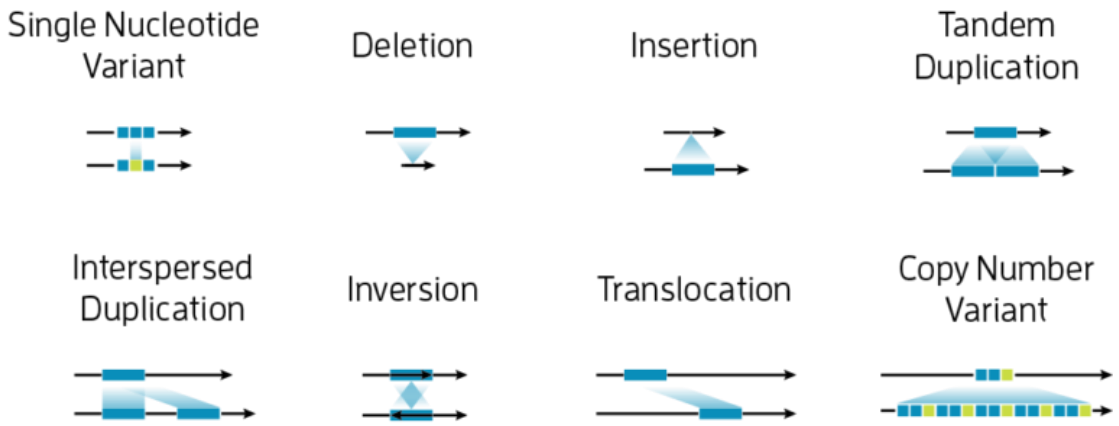
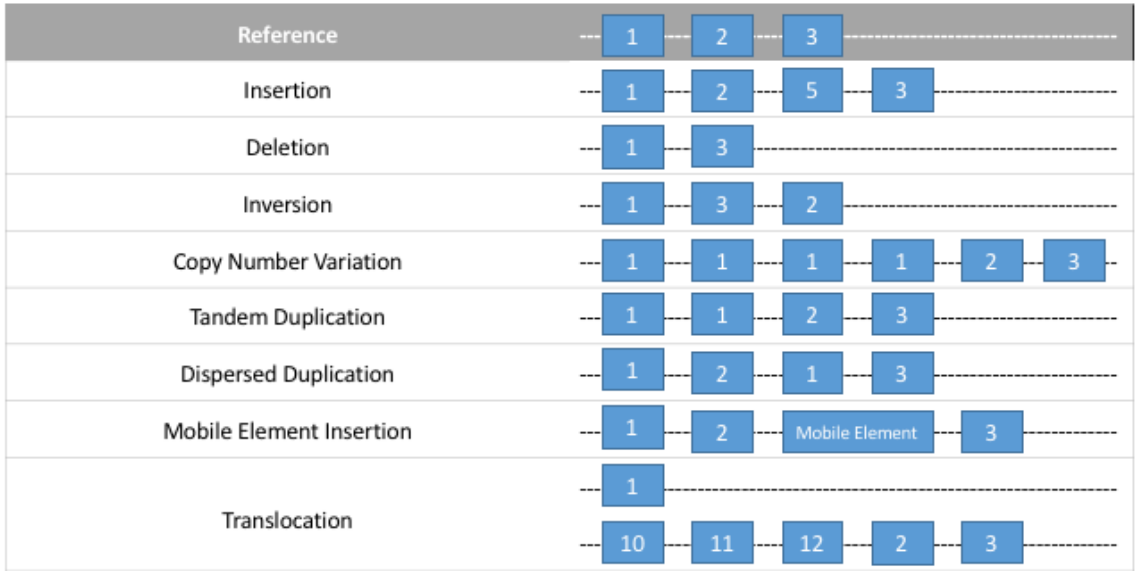
Webinars On:
Basic and Advance Next Generation Sequencing (NGS) data analysis
(July 24th, 2020)

# Variant calling, annotation and Visualization

*By: RAHILA SARDAR*
*ICMR- Senior Research Fellow*
*Translational Bioinformatics Group, ICGEB*

# Genomic Variation




https://en.wikipedia.org/wiki/Structural_variation_in_the_human_genome

# Types of mutations

**Somatic mutation**

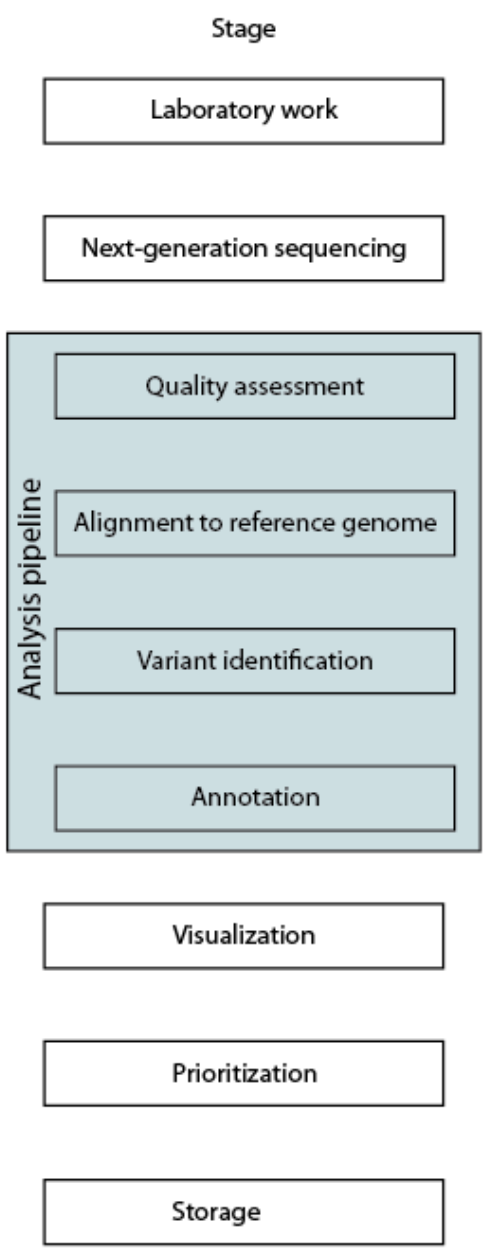
- Synonymous mutation
- Non- synonymous mutation

**Germline mutation**

- Synonymous mutation
- Non-synonymous mutation

- Synonymous mutations: which do not alter amino acid sequences and are (sometimes) silent mutations.
- Non- synonymous mutations: nucleotide mutation that alters the amino acid sequence of a protein.
- Germline mutations: occur in gametes and can be passed onto offspring (every cell in the entire organism will be affected)
- Somatic mutations: occur in a single body cell and cannot be inherited (only tissues derived from mutated cell are affected)

# Variant calling workflow




| Stage | Examples/explanation | File formats |
|---|---|---|
| Laboratory work | Experimental design<br>Library preparation<br>Enrichment (capture) | |
| Next-generation sequencing | Platforms include Illumina, SOLiD, Pacific Biosciences, other | Output: FASTQ-Sanger, FASTQ-Illumina<br>SRA, EBI,ENA etc |
| Quality assessment | Trimming, filtering<br>Software: FastQC | FASTQ<br>TrimGalore, Trimmomatic, PrinSeq |
| Alignment to reference genome | Software: BWA, Bowtie2 | Reference: FASTA<br>Output: SAM/BAM |
| Variant identification | Single nucleotide variants (SNVs), structural variants (e.g. indels)<br>Software: GATK, SAMTools<br>Realignment, recalibration | Variant Call Format<br>( VCF/BCF) |
| Annotation | Comparison to public database (dbSNP, 1000 Genomes); functional consequence scores | |
| Visualization | Variant visualization; read depth; comparison to other samples<br>Software: IGV, BEDTools, BigBED | |
| Prioritization | Discovery of relevant variants<br>Software: PolyPhen-2, VEP, VAAST<br>SIFT, Panther | VCF |
| Storage | Deposit data in ENA, SRA, dbGaP | BAM, VCF |

# Reference databases

**Reference genome database**

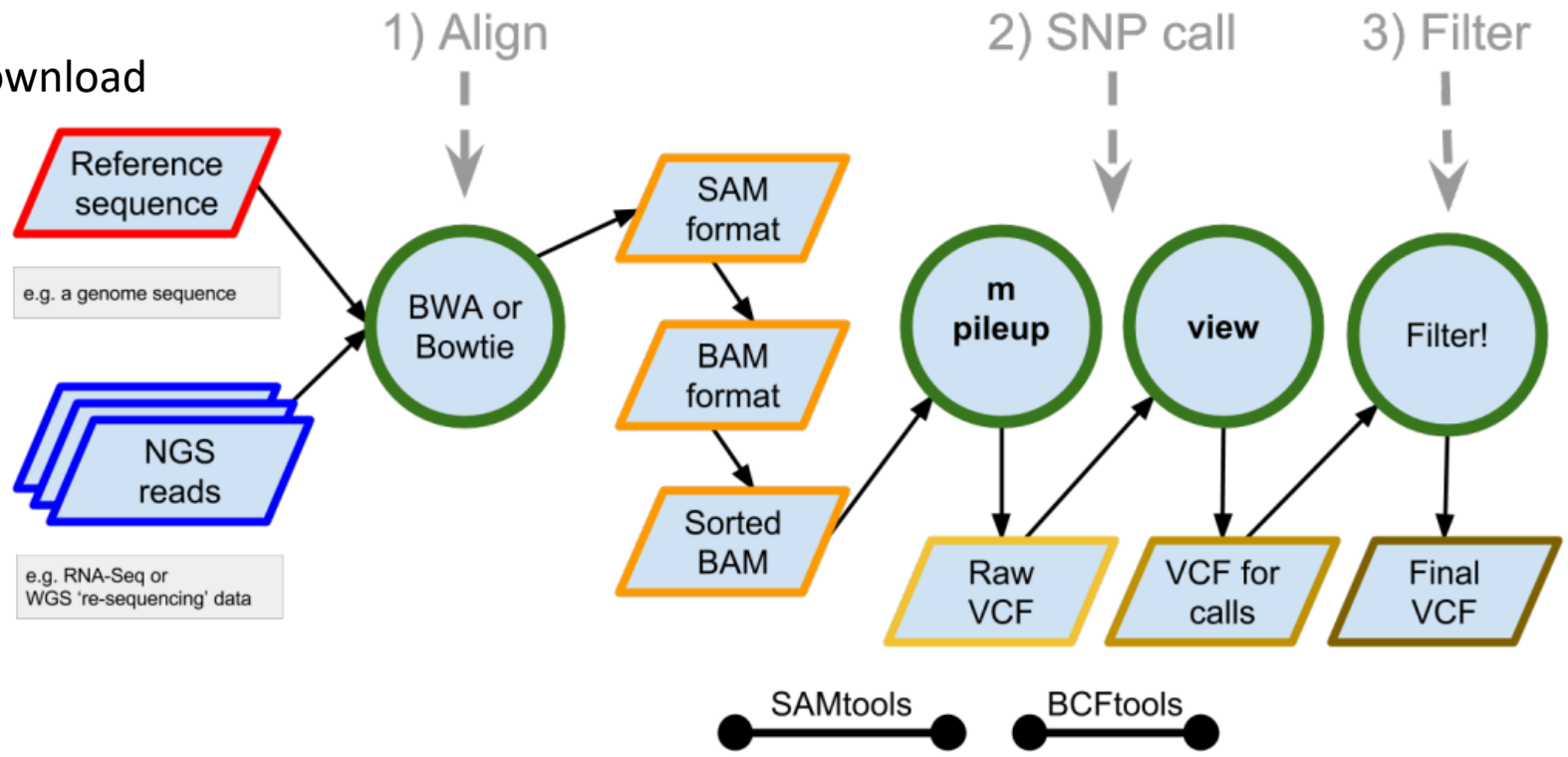
- **NCBIRefSeq:**https://www.ncbi.nlm.nih.gov/refseq/
- **UCSC :** https://genome.ucsc.edu
- **Ensembl: www.ensembl.org**

**SNP database**

- **dbSNP :** Database of short Genetic Variants
- **dbVar :** Database of genomic structural variants
- **ClinVAR:** Genomic variations and their relationship to human health and disease
- **dbGap:** Database of genotypes and phyenotypes

# Variant calling by samtools pipeline

wget link to download

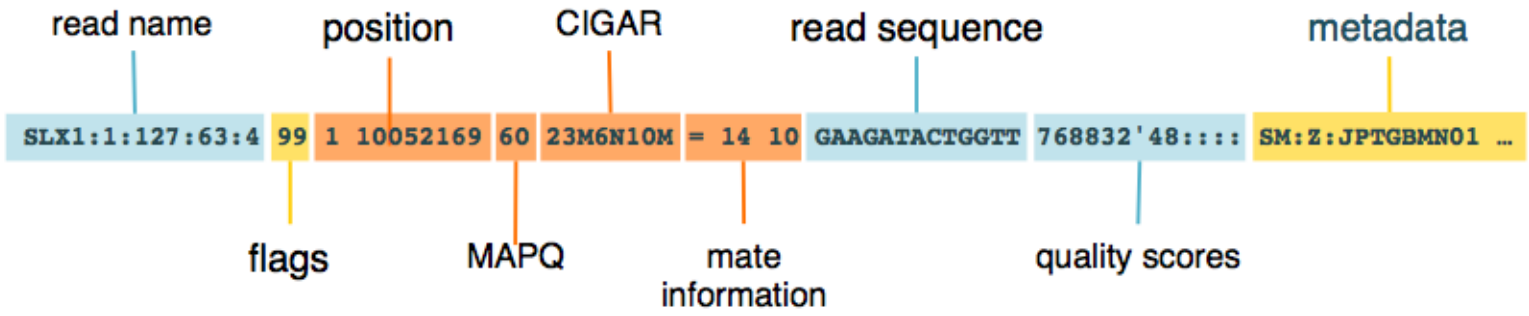



Source: EBI

# Sequence alignment/map format (SAM) and BAM

- SAM is a common format having sequence reads and their alignment to a reference genome.
- BAM is the binary form of a SAM file.
- Aligned BAM files are available at repositories (Sequence Read Archive at NCBI, ENA at Ensembl)
- SAMTools is a software package commonly used to analyze SAM/BAM files.
- Visit http://samtools.sourceforge.net/

**HEADER** containing metadata (sequence dictionary, read group definitions etc)
**RECORDS** containing structured read information (1 line per read record)



# Align reads to reference (using BWA)

1. Index the reference (genome) sequence

➢ bwa index e.coli.fasta

➢ # The various index files are output in the CWD

2. Perform the alignment

➢ bwa aln [opts] e.coli.fasta input1.fastq > output.sai

3. Output results in SAM format

➢ bwa samse e.coli.fasta output.sai input1.fastq > output.sam {SIngle end}

➢ bwa sampe e.coli.fasta output.sai input1.fastq input2.fastq > output.sam {paired end}

$ bwa mem e.coli.fasta output.sai input1.fastq input2.fastq > ouput.sam {paired end}

Index file= e.coli..fasta.ann, e.coli.fasta.bwt, e.coli.fasta.fai,e.coli.fasta.pac, e.coli.fasta.sa
*bwa sampe: single end , sampe for paired end. BWA-MEM also has better performance than BWA-backtrack for 70-100bp Illumina reads.

# 1... convert alignments (using SAMtools)

1. Convert SAM to BAM for sorting

➢ samtools view -S -b output.sam > output.bam

2. Sort BAM for SNP calling

➢ samtools sort output.bam output-sorted

Alignments are both:

- compressed for long term storage and
- sorted for variant discovery. Sort alignments by leftmost coordinates, or by read name when -n is used.

# 2) Call SNPs (using SAMtools)

1. Index the genome assembly (again!)

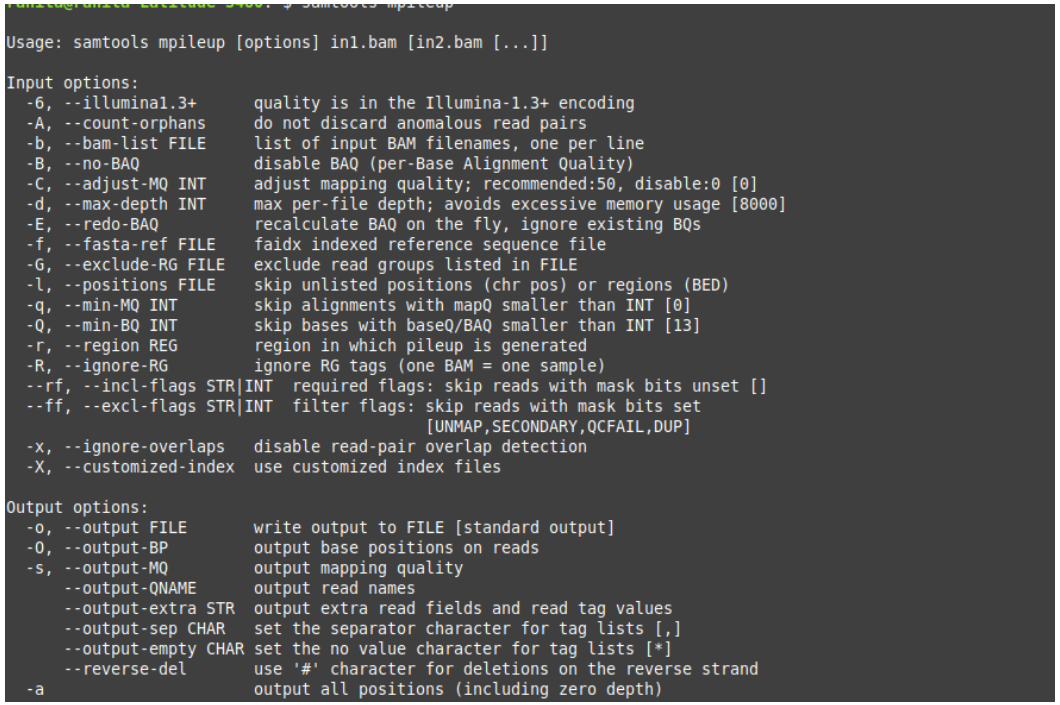
➢ samtools faidx e.coli.fasta

All we did so far (roughly) is to perform a formatconversion from BAM to VCF!

**faidx:** Index reference sequence in the FASTA format or extract subsequence from indexed reference sequence. If no region is specified, faidx will index the file and create <ref.fasta>.fai on the disk. If regions are specified, the subsequences will be retrieved and printed to stdout in the FASTA format.

# 2. Run 'mpileup' to generate VCF format

➢ samtools mpileup  -f e.coli.fasta output-sorted1.bam output-sorted-2.bam output_sorted-n.bam > output_raw.bcf




**mpileup:** Generate text pileup output for one or multiple BAM files.

# 3) Call SNPs (using bcftools)

## 3. Call SNPs Bcf to vcf

bcftools view output.var.bcf >output.var.vcf

```
Output options:
    -G,   --drop-genotypes             drop individual genotype information (after subsetting if -s option set)
    -h/H, --header-only/--no-header    print the header only/suppress the header in VCF output
    -l,   --compression-level [0-9]    compression level: 0 uncompressed, 1 best speed, 9 best compression [-1]
          --no-version                 do not append version and command line to the header
    -o,   --output-file <file>         output file name [stdout]
    -O,   --output-type <b|u|z|v>      b: compressed BCF, u: uncompressed BCF, z: compressed VCF, v: uncompressed VCF [v]
    -r, --regions <region>             restrict to comma-separated list of regions
    -R, --regions-file <file>          restrict to regions listed in a file
    -t, --targets [^]<region>          similar to -r but streams rather than index-jumps. Exclude regions with "^" prefix
    -T, --targets-file [^]<file>       similar to -R but streams rather than index-jumps. Exclude regions with "^" prefix
        --threads <int>                use multithreading with <int> worker threads [0]

Subset options:
    -a, --trim-alt-alleles        trim ALT alleles not seen in the genotype fields (or their subset with -s/-S)
    -I, --no-update               do not (re)calculate INFO fields for the subset (currently INFO/AC and INFO/AN)
    -s, --samples [^]<list>       comma separated list of samples to include (or exclude with "^" prefix)
    -S, --samples-file [^]<file>  file of samples to include (or exclude with "^" prefix)
        --force-samples           only warn about unknown subset samples

Filter options:
    -c/C, --min-ac/--max-ac <int>[:<type>]      minimum/maximum count for non-reference (nref), 1st alternate (alt1), least frequent
                                                   (minor), most frequent (major) or sum of all but most frequent (nonmajor) alleles [nref]
    -f,   --apply-filters <list>               require at least one of the listed FILTER strings (e.g. "PASS,.")
    -g,   --genotype [^]<hom|het|miss>         require one or more hom/het/missing genotype or, if prefixed with "^", exclude sites with hom/het/missing genotypes
    -i/e, --include/--exclude <expr>           select/exclude sites for which the expression is true (see man page for details)
    -k/n, --known/--novel                      select known/novel sites only (ID is not/is '.')
    -m/M, --min-alleles/--max-alleles <int>    minimum/maximum number of alleles listed in REF and ALT (e.g. -m2 -M2 for biallelic sites)
    -p/P, --phased/--exclude-phased            select/exclude sites where all samples are phased
    -q/Q, --min-af/--max-af <float>[:<type>]    minimum/maximum frequency for non-reference (nref), 1st alternate (alt1), least frequent
                                                   (minor), most frequent (major) or sum of all but most frequent (nonmajor) alleles [nref]
    -u/U, --uncalled/--exclude-uncalled        select/exclude sites without a called genotype
    -v/V, --types/--exclude-types <list>       select/exclude comma-separated list of variant types: snps,indels,mnps,ref,bnd,other [null]
```

bcftools view: Applies the prior and does the actual calling and convert bcf to vcf

# 4) Filter SNPs

## 1. Filter SNPs

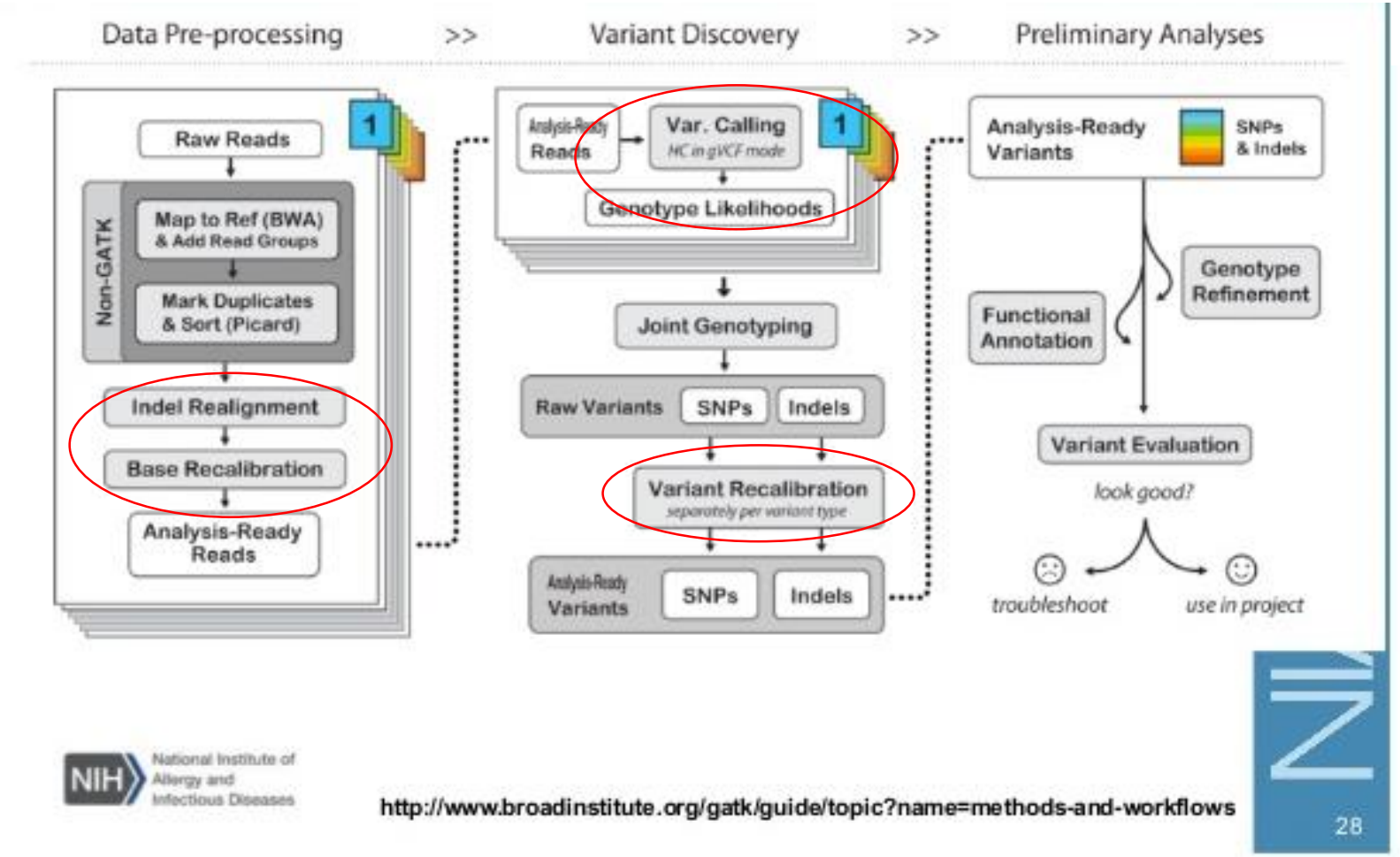
➢ vcfutils.pl varFilter output.var.vcf > outputs.var-final.vcf

```
Options: -Q INT    minimum RMS mapping quality for SNPs [10]
         -d INT    minimum read depth [2]
         -D INT    maximum read depth [10000000]
         -a INT    minimum number of alternate bases [2]
         -w INT    SNP within INT bp around a gap to be filtered [3]
         -W INT    window size for filtering adjacent gaps [10]
         -1 FLOAT  min P-value for strand bias (given PV4) [0.0001]
         -2 FLOAT  min P-value for baseQ bias [1e-100]
         -3 FLOAT  min P-value for mapQ bias [0]
         -4 FLOAT  min P-value for end distance bias [0.0001]
         -e FLOAT  min P-value for HWE (plus F<0) [0.0001]
         -p        print filtered variants

Note: Some of the filters rely on annotations generated by SAMtools/BCFtools.
```

- Note: vcfutils.pl is a perl script that helps filtering variants according to a certain set of parameters

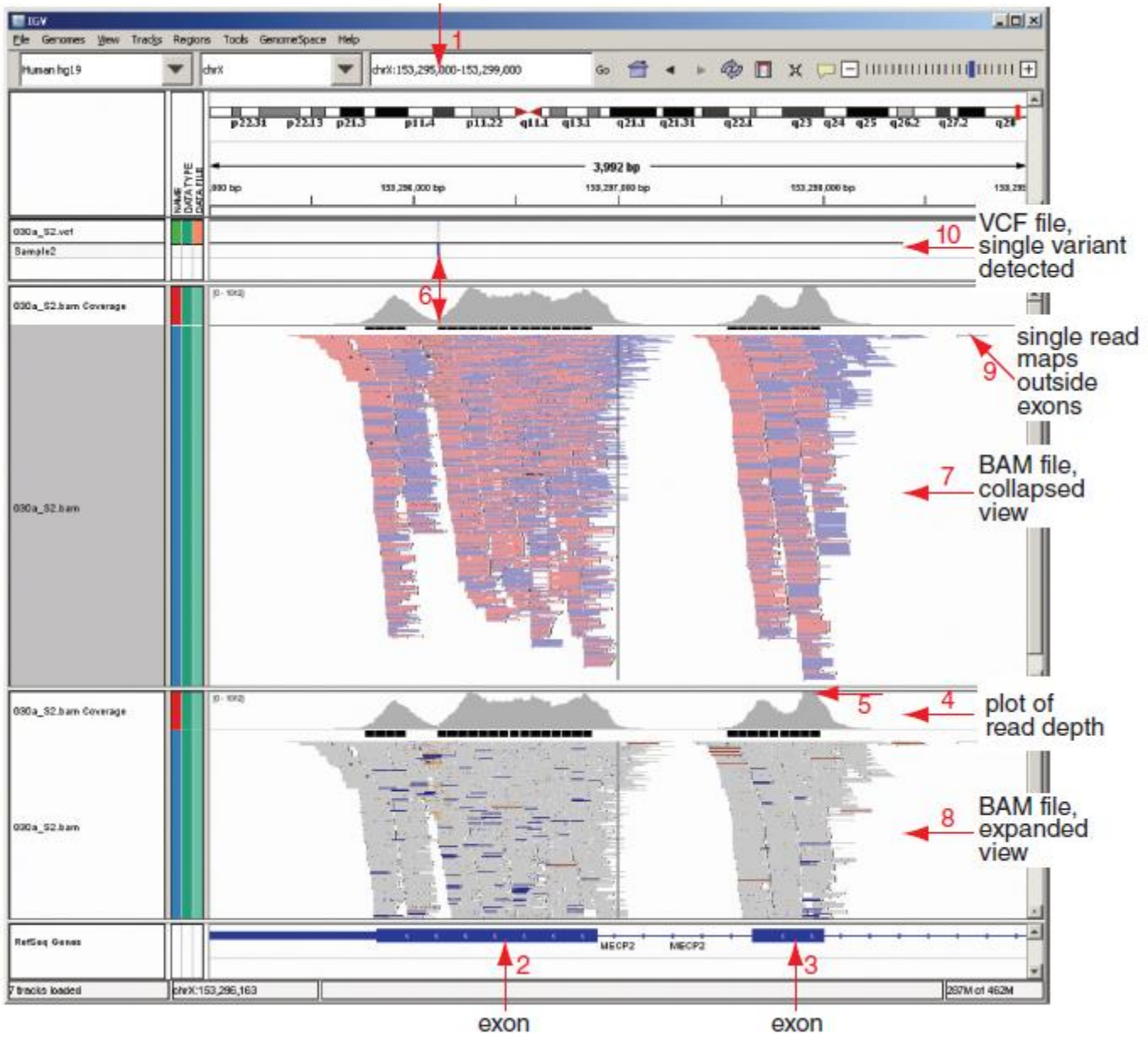
# GATK


Data Pre-processing
>>
Variant Discovery
>>
Preliminary Analyses
Raw Reads
1
Non-GATK
Map to Ref (BWA)
& Add Read Groups
Mark Duplicates
& Sort (Picard)
Indel Realignment
Base Recalibration
Analysis-Ready
Reads
Analysis-Ready
Reads
Var. Calling
HC in gVCF mode
1
Genotype Likelihoods
Joint Genotyping
Raw Variants
SNPs
Indels
Variant Recalibration
separately per variant type
Analysis-Ready
Variants
SNPs
Indels
Analysis-Ready
Variants
SNPs
& Indels
Genotype
Refinement
Functional
Annotation
Variant Evaluation
look good?
troubleshoot
use in project
NIH
National Institute of
Allergy and
Infectious Diseases
http://www.broadinstitute.org/gatk/guide/topic?name=methods-and-workflows
28

# GATK vs Samtools

| Variant callers | Samtools | GATK |
|---|---|---|
| Algorithm | bayesian approach to call the variants | bayesian approach to call the variants |
| Preprocess of alignment. | No | uses picard tools |
| GATK specific features | No other variant callers | **haplotype caller, indel realignment,UnifiedGenotype among many others** |
| Recalibration | No | **Yes using ML algorithm** |
| Organism specific | No | **Yes** |
| Utility | samtools is equally scalable to huge data sets in spite of its much simpler framewor | GATK is clearly the more sophisticated and the more complicated |

# Integrative Genomics Viewer (IGV)



# Variant Call Format (VCF) file summarizes variation

| Column | Mandatory | Description |
|---|---|---|
| CHROM | Yes | Chromosome |
| POS | Yes | 1-based position of the start of the variant |
| ID | Yes | Unique identifier of the variant; the dbSNP entry rs1413368 is given in our example |
| REF | Yes | Reference allele |
| ALT | Yes | A comma-separated list of alternate nonreference alleles |
| QUAL | Yes | Phred-scaled quality score |
| FILTER | Yes | Site filtering information; in our example it is PASS |
| INFO | Yes | A semicolon-separated list of additional information. These fields include the gene identifier GI (here the gene is NEGR1); the transcript identifier TI (here NM_173808); and the functional consequence FC (here a synonymous change, T296T). |
| FORMAT | No | Defines information in subsequent genotype columns; colon separated. For example, GT:AD:DP:GQ:PL:VF:GQX in our example refers to genotype (GT), allelic depths for the ref and alt alleles in the order listed (AD), approximate read depth (reads with MQ=255 or with bad mates are filtered) (DP), genotype quality (GQ), normalized, Phred-scaled likelihoods for genotypes as defined in the VCF specification (PL), variant frequency, the ratio of the sum of the called variant depth to the total depth (VF), and minimum of {genotype quality assuming variant position, genotype quality assuming nonvariant position} (GXQ). |
| Sample | No | Sample identifiers define the samples included in the VCF file |

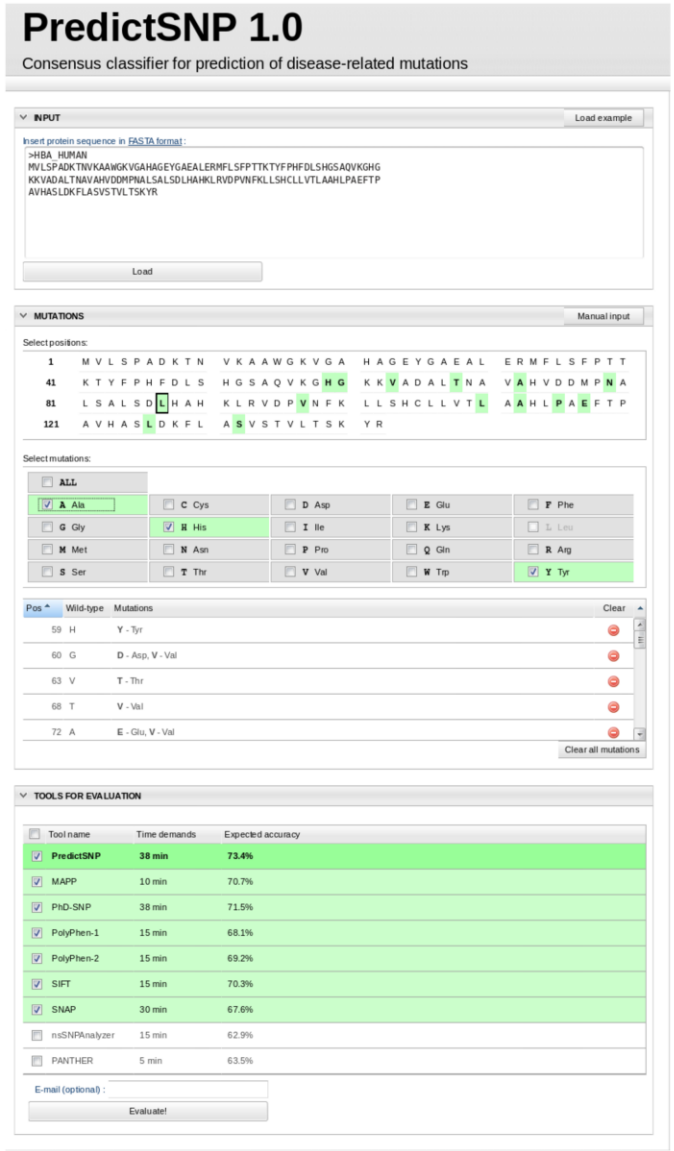
# How to annotate and prioritize SNP?

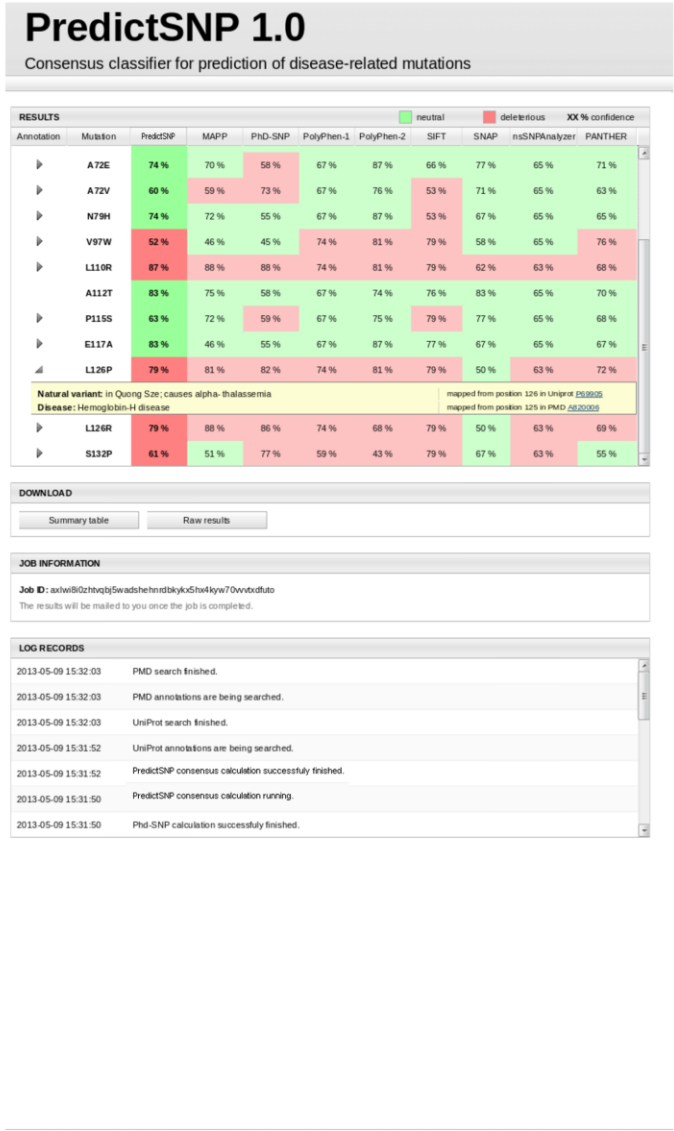

| Tool name | Version | Principle | Training dataset |
|---|---|---|---|
| MAPP | 28.6.2005 | Physicochemical properties and alignment score | No training dataset |
| nsSNPAnalyzer | 12.2.2004 | Random forest | SwissProt 3,511 neutral/502 deleterious |
| PANTHER | 1.02 | Hidden Markov model and alignment score | No training dataset |
| PhD-SNP | 2.06 | Support vector machine | SwissProt 17,983 neutral/16,330 deleterious |
| PolyPhen-1 | 1.18 | Expert set of empirical rules | No training dataset |
| PolyPhen-2 | 2.2.2 | Naïve Bayes | SwissProt, dbSNP 7,070 neutral/5,322 deleteriou |
| SIFT | 4.0.4 | Alignment score | No training dataset |
| SNAP | 1.1.30 | Neural network | SwissProt, Protein Mutant Database 40,830 neutral/39,987 deleterious |

doi:10.1371/journal.pcbi.1003440.t001

https://genomeinterpretation.org/vipDB

# SNP prioritization



# Ensembl “Variation Table” shows SIFT and PolyPhen scores for nonsynonymous variants

## Missense variants

[back t

Show All entries | Show/hide columns | Filter

| ID | Chr: bp | Alleles | Class | Source | Type | AA | AA coord | SIFT | PolyPhen |
|---|---|---|---|---|---|---|---|---|---|
| rs121909815 | 11:5248247 | A/G | SNP | dbSNP | Missense variant | V/A | 2 | 0.01 | 0.119 |
| rs121909830 | 11:5248247 | A/C | SNP | dbSNP | Missense variant | V/G | 2 | 0.07 | 0.007 |
| rs121909815 | 11:5248247 | A/G | SNP | dbSNP | Missense variant | V/A | 2 | 0.01 | 0.119 |
| rs121909830 | 11:5248247 | A/C | SNP | dbSNP | Missense variant | V/G | 2 | 0.01 | 0.007 |
| rs33958358 | 11:5248248 | C/T/**A** | SNP | dbSNP | Missense variant | V/L | 2 | 0.01 | 0.001 |
| rs33958358 | 11:5248248 | C/**T**/A | SNP | dbSNP | Missense variant | V/M | 2 | 0 | 0.271 |
| rs33958358 | 11:5248248 | C/T/**A** | SNP | dbSNP | Missense variant | V/L | 2 | 0.02 | 0.001 |
| rs33958358 | 11:5248248 | C/**T**/A | SNP | dbSNP | Missense variant | V/M | 2 | 0 | 0.271 |
| rs35906307 | 11:5248245 | G/A | SNP | dbSNP | Missense variant | H/Y | 3 | 0.02 | 0.135 |

**Research Article | Open Access**

Volume 2016 | Article ID 9313746 | 5 pages | https://doi.org/10.1155/2016/9313746

# In Silico Analysis of SNPs in *PARK2* and *PINK1* Genes That Potentially Cause Autosomal Recessive Parkinson Disease

**Yousuf Hasan Yousuf Bakhit**, [1] Mohamed Osama Mirghani Ibrahim,[2] Mutaz Amin [4]

Yousi

Show

Acad

## In silico analysis of a novel causative mutation in Cadherin23 gene identified in an Omani family with hearing loss

Mohammed Nasser Al-Kindi, Mazin Jawad Al-Khabouri, Khalsa Ahmad Al-Lamki, Flavia Palombo, Tommaso Pippucci, Giovanni Romeo & Nadia Mohammed Al-Wardy

*Journal of Genetic Engineering and Biotechnology* **18**, Article number: 8 (2020) | Cite this article

**664** Accesses | Metrics

### Abstract

#### Background

Hereditary hearing loss is a heterogeneous group of complex disorders with an overall incidence of one in every 500 newborns presented as syndromic and non-syndromic forms. Cadherin-related 23 (CDH23) is one of the listed deafness causative genes. It is found to be expressed in the stereocilia of hair cells and in the retina photoreceptor cells. Defective CDH23 have been associated mostly with prelingual severe-to-profound sensorineural hearing loss (SNHL) in either syndromic (USH1D) or non-syndromic SNHL (DFNB12) deafness. The purpose of this study was to identify causative mutations in an Omani family diagnosed with severe-profound sensorineural hearing loss by whole exome sequencing technique and analyzing the detected variant in silico for pathogenicity using several in silico mutation prediction software.



OPEN ACCESS PEER-REVIEWED

RESEARCH ARTICLE

## *In silico* identification of genetic mutations conferring resistance to acetohydroxyacid synthase inhibitors: A case study of *Kochia scoparia*

Yan Li, Michael D. Netherland, Chaoyang Zhang, Huixiao Hong, Ping Gong

Published: May 7, 2019 • https://doi.org/10.1371/journal.pone.0216116



### Abstract

Mutations that confer herbicide resistance are a primary concern for herbicide-based chemical control of invasive plants and are often under-characterized structurally and functionally. As the outcome of selection pressure, resistance mutations usually result from repeated long-term applications of herbicides with the same mode of action and are discovered through extensive field trials. Here we used acetohydroxyacid synthase (AHAS) of *Kochia scoparia* (*Ks*AHAS) as an example to demonstrate that, given the sequence of a target protein, the impact of genetic mutations on ligand binding could be evaluated and resistance mutations could be identified using a biophysics-based computational approach. Briefly, the 3D structures of wild-type (WT) and mutated *Ks*AHAS-herbicide complexes were constructed by homology modeling, docking and molecular dynamics simulation. The resistance profile of two AHAS-inhibiting herbicides, tribenuron methyl and thifensulfuron methyl, was obtained by estimating their binding affinity with 29 *Ks*AHAS (1 WT and 28 mutated) using 6 molecular mechanical (MM) and 18 hybrid quantum mechanical/molecular mechanical (QM/MM) methods in combination with three structure sampling strategies. By comparing predicted resistance with experimentally determined resistance in the 29 biotypes of *K. scoparia* field populations, we identified the best method (i.e., MM-PBSA with single structure) out of all tested methods for the herbicide-*Ks*AHAS system, which exhibited the highest accuracy (up to 100%) in discerning mutations

# Thank You and Questions