GULBENKIAN CIÊNCIA

Variant detection: hands on

Precision Oncology Course

Coral Fustero Torre
Bioinformatics Unit,
Structural Biology Programme.
cfustero@cnio.es |
bioinformatics.cnio.es

cnio SPANISH NATIONAL CANCER RESEARCH CENTRE

BU CNIO **BIOINFORMATICS** UNIT

Objectives: Exome analysis (OVCA)

1. Understand how to configure varca
2. Run the varca pipeline
3. Check out the variant calling results
    a. Small mutations: SNV and INDELS
    b. Germline variants
    c. Somatic variants
4. Visualize the variants using IGV

Introduction

1) Understanding the data
2) Varca overview
3) Snakemake

Overview: Exome analysis (OVCA)

Patient suffering ovarian cancer

Whole-exome sequencing data from two samples from the patient:

- Tumour sample.
- Matched normal sample (healthy tissue) from epithelium.

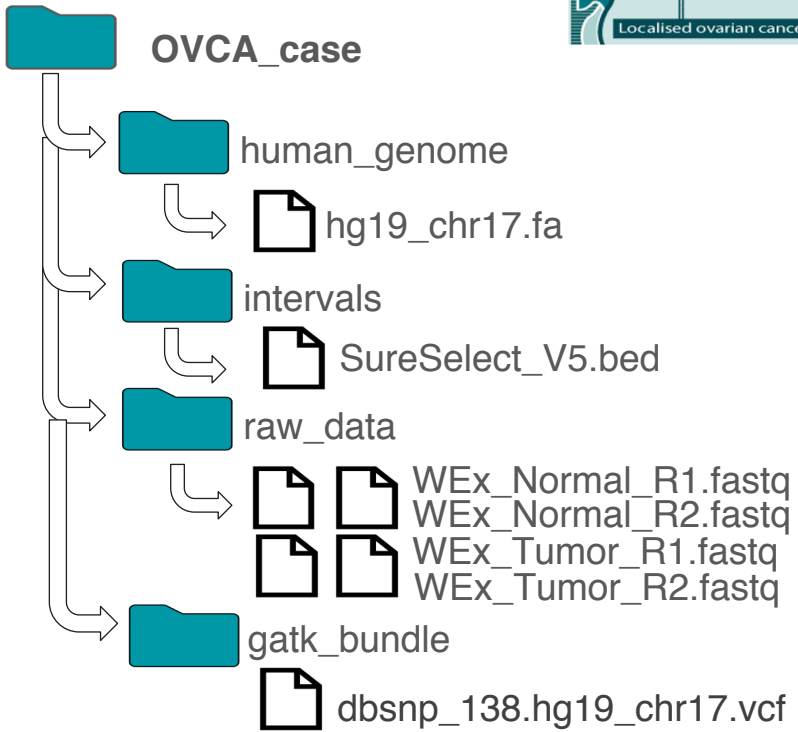
Library protocol: Agilent SureSelect V5 Human All Exons.

Sequencing platform: HiSeq 2000 (Illumina)

Expected outcome:

- ~dozens of germ-line variants.
- A few somatic cancer mutations (SNV, indel).

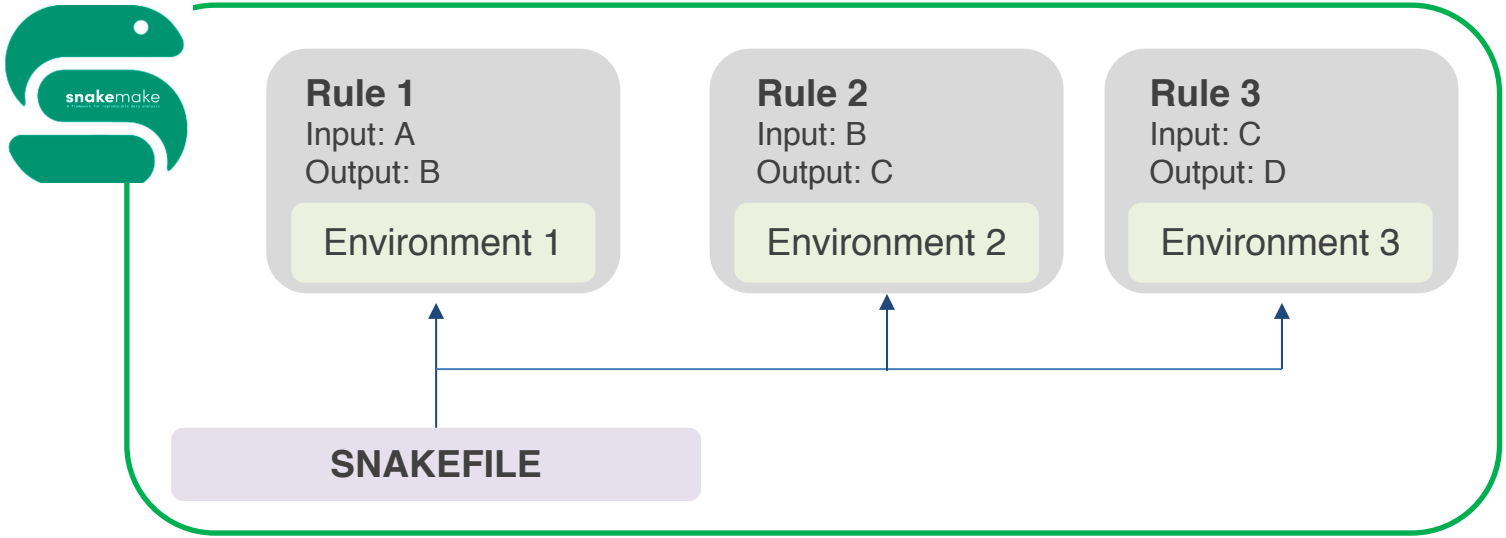
NOTE: This data was simulated and reduced in order to perform the computational analysis in class time.



Download the data:
https://drive.google.com/drive/folders/1oe007Qh1LTKoCGHLGE9XKrxG8_BL8GsL?usp=sharing

Overview: Snakemake

Snakemake is a workflow management system. **In general terms,** it's formed by:
1) Rules - actions to be applied to an input file
2) Environments - packages needed in order to run a rule
3) Snakefile - record of the inputs and outputs from all steps in our analysis as well as the dependencies between files



Road map

QUALITY CONTROL	Quality Control of raw data		FastQC
TRIMMING	Trimming of fastq reads		Trimmomatic
	Tumor sample	Normal sample	
ALIGNMENT	Indexing of Reference Genome		BWA-index
	Sample alignment	Sample alignment	BWA-MEM
REFINEMENT OF ALIGNMENT	Mark duplicates	Mark duplicates	Picard
	Base quality recalibration	Base quality recalibration	GATK

varc Road map

	Tumor sample	Normal sample	
VARIANT CALLING	Tumor Genomic positions	Normal Genomic positions	Germinal: Haplotypecaller Somatic: MuTect2
FILTERING	Tumor Calling & Filtering	Normal Calling & Filtering	Germinal: VQRS/Hardfilter Somatic: MuTect2
ANNOTATION	Multisample file annotation		Germinal: SnpEff Both: VEP
VISUALIZATION	Visualization of variants in alignments		IGV
STATISTICS	Quality control and Alignment stats		MultiQC

Getting started

1) Make a local copy of the data
2) Set a conda environment
3) Download varca

Getting started: 1) Make a local copy of the data

- Download the data into your home
- Take a look at the data for the exercise

```
$ cd /home/$USER/OVCA_case
$ ls -l *
```

human_genome:
hg19_chr17.fa

intervals:
SureSelect_V5_human_all_Exons_chr17.bed

raw_data:
WEx_Normal_R1.fastq
WEx_Normal_R2.fastq
WEx_Tumour_R1.fastq
WEx_Tumour_R2.fastq

gatk_bundle
dbsnp_138.hg19_chr17.vcf

Human reference genome.
Alignment on a consensus genome reference.
Source: UCSC / 1000G project

WEx Library design.
Predefined genomic regions of interest.
Source: manufacturer
Useful for : visualize the alignment

Raw exome sequencing data.
Patient's sample data, generated by the sequencing machine.
Source: Collaborator / Consumer provider

dbsnp annotation file
Source: dbsnp

Getting started: 2) Conda installation

- Go to https://docs.conda.io/en/latest/miniconda.html#linux-installers
- Download the miniconda bundle (choose the optimal version for your computer)

```
$ wget https://repo.anaconda.com/miniconda/Miniconda3-py39_4.10.3-Linux-x86_64.sh
```

- In the directory where the file Miniconda3-py39_4.10.3-Linux-x86_64.sh

```
$ bash Miniconda3-py39_4.10.3-Linux-x86_64.sh
```

- Once the installation is over, restart the terminal.

Visit https://anaconda.org/ to find packages of interest

Getting started: 2.1) Anatomy of a conda command

this means you are in the base environment

```
(base)$ conda create -c CHANNEL_NAME -n ENV_NAME optional_packages
```

Action | Channels you want to add (if any) | Name of your new environment | Packages you want to install

```
(base)$ conda activate ENV_NAME
```

Action | Name the environment you want to activate

```
(base)$ conda install -c CHANNEL_NAME PACKAGE_NAME
```

Action | Channel where to find the package of interest | Package you want to install

Getting started: 2.2) Snakemake installation

- Install snakemake through mamba:

```
(base)$ conda install -c conda-forge mamba
(base)$ mamba create -c conda-forge -c bioconda -n snakemake snakemake
(base)$ conda activate snakemake
```

Mamba

https://github.com/mamba-org/mamba

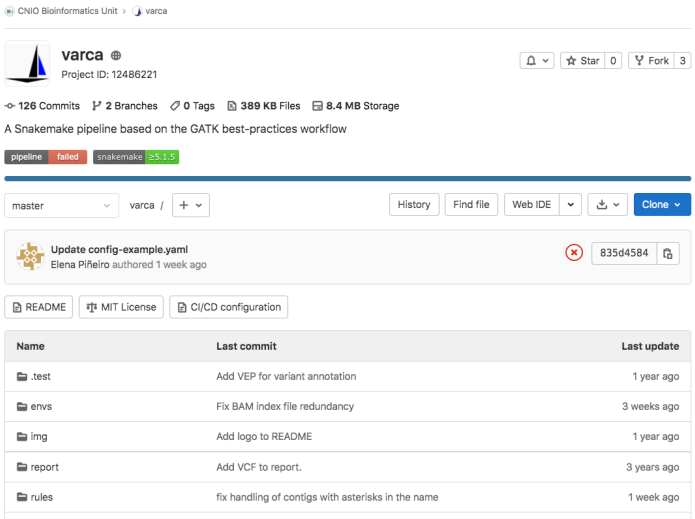
At the same time, mamba utilizes the same command line parser, package installation and deinstallation code and transaction verification routines as conda to stay as compatible as possible.

Mamba is a reimplementation of the conda package manager in C++. Allows:

- Parallel downloading of repository data and package files using multi-threading
- libsolv for much faster dependency solving, a state of the art library used in the RPM package manager of Red Hat, Fedora and OpenSUSE
- Core parts of mamba are implemented in C++ for maximum efficiency

Getting started: 3) Downloading Varca

https://gitlab.com/bu_cnio/varca




```
(snakemake)$ mkdir varca
(snakemake)$ cd varca
(snakemake)$ git clone
https://gitlab.com/bu_cnio/varca.git .
Cloning into '.'...
remote: Enumerating objects: 853, done.
remote: Counting objects: 100% (128/128),
done.
remote: Compressing objects: 100% (74/74),
done.
remote: Total 853 (delta 76), reused 103
(delta 53), pack-reused 725
Receiving objects: 100% (853/853), 1.47
MiB | 0 bytes/s, done.
Resolving deltas: 100% (545/545), done.
Checking connectivity... done.
```

Configuring varca

1) contigs.tsv
2) samples.tsv
3) units.tsv
4) config.yaml

Configuring varca: contigs file

contigs-example.tsv
It contains the contigs of the reference genome to include in the analysis (one contig per line). If empty, the analysis is performed using all contigs of the fasta index.
In our case, we just need chromosome 17

- Open the file using a text editor
- Erase all the chromosomes except **chr17**
- Save all changes
- Rename the file from contigs-example.tsv to contigs.tsv

Configuring varca: samples file

samples-example.tsv

Lists all samples to be included in the run, the use of MuTect2 and its execution mode.

HaplotypeCaller will be executed for each sample in the sample column. To activate the execution of MuTect2 and set its execution mode, the control column is used.

If control contains:

- - MuTect2 is not executed for that sample
- The same sample name as in the sample column: MuTect2 is executed for that sample in tumor-only mode.
- A different sample name than in the sample column: MuTect2 is executed in tumor-normal mode; being the tumor sample the one indicated in the sample column, and the normal sample the one indicated in the control column. This way we will detect somatic mutations

sample	control
tumor	-

sample	control
tumor	tumor

sample	control
tumor	control
control	-

Configuring varca: samples file

samples-example.tsv

- Open the file
- Modify it so we can obtain both somatic and germline mutations

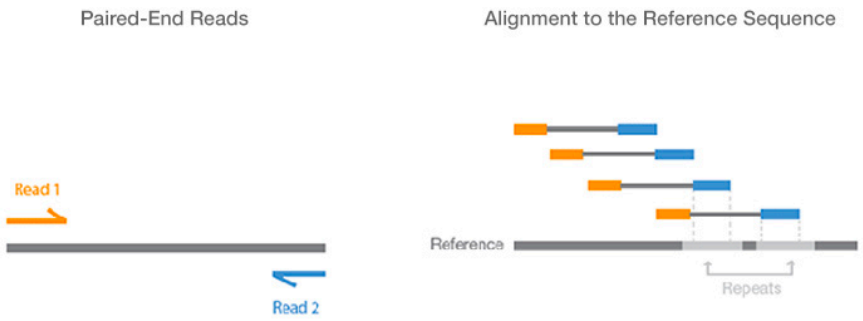
sample	control
tumor	control
control	-

- Save all changes
- Rename the file from samples-example.tsv to samples.tsv

Configuring varca: units file

units-example.tsv
It contains specifications of the samples (sequencing units, sequencing platform and fastq files) listed in samples.tsv.

In our case, we define tumor and control samples with two files for each, **read 1** and **read 2**




```
sample   unit     platform fq1      fq2
tumor    1        ILLUMINA path_to_WEx_Tumour_R1.fastq path_to_Wex_Tumour_R2.fastq
control  1        ILLUMINA path_to_WEx_Normal_R1.fastq path_to_Wex_Normal_R2.fastq
```

- Modify the file
- Rename it from units-example.tsv to units.tsv

Configuring varca: config file

config-example.yaml
This file contains the paths to files required in the analysis, the enabling/disabling of optional steps, as well as additional parameters for some of the programs used in the process. The file is structured in several sections to be customized according to the characteristics of the analysis. The indications for its filling are provided inside the file.

- Open and modify the file (see the instructions in the following slides)
- Rename it from config-example.tsv to config.tsv

Configuring varca: config file

config.yaml: main parameters

```yaml
samples: samples.tsv
units: units.tsv
contigs: contigs.tsv

outdir: "out"
logdir: "log"

ref:
# Genome database of snpeff to be used in the annotation with this resource.
Available databases can be checked with java -jar snpEff.jar databases
name: GRCh37.75
# Path to the reference genome, ideally as it is provided by the GATK bundle.
genome: /path_to_/hg19_chr17.fa
# Path to any database of known variants, ideally as it is provided by the
GATK bundle.
known-variants: /path_to/dbsnp_138.hg19_chr17.vcf
```

Configuring varca: config file

config.yaml: filtering parameters

```
filtering:
# Set to true in order to apply machine learning based recalibration of
# quality scores instead of hard filtering.
vqsr: false
hard:
# hard filtering as outlined in GATK docs
# (https://gatkforums.broadinstitute.org/gatk/discussion/2806/howto-apply-
hard-filters-to-a-call-set)
snvs:
"QD < 2.0 || QUAL < 100.0 || DP < 50.0 || SOR > 3.0 || FS > 60.0 || MQ < 40.0
|| MQRankSum < -12.5 || ReadPosRankSum < -8.0"
indels:
"QD < 2.0 || QUAL < 100 || DP < 50.0 || FS > 200.0 || ReadPosRankSum < -20.0"
#depth of coverage threshold to apply to variants identified with MuTect2
depth: "DP < 50"
```

Configuring varca: config file

config.yaml: filtering parameters

```
processing:
remove-duplicates: true
# Uncomment and point to a bed file with, e.g., captured regions if necessary,
# see https://gatkforums.broadinstitute.org/gatk/discussion/4133/when-should-
i-use-l-to-pass-in-a-list-of-intervals.
restrict-regions: /path_to/SureSelect_V5_human_all_Exons_chr17.bed
# If regions are restricted, uncomment this to enlarge them by the given value
in order to include
# flanking areas.
# region-padding: 100
```

Running varca

1) Dry run
2) Pipeline execution

Running varca: dry run

Activate snakemake environment (in case you haven't done it yet)

```
(base) $ conda activate snakemake
```

Test dry run: this will tell you if you're good to go!

```
(snakemake) $ snakemake --use-conda -n
```

```
Building DAG of jobs...
Conda environment envs/stats.yaml will be created.
Conda environment https://github.com/snakemake/snakemake-wrappers/raw/0.77.0/bio/samtools/stats/environment.yaml will be created.
Conda environment https://github.com/snakemake/snakemake-wrappers/raw/0.77.0/bio/multiqc/environment.yaml will be created.
Conda environment https://github.com/snakemake/snakemake-wrappers/raw/0.77.0/bio/snpeff/download/environment.yaml will be created.
Conda environment https://github.com/snakemake/snakemake-wrappers/raw/0.77.0/bio/snpeff/annotate/environment.yaml will be created.
Conda environment envs/rbt.yaml will be created.
Job stats:
job                count    min threads    max threads
--------------  -------  -------------  -------------
all                    1              1              1
multiqc                1              1              1
plot_stats             1              1              1
samtools_stats         2              1              1
snpeff                 1              1              1
snpeff_download        1              1              1
vcf_to_tsv             1              1              1
total                  8              1              1


[Thu Oct  7 21:32:06 2021]
rule snpeff_download:
    output: resources/snpeff/GRCh37.75
```

Running varca: executing the pipeline

In this test, we are just going to run the pipeline until the mutect filtered results (no VEP annotation)
For this purpose, first run:

```
(snakemake) $ snakemake --use-conda --cores 2 --until merge_calls
```

Once it finishes, run the following command to obtain the somatic variants using MuTect2:

```
(snakemake) $ snakemake --use-conda --cores 2 --until filter_mutect_2
```

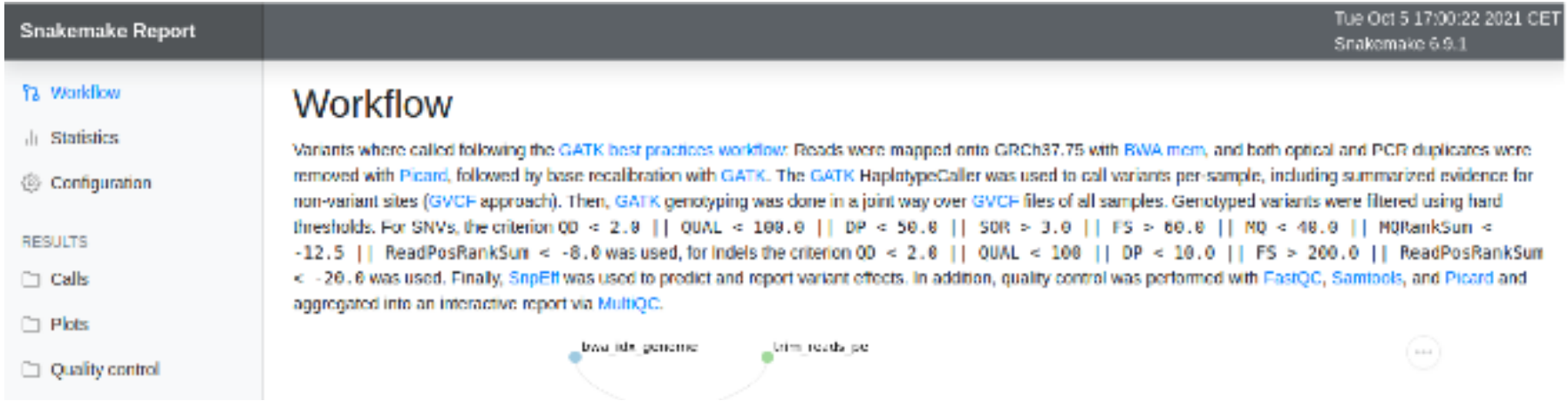
Running varca: full execution

In case you wanted to run the whole pipeline:

```
(snakemake) $ snakemake --use-conda --cores 2
```

Once finished, generate report (this step only works if **all steps** have been previously run)

```
$ snakemake --report report.html
```



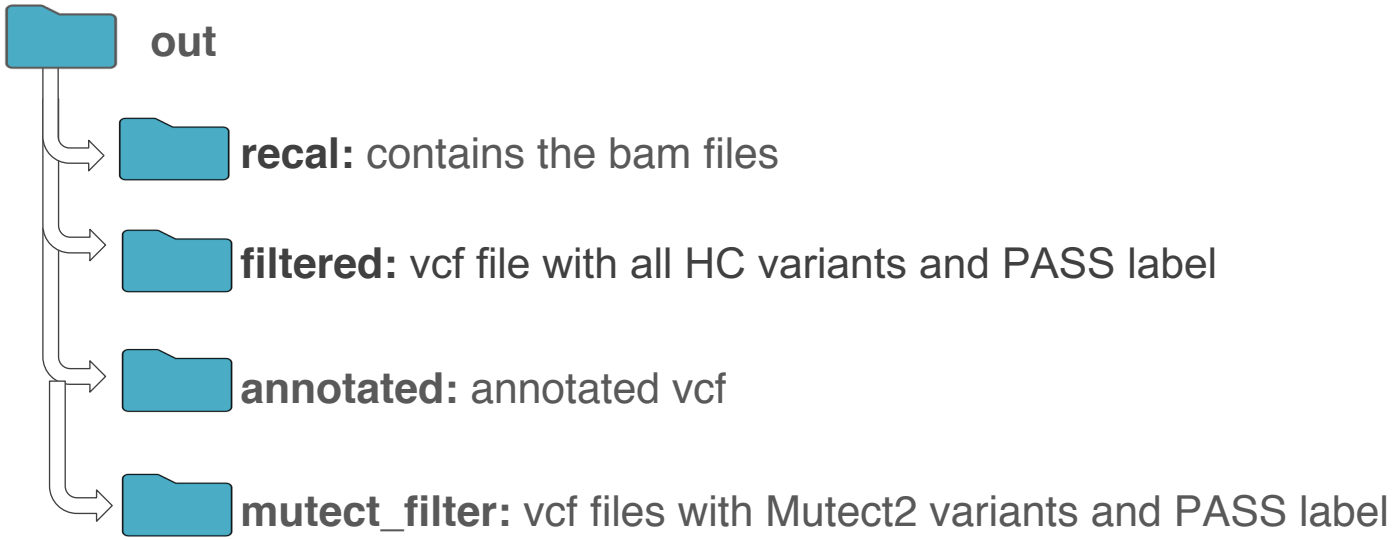
Note: if you want to run the complete pipeline, make sure you complete the **config.yaml** file properly.

Understanding the results

1) Where to find the main results
2) Results visualization

Understanding the results: Where to find them

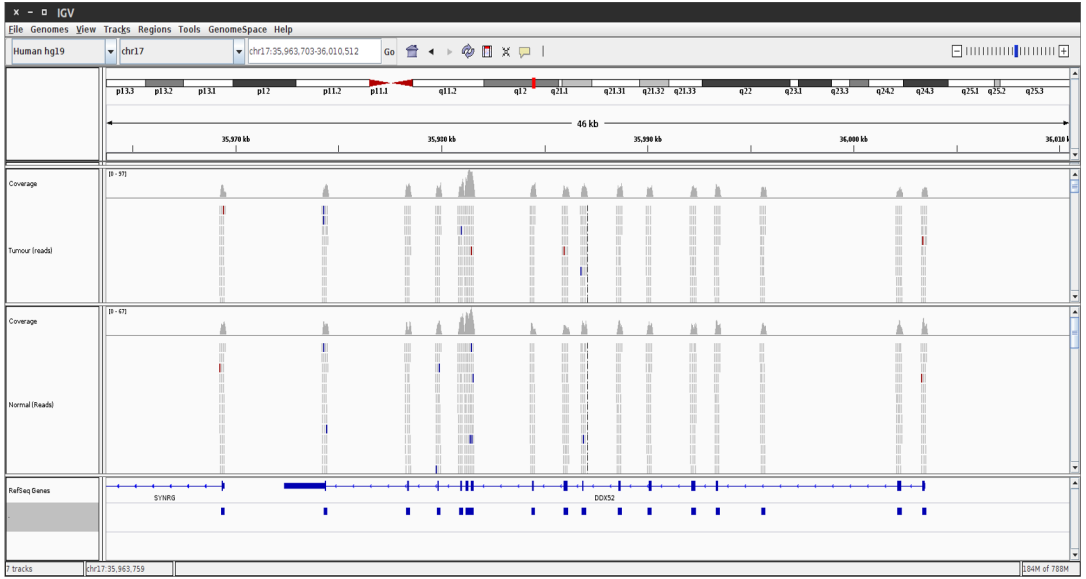
NOTE: These are just the main results. Check out varca's documentation for more.



Understanding the results: Visualization

1. Open a terminal, and execute the command:

```
(snakemake) $ conda install -c bioconda igv
(snakemake) $ igv
```



2. Open the BAM files for each sample

- tumor-1.bam
- normal-1.bam

3. Open the BED file (intervals file)

- SureSelect_V5.bed

Manual: http://software.broadinstitute.org/software/igv/UserGuide

Integrative Genomics Viewer

Can you answer the following questions?

Results

Germline variants

There were detected ___ germline variants in total:

- ___ Single Nucleotide Variants.
- ___ Indels.

Somatic variants

There were detected ___ somatic SNVs.

Genes affected and type of mutations (see alignment using IGV on chr17):

- _____.
- _____.

Integrative Genomics Viewer

TIP: Consider only those variants with the PASS label

Remember

Extra: VCF format

Header

```
##fileformat=VCFv4.2
##fileDate=20090805
##source=myImputationProgramV3.1
##reference=file:///seq/references/1000GenomesPilot-NCBI36.fasta
##contig=<ID=20,length=62435964,assembly=B36,md5=f126cdf8a6e0c7f379d618ff66beb2da,species="Homo sapiens",taxonomy=x>
##phasing=partial
##INFO=<ID=NS,Number=1,Type=Integer,Description="Number of Samples With Data">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Total Depth">
##INFO=<ID=AF,Number=A,Type=Float,Description="Allele Frequency">
##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
##INFO=<ID=DB,Number=0,Type=Flag,Description="dbSNP membership, build 129">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
##FILTER=<ID=q10,Description="Quality below 10">
##FILTER=<ID=s50,Description="Less than 50% of samples have data">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##FORMAT=<ID=HQ,Number=2,Type=Integer,Description="Haplotype Quality">
```

single Variant calls

#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO	FORMAT	NA00001	NA00002	NA00003
20	14370	rs6054257	G	A	29	PASS	NS=3;DP=14;AF=0.5;DB;H2	GT:GQ:DP:HQ	0|0:48:1:51,51	1|0:48:8:51,51	1/1:43:5:.,.
20	17330	.	T	A	3	q10	NS=3;DP=11;AF=0.017	GT:GQ:DP:HQ	0|0:49:3:58,50	0|1:3:5:65,3	0/0:41:3
20	1110696	rs6040355	A	G,T	67	PASS	NS=2;DP=10;AF=0.333,0.667;AA=T;DB	GT:GQ:DP:HQ	1|2:21:6:23,27	2|1:2:0:18,2	2/2:35:4
20	1230237	.	T	.	47	PASS	NS=3;DP=13;AA=T	GT:GQ:DP:HQ	0|0:54:7:56,60	0|0:48:4:51,51	0/0:61:2
20	1234567	microsat1	GTC	G,GTCT	50	PASS	NS=3;DP=9;AA=G	GT:GQ:DP	0/1:35:4	0/2:17:2	1/1:40:3

Genotype variables ↔ Individual Samples

valid?

Genotype specifications

- VCF records are oriented to provide details of single variant calls.
- Not all records in a VCF are true calls, the **FILTER** column specifies those which passed the calling.
- **QUAL** is the score assigned to a given call. The greater QUAL is, the more reliable is. It is in log-scale.
- **ID** is an identifier. E.g. a dbSNP id.

A PDF with the v4.2 specifications: http://samtools.github.io/hts-specs/VCFv4.2.pdf

What is a VCF and how to interpret it : https://software.broadinstitute.org/gatk/guide/article?id=1268

Extra: VCF from callers

Allele1 / Allele2 (diploid)
1/1 → homozygous mutant
0/1 → heterozygous mutant
0/0 → homozygous reference

```
## HEADER
#CHROM    POS       ID   REF   ALT   QUAL   FILTER     INFO    FORMAT   SampleName
chr17     87234     .    G     A     2000   PASS       DP=80   GT:PL    1/1:3000,220,0
chr17     98764     .    T     C     340    PASS       DP=30   GT:PL    0/1:1200,0,200
chr17     108764    .    G     C     10     FILTERED   DP=7    GT:PL    0/1:37,0,200
```

Genomic coordinates

Nucleotide change

score
(higher → better)

filtered?

Likelihood for each GT:
0/0, 0/1, 1/1.
(lower→ better)
0 is the best score.

More info.:
https://samtools.github.io/hts-specs/VCFv4.2.pdf
https://www.broadinstitute.org/gatk/guide/article?id=1268

Extra: Data formats cheat sheet

Format	Uses	Example	File type	Software Management	File Extension
Fasta	**Human genome** Define biological sequences (DNA, RNA, cDNA, proteins).	human_genome.fa	Plain text	samtools, picard-tools	.fa; .fasta
FastQ	**Raw sequencing data** Single-end sequencing → 1 file Paired-end sequencing → 2 files (R1 and R2 for each end, respectively)	DNAseq_raw_data.fastq (DNAseq_R1.fastq and DNAseq_R2.fastq)	Plain text	samtools, picard-tools **Aligners**	.fq; .fastq
SAM	Define read alignments. Store alignment meta-info (reference, methods, one- or multi-sample).	mapped_reads.sam	Plain text	samtools, picard-tools	.sam
BAM	**VISUALIZE ALIGNMENTS (IGV)** The same as SAM, but compressed and indexed. Also to store UNMAPPED reads (compressed).	mapped_reads.bam unmapped_reads.bam	Binary	samtools, bcftools, picard-tools, **IGV** (Integrative Genome Viewer)	.bam
VCF	**SNV & Indels calls** Indicates genomic variations. Store Variant calling meta-info (reference, methods,one- or multi-sample).	point_variants.vcf	Plain text	bcftools, Unix	.vcf
BED	**Intervals** Delimit genomic regions (i.e. intervals) w or w/o annotations.	targeted_regions.bed intervals.bed	Plain text	bedtools, Unix **GATK, picard-tools**	.bed
TSV or CSV	Create data matrix (rows X Columns)	annotated_variants.tsv	Plain text	Unix, Microsoft Excel, OpenOffice	.tsv; .csv; .txt

GULBENKIAN CIÊNCIA

Thanks!

Credits for many class materials to:

Elena Piñeiro: epineiro@cnio.es
Javier Perales-Patón: jperales@cnio.es

cnio SPANISH NATIONAL CANCER RESEARCH CENTRE

BU

CNIO **BIOINFORMATICS** UNIT

Vep annotation cache

MENU

Variant Effect Predictor Download and install

Download

Download ensembl-vep package (see below the different ways to download it) and then follow the installation instructions.

https://m.ensembl.org/info/docs/tools/vep/script/vep_download.html#installer

```
$ mamba install -c bioconda perl-dbi
```

```
$ git clone https://github.com/Ensembl/ensembl-vep.git
$ cd ensembl-vep
$ git pull
$ git checkout release/104
$ perl INSTALL.pl --AUTO cf --ASSEMBLY GRCh37 --SPECIES "Homo_sapiens"
```