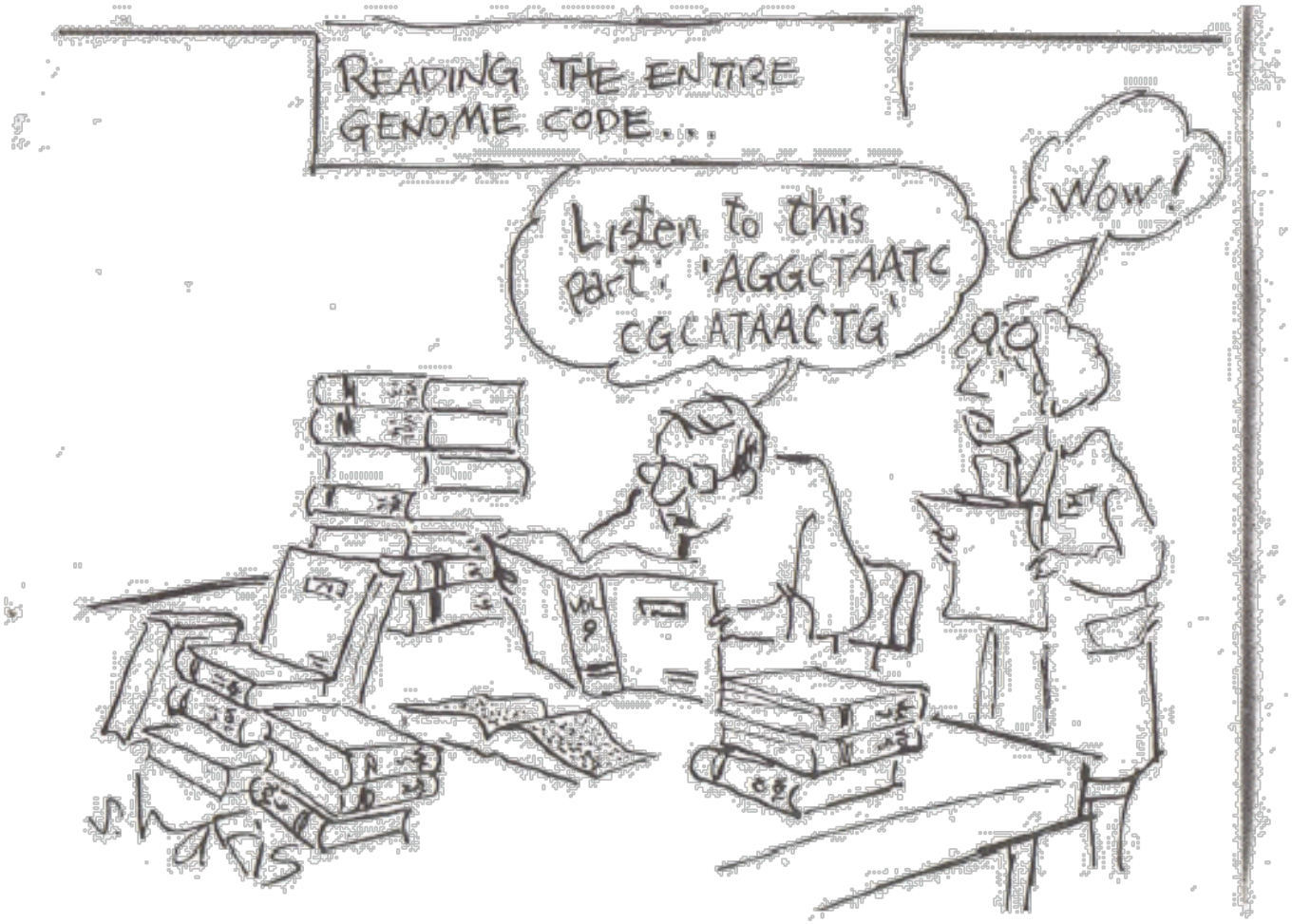
Scripting Challenges

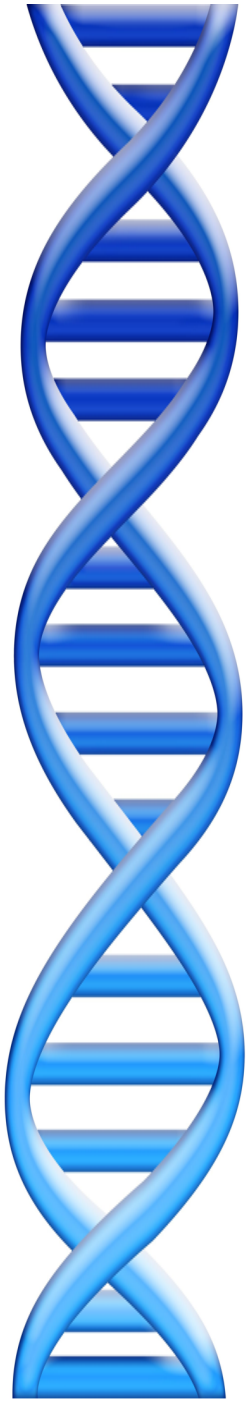
- Create 1000 files named mutantA.X.txt with X in [1,1000] that each contain 'gene'
  - That each contain the numbers 1 to X

- How do I rename 1000 files named mutantA.X.txt to mutantB.X.txt?

- How can I create a directory with just the files that contain 'special gene'

Break


READING THE ENTIRE GENOME CODE...
Listen to this part: 'AGGCTAATC CGCATAACTG'
Wow!



Outline

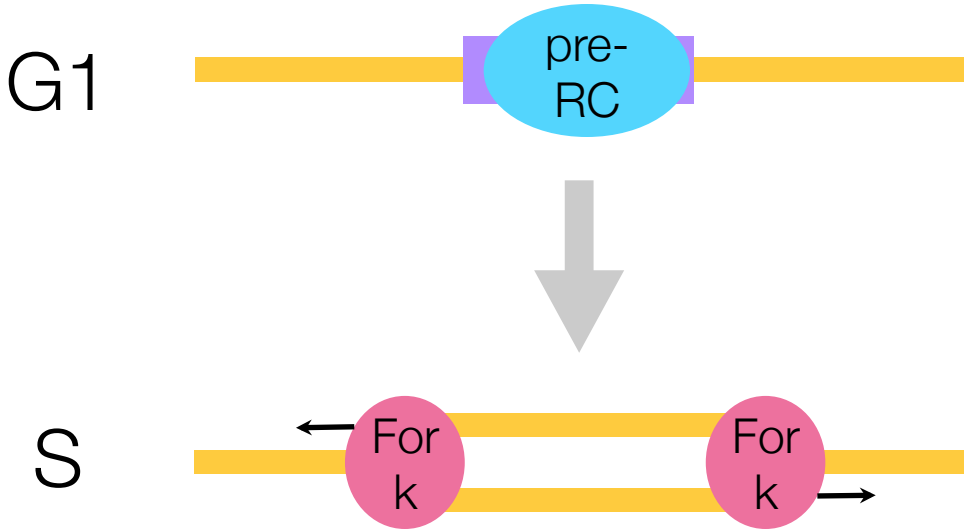
Part 1: Overview & Fundamentals

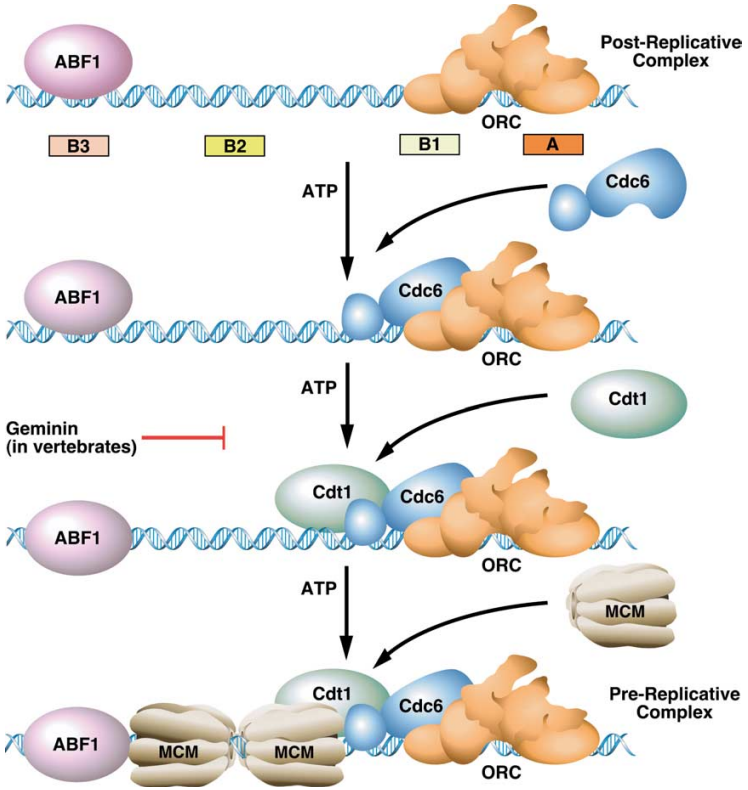
Part 2: Example Analysis

- Background on tracking DNA replication with next-gen sequencing
- Walk-through of analysis steps
- Visualization of discovered replication sites

~300 separate loci direct DNA replication initiation in Saccharomyces cerevisiae

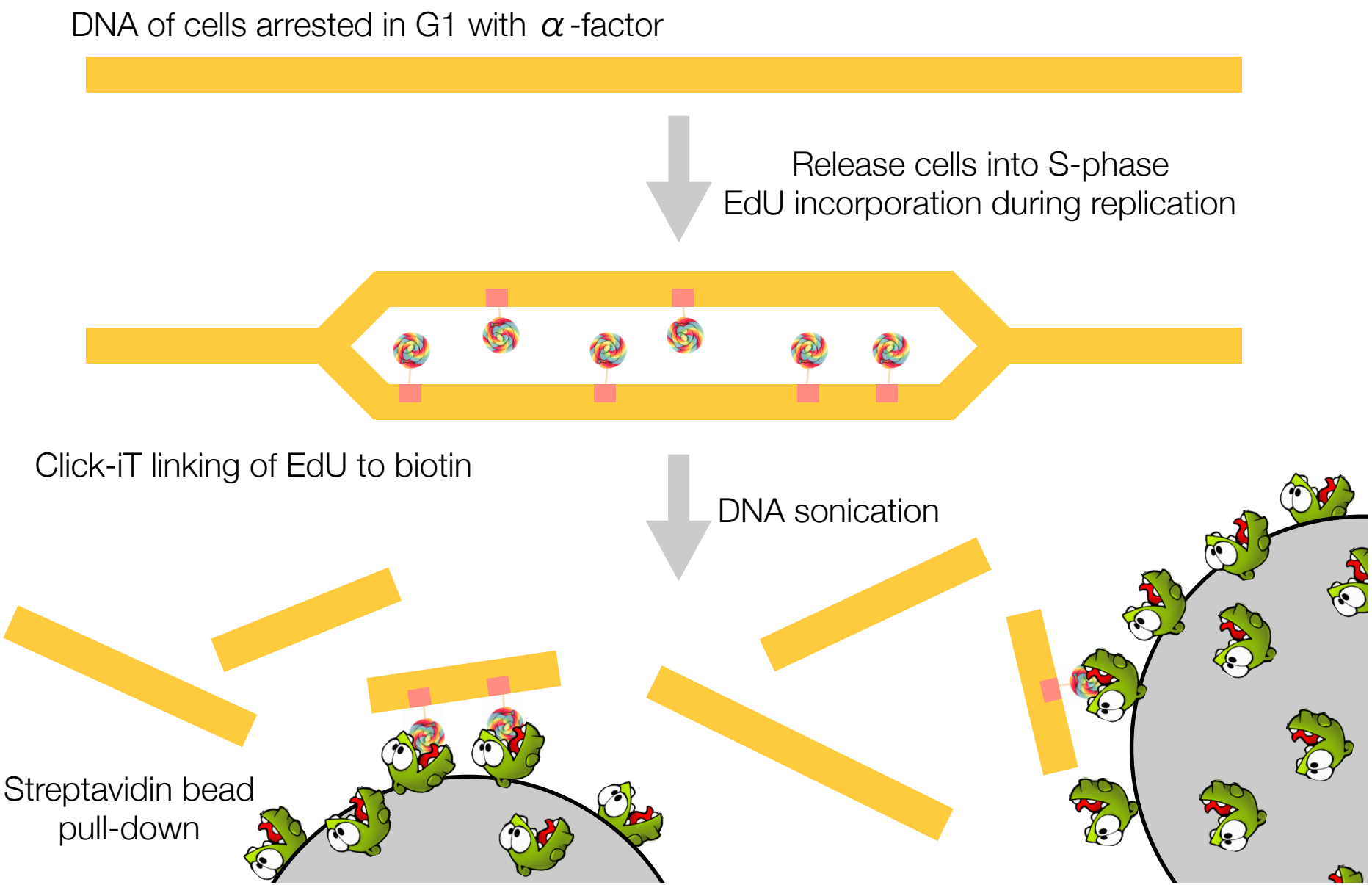
ARS: autonomously replicating sequence



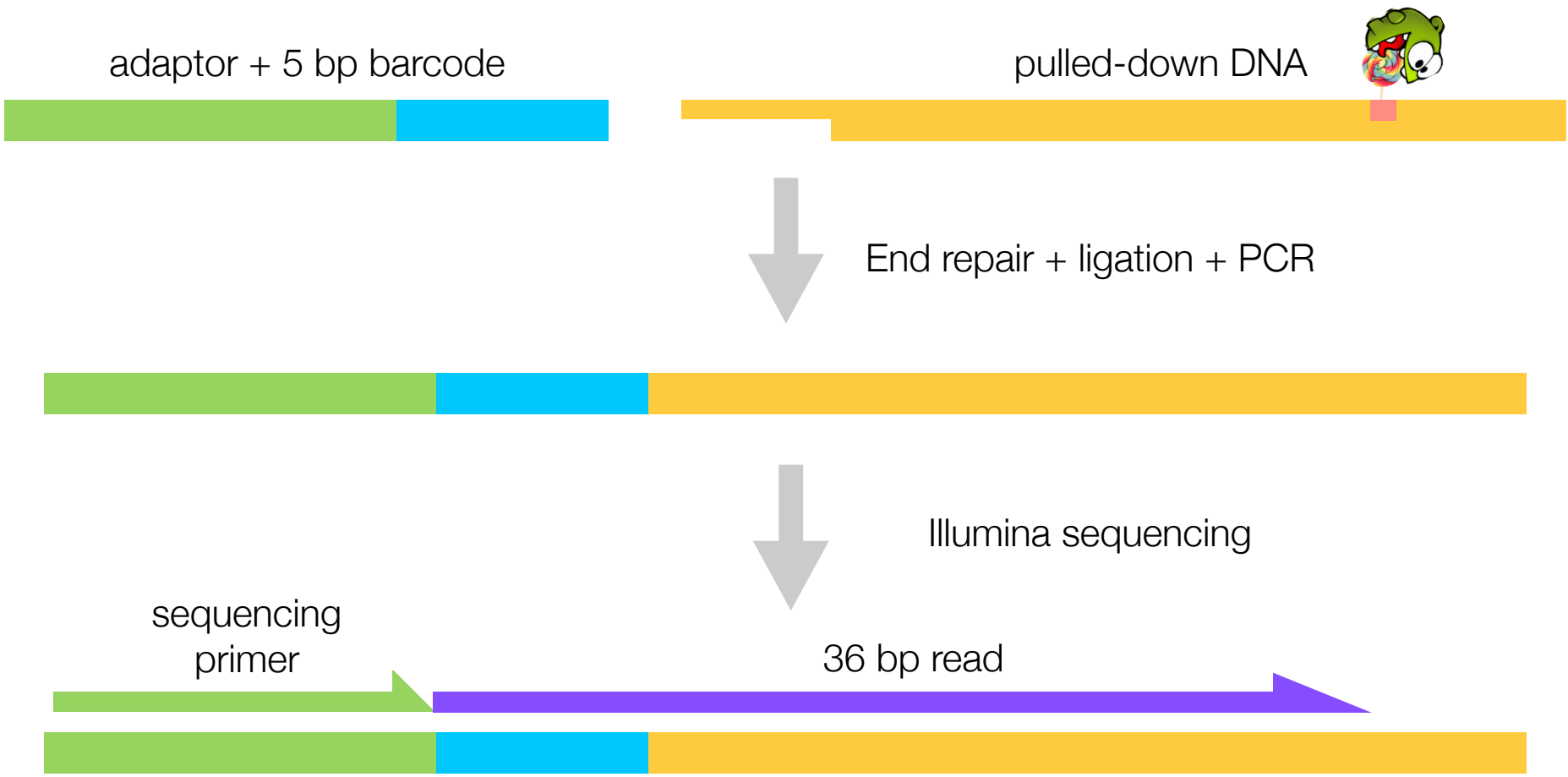


The Stillman lab is interested, in part, in the signaling mechanisms governing pre-RC firing -> genome-wide replication tracking

Tracking replication with EdU pulldown + sequencing



Barcoding samples for sequencing



~15 M reads for 14 barcoded samples

Thanks Yi-Jun!

What we will do

- Today
  - Map reads to the yeast genome
  - Compute “replication profiles”: # of reads covering each genomic position
  - View these data using the UCSC genome browser; compare to known ARSs
- Tomorrow
  - Matlab tutorial
  - Load replication profiles into Matlab
  - Smooth and plot replication profiles
- Homework: compare replication profiles for 3 different strains

Analysis Pipeline

- No single application available that will let us analyze these data
  - Just 4 steps to go from raw observations to biological discovery
- Each step requires selection, tuning, and debugging
  - Analogous to a wetlab protocol for running an experiment
- The components of the pipeline can be used in many other assays
  - Reads => Comparative Genomics, Transcriptome Analysis, de novo sequencing, Protein binding sites, Chromatin regulation…
  - Alignment => Forms the basis for almost every assay
  - SAMTools => Filtering, selection, interpretation of alignments

Analysis Pipeline

- Get the files (curl dash Capital-O)

```
$ curl –O http://schatzlab.cshl.edu/data/challenges/replication_exercise.tgz
```

- Unpack the files

```
$ tar xzvf replication_exercise.tgz
```

- Check out the files

```
$ cd replication_exercise/
$ ls -R
$ less *.txt
$ less reads/A1.fastq
```

[What is the secret phrase?]

Analysis Pipeline

- Check out the analysis script

```
$ cat course_pipeline.sh
```

- We have done already done the first steps to partition reads into batches

```
# Quality filter reads
# fastq_quality_filter -q 10 -p 90 -i /data/kinney/data/illumina_sequencing/
11.01.24_sheu_edu/reads.fastq -o reads/reads_qual.fastq

# Split reads by batch
# cat reads/reads_qual.fastq | fastx_barcode_splitter.pl --bcfile /data/
kinney/data/illumina_sequencing/11.01.24_sheu_edu/barcodes.txt --prefix reads/
tmp1_ --suffix .fastq --mismatches 0 –bol
```

- You can embed comments into scripts with '#'

Analysis Pipeline

- Now that the reads are prepared, next step is to align

```
# Create bwa index for genome
# bwa index genome/genome.fasta

# Align reads using bwa
bwa aln genome/genome.fasta reads/A1.fastq > mappings/A1.sai
bwa samse genome/genome.fasta mappings/A1.sai reads/A1.fastq > mappings/A1.sam
```

- BWA (Li & Durbin, 2009) is one of the most popular tools for aligning short reads to a reference genome. It is used in almost every sequencing assay that start from short reads. It takes a few steps to run because it uses a special index of the genome for making the alignments fast. We will talk about it in detail at the end of the course

Analysis Pipeline

- Now that the reads are aligned, need to transform and sort them

```
# Create pileup using samtools
samtools view -bS mappings/A1.sam > mappings/A1.bam
samtools sort mappings/A1.bam mappings/A1.sorted
samtools index mappings/A1.sorted.bam
samtools pileup -c -f genome/genome.fasta mappings/A1.sorted.bam > pileups/A1.pileup
```

- The pileup file encodes how many reads align to each position in the genome

```
$ less pileups/A1.pileup
```

- Run a quick command to find positions with deep coverage

```
$ awk '{if ($8>50){print}}' A1.pileup | less
```

[AWK is a really powerful, if arcane filter]

Analysis Pipeline

- Now run a custom script to summarize the depth information

```
$ ./pileup2bedfile.py pileups/A1.pileup 31
$ less pileups/A1.pileup.bed
```

- This file can then be loaded into the UCSC Genome Brower for inspection, and relate it to known annotations

See http://genome.ucsc.edu/

Homework

- Replication Analysis
  - Modify course_pipeline.sh to analyze B1, C1, D1
  - Load the bed files into the UCSC genome browser
  - See if you can spot and interesting variations between the data sets

- Read the Matlab Getting Started Guide. This is available as a pdf here:
  http://www.mathworks.com/help/pdf_doc/matlab/getstart.pdf

- Focus on these sections
  - Introduction
  - Matrices and Arrays
  - Graphics, starting with Basic Plotting Functions
  - Programming
  - Data Analysis
  - Desktop Tools and Development Environment

Resources

- Much like learning a new spoken language, computer languages have their own syntax and grammar that will be unfamiliar at first, but get easier and easier over time
  - There are many ways to accomplish the same task
  - You can quickly become a data magician

- The way to learn a new computer language is to practice speaking it
  - The ~30 commands you have seen today can be combined together into an infinite number of combinations
  - Lots of good resources available online:
    - http://www.molvis.indiana.edu/app_guide/unix_commands.html
    - http://tldp.org/LDP/abs/html/index.html
    - http://stackoverflow.com/
    - http://google.com

- WARNING: Computers can be very unforgiving
  - 'rm –rf /' <= delete every file on your computer
  - 'cp junk.doc thesis.doc' <= overwrite your thesis with junk.doc
  - 'cat results.partial > results.all' <= oops, should have appended with >>