Pontificia Universidad
Católica de Chile
Facultad de Ciencias
Biológicas

WELLCOME GENOME CAMPUS
CONNECTING SCIENCE
ADVANCED COURSES+ SCIENTIFIC CONFERENCES

wellcome connecting science

Next Generation Sequencing Bioinformatics, Santiago, Chile 23-27 January 2023

Project Descriptions
&
General instructions

1. Choose a project from the following project descriptions
2. Develop it in a team of 3 during the course of the week during the slots named Group Task Preparation.
3. Prepare a short 7 min power point presentation describing the results of your analyses for Friday 27th of January, afternoon session – Group Presentations.

PROJECT LIST:
Project 1. RNAseq. Analysis of the transcriptional landscape in a knockout parasite
Project 3. ChIP-Seq. Sequencing depth and Chromatin Environment
Project 4. Variant calling. Exploring variation data across human populations
Project 5. Structural variation. Calling structural variants in human genome data

Next Generation Sequencing Bioinformatics, Santiago, Chile 23-27 January 2023

Project 1 (XI)

GROUP 1

1.
2.
3.

GROUP 2

1.
2.
3.

Project 3 (AMe)

GROUP 1

1.
2.
3.

GROUP 2

1.
2.
3.

Project 4 (DR)

GROUP 1

1.
2.
3.

GROUP 2

1.
2.
3.

Project 5 (ED)

GROUP 1

1.
2.
3.

GROUP 2

1.
2.
3.

Project 1

RNAseq. Analysis of the transcriptional landscape in a knock out parasite

Plasmodium falciparum is the causative agent of the most dangerous form of malaria in humans. The reference genome for *P. falciparum* strain 3D7 was determined and published about 15 years ago (Gardener et al., 2002). Since then, the genomes of several other species of Plasmodium that infect humans or animals have been elucidated.

Malaria is widespread in tropical and subtropical regions, including parts of Asia, Africa, and the Americas. Each year, there are approximately 350–500 million cases of malaria killing more than one million people, the majority of whom are young children in sub-Saharan Africa.

As working within human is not feasible, mouse models are used like *Plasmodium berghei* or *Plasmodium chabaudi*. They can easily be maintained in the lab and mirror specific phenotypes as found in human malaria.

A colleague of yours knocked out a gene (**PBANKA_KO**) in the rodent malaria parasite ***Plasmodium berghei***. Over the last three months, she generated different biological replicates of the **wild type (WT)** and the **knock out (KO)** to understand differences between both lines.

This afternoon she has a meeting with her boss to show **FINAL** results for a paper submission, what the function of the gene to finalize a grant that needs submissions today.

Unfortunately, on the way to work, she broke both her hands cycling and cannot do the analysis!!!! Can you help her out?

Data:
/home/manager/course_data/group_projects/RNASeq

Your job is to understand the implication of the knock out of the gene.

- What influence does it have?
- How can you define that?
- Use your knowledge from the RNA-Seq module. Careful, some of the needed files might not be present...
- Useful sites:
    - www.plasmodb.org - GO enrichment – ask for help
    - www.genedb.org
    - ftp://ftp.sanger.ac.uk/pub/project/pathogens/gff3/CURRENT/ - for annotation – look for Pberghei*gff3*

Can you save your poor colleagues and save the grant....?

Project 3.

1 ChIP-Seq Project 1: Sequencing depth and Chromatin Environment

The following tasks have been adapted from materials developed by Angela Goncalves, Myrto Kostadima, Steven Wilder and Maria Xenophontos.

1.1 Sequencing depth

One of the most frequent questions that come up in ChIP-seq experiments is whether the sequencing depth is sufficient.

The more we sequence a ChIP-seq library, the more peaks of low fold change we will identify. Therefore, the only way to answer that question is to look for the number of peaks identified when we down sample our library. To test for sufficient sequencing depth in our sample we will down sample our ChIP and Control datasets to 10%, 20%, .., 90% of the initial library size and call peaks. To do so, we will use the functions **randsample** and **callpeak** from macs2, respectively.

First, go to the group_projects folder.

```
cd /home/manager/course_data/group_projects
```

Check to see if the ChIPSeq-Project1 folder exists.

```
ls ChIPSeq-Project1
```

If this folder doesn't exist, please check with your course instructor.

Once you have the data, go into the ChIPSeq-Project1 directory.

```
cd ChIPSeq-Project1
```

Below are a series of commands which will downsample the ChIP and Control datasets to 10%, 20%, .., 90% of the initial library size and call peaks. To do this we use a while loop which will start counting from 10 (`i=10`), in intervals of 10 (e.g. 10, 20, 30...) (`((i = i + 10))`) up to 100 (`$i -lt 100`). The value which is being counted is stored in a variable called `i` that is then referenced in other commands using `$i`. Finally, to make sure we know where we are in the loop, we will print a message in the terminal `echo "Looking at ${i}% of the reads"`.

The command we are using to downsample the ChIP and Control datasets reads is `macs2 randsample` which we will be giving an input file `-t`, a percentage `-p`, an output file `-o`, an output directory `--outdir` and the format of the input file `-f`.

Look at the usage for `macs2 randsample`.

```
macs2 randsample -h
```

We then call the peaks for each set of downsampled reads using `macs2 callpeak` in a similar way to the ChIP-Seq tutorial.

Type the commands below into a file called `project1.sh`.

```
i=10

       while [[ $i -lt 100 ]]
       do

       echo "Looking at ${i}% of the reads"

       macs2 randsample -t PAX5.bam -p $i -o PAX5.perc${i}.bed \
       -outdir macs2_downsample -f BAM

       macs2 randsample -t Control.bam -p $i -o Control.perc${i}.bed \
       -outdir macs2_downsample -f BAM

       macs2 callpeak -t macs2_downsample/PAX5.perc${i}.bed \
       -c macs2_downsample/Control.perc${i}.bed -gsize 138000000 \
       -format BED -name macs2_downsample/PAX5.perc${i} \
       -pvalue 1e-3 -call-summits

       ((i = i + 10))

       done
```

Now, make your script executable.

```
chmod u+x project1.sh
```

And run the script.

```
./project1.sh
```

Next, we want to plot the number of peaks called in each of the downsampled datasets. To do this we will use `R`. First we tell R where to find our output files using `setwd`. We then use a for loop to extract only the peaks whose overall enrichment meets a threshold value (`fc.thres <- 4` and `peaks <- peaks[peaks[, 7] > fc.thres, ]`). We then count those peaks (`no.peaks <- c(no.peaks, nrow(peaks))`). Finally, we plot the number of peaks vs the percentage, saving the output to `peaks_vs_percentage.jpg`.

Type the following commands into a file called `script.R`.

```
rm(list = ls())

       options(stringsAsFactors=F)
       setwd( "/home/manager/course_data/group_projects/ChIPSeq-Project1" )

       fc.thres <- 4

       no.peaks <- c()
       for(row in seq(from=10, to = 90, by = 10))
       {
               print(row)

               peaks <- read.table(paste("macs2_downsample/PAX5.perc",
           row, "_peaks.narrowPeak", sep=""))

               peaks <- peaks[peaks[, 7] > fc.thres, ]

               no.peaks <- c(no.peaks, nrow(peaks))
       }

       peaks <- read.table("PAX5_peaks.narrowPeak")
       peaks <- peaks[peaks[, 7] > fc.thres, ]
       no.peaks <- c(no.peaks, nrow(peaks))


       jpeg('peaks_vs_percentage.jpg')

       plot(seq(from=10, to = 100, by = 10), no.peaks,
       type="o", col="blue", xlab="Percentage of reads", ylab="Number of peaks"

       dev.off()
```

Use `Rscript` to run your R script.

```
Rscript script.R
```

Let's take a look at the plot.

```
eog peaks_vs_percentage.jpg
```

Q1: Do you think that we have sequenced enough?

Close the file when you have finished looking at the plot.

1.2 Chromatin Environment

The goal of this hands-on session is to investigate the chromatin environment around gene features and regulatory elements. We will be plotting where the different histone modifications occur in relation to genes. We will also look at how we can distinguish different genes based on transcription and chromatin environment.

1.2.1 Prepare environment

We will be using ENCODE GM12878 histone modifications. The BAM alignment files have been downloaded from http://www.encodeproject.org. All ENCODE experiments had at least 2 technical replicates, but we will only be using the first replicate for simplicity.

We will be using **ngsplot** https://github.com/shenlab-sinai/ngsplot to visualise these datasets at transcription start sites. The ngsplot database for the human gene set has been generated from the Ensembl http://www.ensembl.org and RefSeq http://www.ncbi.nlm.nih.gov/refseq/ gene sets, by default using the Ensembl set.

Make sure that the location where we want to write the database to is writable.

```
sudo chmod -R 777 /usr/local/bioinf-recipes/ngsplot-2.63
```

When prompted, type the password for manager which is `manager`.

Next, make sure that the following R libraries are installed.

```
sudo R
```

In R, type the following commands. When prompted, type `n` so that other packages are not updated.

```
source("https://bioconductor.org/biocLite.R")
BiocInstaller::biocLite(c("ShortRead", "BSgenome", "doMC"))
```

To quit R type `q()` and enter `n` when prompted.

Now, install the database using `ngsplotdb.py`.

```
echo 'Y' | ngsplotdb.py install ngsplotdb_hg19_75_3.00.tar.gz
```

Look at the usage for `ngs.plot.r`.

```
ngs.plot.r
```

Further help on the ngsplot options can be found at https://github.com/shenlab-sinai/ngsplot/wiki/ProgramArguments101.

The histone modifications we will be looking at today are all thought to play different roles in the regulation of gene expression and the putative functions, according to the ENCODE Project, are summarised in the table. We will look at them individually in this exercise, but later will be integrating multiple histone modifications.

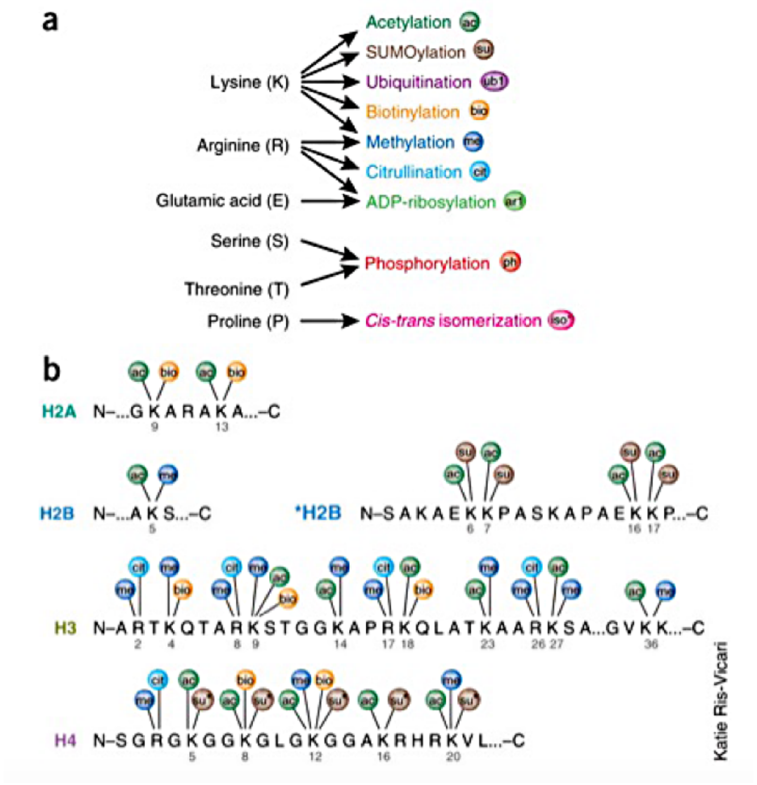
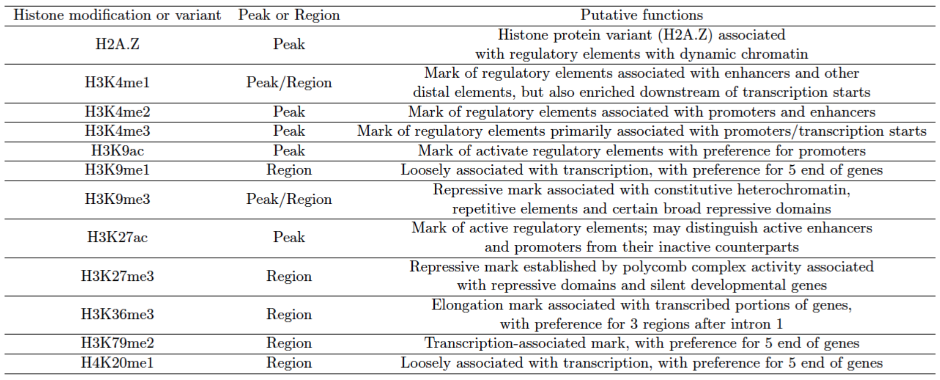


figure 1

Figure 1: The histone modifications are named by the histone tail, location in the protein sequence and the biochemical modification, e.g. H3K4me3, refers to the trimethylation of the lysine (K) in the fourth position in the protein sequence of histone 3 tail.



Histone modification or variant	Peak or Region	Putative functions
H2A.Z	Peak	Histone protein variant (H2A.Z) associated with regulatory elements with dynamic chromatin
H3K4me1	Peak/Region	Mark of regulatory elements associated with enhancers and other distal elements, but also enriched downstream of transcription starts
H3K4me2	Peak	Mark of regulatory elements associated with promoters and enhancers
H3K4me3	Peak	Mark of regulatory elements primarily associated with promoters/transcription starts
H3K9ac	Peak	Mark of activate regulatory elements with preference for promoters
H3K9me1	Region	Loosely associated with transcription, with preference for 5 end of genes
H3K9me3	Peak/Region	Repressive mark associated with constitutive heterochromatin, repetitive elements and certain broad repressive domains
H3K27ac	Peak	Mark of active regulatory elements; may distinguish active enhancers and promoters from their inactive counterparts
H3K27me3	Region	Repressive mark established by polycomb complex activity associated with repressive domains and silent developmental genes
H3K36me3	Region	Elongation mark associated with transcribed portions of genes, with preference for 3 regions after intron 1
H3K79me2	Region	Transcription-associated mark, with preference for 5 end of genes
H4K20me1	Region	Loosely associated with transcription, with preference for 5 end of genes

figure 2

Figure 2: A summary of the putative functions of various histone marks, according to the ENCODE Project.

1.2.2 Transcription Start Sites

It has been shown that the chromatin state, i.e. the combination of all regulatory variations at the chromatin level, in the neighbourhood of a gene has a large effect on its transcription. We will therefore start by looking at the genome-wide profiles of the selected histone marks around TSS.

Plot the distribution of H3K4me1 around protein-coding gene TSS (this may take a few minutes to run).

```
ngs.plot.r -G hg19 -O H3k4me1.tss -FL 150 -R cgi \
       -C H3k4me1.bam -T H3k4me1
```

Q2: You can now view the H3K4me1 aggregation plots and heatmaps in your folder. Where does the majority of the H3K4me1 signal appear in relation to TSS?

Q3: What proportion of genes display this pattern?

Repeat these plots for other histone modifications.

```
ngs.plot.r -G hg19 -O H3k4me3.tss -FL 150 -R tss \
       -C H3k4me3.bam -T H3k4me3
```

ngsplot also enables you to correct the ChIP sample using the control sample. This aims to correct for any genome biases in alignability, GC content etc.

Let's give this a try.

```
ngs.plot.r -G hg19 -O H3k4me3_Control.tss -FL 150 \
       -R tss -C H3k4me3.bam:Control.bam -T H3k4me3:Control
```

Note: If you get an error about the alignment not being sorted, try sorting your BAM file and using the sorted file instead.

You can now also plot the signal aroung specific annotated regions, such as Ensembl gene bodies. H3K36me3 is known to be enriched over transcribed genes.

Let's give this a try.

```
ngs.plot.r -G hg19 -O H3k36me3 -FL 150 -R genebody \
       -C H3k36me3.bam -T H3k36me3
```

Q4: Can you detect any patterns in this enrichment?

You can use configuration files in ngsplot to draw multiple graphs on the same plot. For instance, we will plot multiple histone modifications on the same plot, and also subset the genes, based on gene expression.

Using a text editor, create the following tab-separated file, called `multhist.txt` for drawing two histones modifications on the same plot. -1 corresponds to looking at the whole genome.

```
|-------------|----|---------|
| H3k4me1.bam | -1 | H3k4me1 |
| H3k4me3.bam | -1 | H3k4me3 |
```

Run the following command to plot the graphs:

```
ngs.plot.r -G hg19 -O H3k4me1.H3k4me3 -FL 150 -R tss \
        -C multhist.txt -T H3k4me1.H3k4me3
```

Q5: Are the patterns you observe consistent with the putative functions of the modification in the table given above?

1.2.3 Gene Expression

You will now use RNA-seq data form the same cell type results to separate all of the Ensembl genes into three categories, based on the expression (FPKM values) of the genes. The file containing the FPKMs for GM12878 whole cell RNA-seq is called `genes.fpkm_tracking`. These FPKM values have been computed by Cufflinks.

Read the first few lines of the GM12878 whole cell genes.fpkm_tracking file.

```
head -n 10 genes.fpkm_tracking
```

Remove the header line.

```
sed '1d' genes.fpkm_tracking > genes.fpkm.txt
```

Sort the genes by FPKM score.

```
sort -k10 -n -r genes.fpkm.txt > genes.fpkm.sorted.txt
```

Count the number of genes.

```
wc -l genes.fpkm.sorted.txt
```

Use `head`, `sed` and `awk` to pull out the Highest, Lowest and Intermediate expression genes.

```
head -n 5000 genes.fpkm.sorted.txt | \
        awk '{print $1}' > high_expressed_genes.txt
```

```
awk '{ if($10 ==0) {print $1} }' \
        genes.fpkm.sorted.txt > low_expressed_genes.txt
```

```
sed -n '10001,15000p' genes.fpkm.sorted.txt | \
       awk '{print $1}' > mid_expressed_genes.txt
```

Create a tab-delimited configuration file expression.txt in a text editor containing the following lines.

```
|--------------|--------------------------|--------|
| H3k27me3.bam | high_expressed_genes.txt | "High" |
| H3k27me3.bam | mid_expressed_genes.txt  | "Med"  |
| H3k27me3.bam | low_expressed_genes.txt  | "Low"  |
```

Then run the following command.

```
ngs.plot.r -G hg19 -R genebody -C expression.txt \
       -O H3k27me3.express.genebody -D ensembl \
       -FL 300 -T H3k27me3.expression
```

Q6: What patterns can you see?

If you made it to here, well done!

Project 4. Variant calling

Bioinformatics Summer School: Exploring variation data across human populations

Scenario

Natural variation between individuals within a species is required to generate the broad range of traits and phenotypes that exist between single individuals and between different populations spread around the globe. Over time some mutations may become fixed or found at increasing frequencies in one population and not another.

The rapid adoption of next generation sequencing technology has quickly enabled researchers to generate and interrogate sequence data from multiple individuals across populations, or among individuals with particular traits/phenotypes. In particular, finding and understanding genetic differences among individuals is paramount to accurately assess the role that variation plays in human biology and disease.

You are interested in genomic variation between two separate human populations of European and Asian descent. You have sequenced a number of different individuals from each population and in the first section of this project, you wish to find variants that exist between them. In the second section, you will use a range of bioinformatics tools to analyse the functional consequences of the variants you have identified, linking them to phenotypes.

Dataset

All of the data for this project is available in a compressed tar archive (.tar.gz). You will need to download and unpack the project data to carry out the analysis described above. Once downloaded, you should have paired-end fastq files for 8 samples obtained from the 1000 genomes project and a reference genome (fasta format).

The data can be obtained from here: http://bit.ly/2r97OWf

Project aims

- Align the fastq files to human reference to produce BAM files.
- Perform variant calling of the aligned BAM files to produce a VCF file.
- Perform quality filtering of VCF to obtain a list of high quality candidate variants.
- Use a variety of online tools (VEP, Ensembl, UniProt etc.) to find the functional consequences of the variants.

.

Project 5. Structural variation

Title: Calling structural variants in human genome data

Background
Structural variants (SVs) are large (usually defined as >50 basepairs) differences within a genome. While structural variants occur much less frequently than SNVs and indels[1], they are important contributors to human phenotypes.

In this exercise, we will call structural variants in short reads from HG002 Genome in a Bottle individual. We will use svaba[2] to call variants, then compare these variants to calls from Manta[3] run on the same sample and a truth set from Genome in a Bottle[4].

Data
Genome in a Bottle is an initiative from the US National Institutes for Standards and Technology to provide standardized genomes and variant calls for scientific analysis. HG002 is the child of an Ashkenazi Jewish trio; this individual has been sequenced on many sequencing platforms and their genome has been extensively characterized.

Reads from HG002 have already aligned to hg38 using BWA. The BAM file is available here:
https://www.dropbox.com/s/ai4cvt5c3dtut2w/hg0002.grch37.22.bam
and the .bai index file is here:
https://www.dropbox.com/s/a8t4mb2jexm582u/hg0002.grch37.22.bam.bai

A reference FASTA file for chromosome 22 (GRCh37 / hg19) is provided here:
https://www.dropbox.com/s/pyxhf1p2zolm7ot/22.fa

Task

1. Create a directory called “sv_project” to work in, and cd into it. Hint: remember the “mkdir” command?

2. Download and install prerequisites
    a. To install svaba and BWA, you’ll need to install the GNU C compiler (GCC) as well as several libraries that are used to compress and decompress BAM files. You can do that in your VM with the following commands
        i. sudo apt-get update
            1. When prompted for a password, type “manager”; type “Y” if asked for permission to continue.
        ii. sudo apt-get install gcc g++

[1] Sudmant et al. 2015, An integrated map of structural variation in 2,504 human genomes. https://www.nature.com/articles/nature15394
[2] Wala et al. 2018, SvABA: genome-wide detection of structural variants and indels by local assembly. https://genome.cshlp.org/content/early/2018/03/13/gr.221028.117
[3] Chen et al. 2016, Manta: rapid detection of structural variants and indels for germline and cancer sequencing applications. https://academic.oup.com/bioinformatics/article/32/8/1220/1743909
[4] Chapman et al 2020, A crowdsourced set of curated structural variants for the human genome. https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1007933

        1. When prompted for a password, type “manager”; type “Y” if asked for permission to continue.

    iii. sudo apt-get install zlib1g-dev liblzma-dev liblz4-dev libbz2-dev

3. Next, we’ll build svaba
    i. Svaba is available on github at https://github.com/walaj/svaba . Use git to clone the repo (hint: try “git clone” followed by the github link)
    ii. Change into the newly created “svaba” directory and see what’s in it.
    iii. We need to run a program called autoconf to find all the dependencies we just installed. Run it by typing “./configure”
    iv. We can now build svaba using Make. Type “make” to build svaba from source.
        1. Pro tip: if you want it to build faster, run “make -j” to run the build in parallel.
    v. Lastly, type “make install” to install svaba into the “bin” directory.
    vi. Test it out! Running “./bin/svaba” should show you the usage of the program.

4. Download and install BWA
    a. Go back up to your sv_project directory and use git clone again to download the BWA source code from https://github.com/lh3/bwa
    b. CD into the bwa directory. BWA is easy to build - just run “make”

5. Download project data
    a. We’ll use wget to grab the remote data files. Just run “wget https://www.dropbox.com/s/18m3wvlhvtbap67/hg0002.novaseq.pcr-free.35x.chr22.rg.bam” to get the BAM.
    b. Download the .BAI file and the fasta reference listed in the Data section as well.

At this point, you should have a directory with the BAM, the BAI, and the reference genome in it. Now we’ll get started on analyzing our structural variants (after we index our reference genome).

6. Create a FASTA index and the BWA index files for the reference genome
    a. Use “samtools faidx” to create a FASTA index.
    b. Use “bwa index” to create the BWA index files. These are used by svaba to perform targeted realignment. **Remember that you will need to use the full path to bwa** (e.g., if you’re in your sv_project directory, you would run “./bwa/bwa index 22.fa”). This should take about 1 minute to complete.

7. Run svaba to detect genomic SVs
    a. We need to run svaba in germline mode. Details of how to do so are here: https://github.com/walaj/svaba#whole-genome-germline-sv-and-indel-detection. Substitute the right file names for GERMLINE_BAM and REF and run using the

number of cores assigned to your VM (likely 2). Again, remember that you'll need to use the relative path to svaba (e.g., from your sv_project directory, ./svaba/bin/svaba).

8. Load the BAM file and svaba SV VCF into IGV.
    a. You should see the reads as well as a track corresponding to the svaba VCF.
        i. Look at some of the SVs by zooming in (the zoom slider is in the top right of the IGV window). Do you see any with evidence (e.g., discordant reads or altered depth)?
        ii. Are any SVs in genes? If so which ones? If not, why do you think this is the case?

9. Compare the VCF you made using svaba to the composite callset from Genome in a Bottle (HG002_SVs_Tier1_v0.6.vcf.gz) and a callset from Manta.
    a. The GIAB SV calls can be downloaded from here: https://www.dropbox.com/s/3mlyaxaaydt7xvc/HG002_SVs_Tier1_v0.6.vcf.gz
    Manta calls are available here (you'll want both the vcf.gz and the vcf.gz.tbi): https://www.dropbox.com/s/vmkx7ltd6n9whpk/manta.hg0002.22.diploidSV.vcf.gz
    and
    https://www.dropbox.com/s/3k0mq9qyde40x3z/manta.hg0002.22.diploidSV.vcf.gz.tbi
    b. You might want to use IGV or bedtools. Remember, you might need to unzip the VCF file(s) first.
    c. Also, examine the SV calls from SVABA. You should see multiple different values for the EVDNC info tag - what do you think they mean?
    d. Find a high quality SV (or a few) - these will have the strongest combined evidence (ASDIS), a high discordant read MAPQ, and a high alt read MAPQ. Record some of these, look at them in IGV, and see if you can verify and describe them. Take some screenshots and note your findings so you can share them with the class!

Setup:
You may want to enable clipboard sharing to make copy-pasting links easier. To do so, click the devices->shared_clipboard->bidirectional setting in the VirtualBox menu bar.

Instructor: Eric Dawson