VacCAP Final Documentation

All scripts used are located at:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts

**Raw data collection**

NCBI – NCBI.xlsx

* Identified all Vaccinium SRA files – 387 SRR entries
* Downloaded files for species used in blueberry breeding – 255 SRR entries

|  |
| --- |
| Genus Species |
| Vaccinium corymbosum |
| Vaccinium macrocarpon |
| Vaccinium virgatum |
| Vaccinium angustifolium x Vaccinium corymbosum |
| Vaccinium ashei |
| Vaccinium oxycoccos |
| Vaccinium caespitosum |
| Vaccinium myrtilloides |

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/fastq-dump.sh

Base Code:

fastq-dump --split-files SRR#

* Concatenated files for the same cultivar in the same BioProject – 94 SRR entries

NCGR – NCGR.xlsx

* Identified Vaccinium files stored on the CGRB – 9 samples
* Concatenated replicate files for the same cultivar – 5 cultivar entries

Collaborator – Collaborator.xlsx

* Received 99 raw samples from Breeding insight (59), Pat Edger (36), and Susan Thompson (4), respectively
* Concatenated replicate files of the same cultivar – Breeding insight (31), Pat Edger (36), and Susan Thompson (2)
* Received SNP/SV data from Breeding Insight, David Chagne, Driscoll’s, Ebrahiem Babiker, Hamid Ashrafi, Jeannine Rowland, Kalpalatha Melmaiee, Massimo Iorizzo, and Patricio Munzo

**Cleaning**

* Ran FastQC on all files to identify contaminates (adapters, poly sequences, fungal/bacterial, and plasmid), minimum read length cutoff, minimum quality score cutoff

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/fastqc\_single-end.sh

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/fastqc\_paired\_end.sh

Base Code:

fastqc -t 10 contaminated\_file.fq -o path/to/directory

* Cleaned files with bbduk

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/bbduk\_filtering\_contam.sh

Base Code:

bbduk.sh \

in1=contaminated\_1.fq \

in2=contaminated\_2.fq \

out1=clean\_1.fq \

out2=clean\_1.fq \

ref=adapters.fa, fastqc\_adapters.fa, poly.fa, contamination.fa \

ktrim=r k=21 mink=11 hdist=2 tpe tbo maq=25 minlen=25

* Ran FastQC on all decontaminated files to double check cleaning worked.

Base code:

fastqc -t 10 clean\_file.fq -o path/to/directory

**Alignement**

Genomic

* Aligned files with BWA mem

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/BWA-index.sh

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/BWA-mem.sh

Base Code:

bwa index -a bwtsw reference.fa

bwa mem reference.fa \

clean\_1.fq \

clean\_2.fq \

| samtools sort -o sorted.bam

Transcriptomic

* Find minimum and maximum intron lengths

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/intron-length.awk

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/intron-length.sh

Base Code:

intron-length.awk TYPE=CDS \

reference.gff \

< intron\_stats.tsv

* Aligned file with STAR 2 Pass mode

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/STAR\_Indexes.sh

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/STAR\_Mapping.sh

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/STAR\_Prepare\_Splice\_Junctions.sh

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/STAR\_Indexes2.sh

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/STAR\_Mapping2.sh

Base Code:

STAR --runMode genomeGenerate \

--runThreadN 50 \

--genomeDir Vce1.0\_P0\_STAR \

--genomeFastaFiles reference.fa \

--sjdbOverhang 100 \

--sjdbGTFfile reference.gtf \

--genomeSAindexNbases 13

STAR –genomeDir Vce1.0\_P0\_STAR \

--readFilesIn clean\_1.fq clean\_2.fq \

--alignIntronMin 25 --alignIntronMax 44760 –outFileNamePrefix unique\_name\_

awk 'BEGIN {OFS="\t"; strChar[0]="."; strChar[1]="+"; strChar[2]="-";} {if($5>0){print $1,$2,$3,strChar[$4]}}' unique\_name\_SJ.out.tab > unique\_name\_SJ.in.tab

STAR --runMode genomeGenerate \

--runThreadN 14 \

--genomeDir unique\_name\_

--genomeSAindexNbases 13 \

--genomeFastaFiles reference.fa \

--sjdbGTFfile reference.gtf \

--sjdbOverhang 100 \

--sjdbFileChrStartEnd unique\_name\_SJ.in.tab

STAR –genomeDir unique\_name\_

--readFilesIn clean\_1.fq clean\_2.fq \

--alignIntronMin 25 --alignIntronMax 44760 –outFileNamePrefix unique\_name\_2PASS

**Processing Alignment Files**

* Converted sam files to bam files and sorted if needed

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/samtools\_sam\_to\_bam.sh

Base Code:

samtools view -S -b file.sam > file.bam

samtools sort file.bam \

-o sorted.bam \

-@ 50

* Removed PCR and sequencing duplicates

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/GATK\_Markduplicates.sh

Base Code:

gatk MarkDuplicates \

--INPUT sorted.bam \

--OUTPUT deduped.bam \

-M deduped\_matrix.txt \

--REMOVE\_DUPLICATES true \

--MAX\_RECORDS\_IN\_RAM 1000000 \

--TMP\_DIR /path/to/temp/direct

**Processing Alignment and Reference Files for Variant Calling**

* Added Read group Information for Variant Calling

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/GATK\_AddOrReplaceReadGroups.sh

Base Code:

gatk AddOrReplaceReadGroups \

-I deduped.bam \

-O rg.bam

--RGID read\_group\_name \

--RGLB Project\_name \

--RGPL DataType\_Layout \

--RGPU read\_group\_name \

--RGSM read\_group\_name

* Determined coverage for bam files that will be called jointly

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/samtools\_coverage.sh

Base Code:

samtools coverage --bam-list list\_of\_bams.txt \

-r chromosome

**Processing Alignment and Reference Files for Variant Calling – Genomic Pipeline**

* Merged and sort bam files that had high alignment coverage across most chromosomes

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/samtools\_merge\_sort.sh

Base Code:

samtools merge -b list\_of\_bams.txt \

merged.bam \

-m 10G \

--threads 50

samtools sort -o sorted\_merged.bam \

merged.bam \

-m 10G \

--threads 50

* Split reference.fa by regions for Freebayes

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/freebayes\_config\_VacCAP\_GenerateFreebayesRegions\_OSU.yaml

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/snakemake-freebayes-parallel\_VacCAP\_GenerateFreebayesRegions\_OSU.smk

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/snakemake-freebayes-parallel\_VacCAP\_GenerateFreebayesRegions\_OSU.sh

Base Code:

fasta\_generate\_regions.py \

reference.fa \

number\_chunks \

--chunks \

--bed prefix\_of\_out\_bed \

--chromosome chromosome

* Split merged bam by same reference regions

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/samtools\_split\_bam.sh

Base Code:

samtools view -b --threads 50 chromosome:start-stop > chrom\_start\_stop.bam

**Variant Calling**

* Variant calling with Freebayes

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/freebayes\_config\_VacCAP\_VariantCallingFreebayes\_OSU.yaml

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/snakemake-freebayes-parallel\_VacCAP\_VariantCallingFreebayes\_OSU.sh

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/snakemake-freebayes-parallel\_VacCAP\_VariantCallingFreebayes\_OSU.smk

Base Code:

freebayes -f reference.fa -t region.bed -b chrom\_start\_stop.bam --cnv-map ploidy.txt --use-best-n-alleles 3 > chrom\_start\_stop.vcf

**Process and Filter VCF Files**

* Concatenate VCF files, remove duplicate calls, and sort VCF files by choordinates

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/bcftools\_concat\_sort.sh

Base Code:

bcftools concat chrom\_\*\_\*.vcf --threads 50 | vcfuniq > ./concat\_chrom.vcf

bcftools sort ./concat\_chrom.vcf -m 10G --temp-dir path/to/temp/directory --output-file sorted\_chrom.vcf

* Filter VCF file calls per chromosome based on agreed requirements (Quality >= 20, MAF >= 10% Max depth +2 SD, each allele supported by 5 reads if present, missing allele frequency less than 20%)

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/bcftools\_filt\_vcf.sh

Base Code:

cat sorted\_chrom.vcf | bcftools view -i '%QUAL>=20 && MIN(INFO/AF)>=.10 && MAX(INFO/DP)<=7146' --threads 50 \

bcftools filter -e '(FORMAT/RO)=1 | (FORMAT/RO)=2 | (FORMAT/RO)=3 | (FORMAT/RO)=4' -S . --threads 50 \

| bcftools filter -e '(FORMAT/AO)=1 | (FORMAT/AO)=2 | (FORMAT/AO)=3 | (FORMAT/AO)=4' -S . --threads 50 \

| bcftools view -i 'F\_MISSING<=0.2' --threads 50 \

> filtered\_chrom.vcf

* Extract SV flanking sequences and create fastq to identify single mapping SVs with BWA mem

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/extract\_flanking\_sequences\_for\_SV.py

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/extract\_flanking\_sequences\_for\_SV.sh

Base Code:

extract\_flanking\_sequences\_for\_SV.py -f refercence.fasta -v filtered\_chrom.vcf -o filtered\_chrom.fastq

* Align filtered SV’s fastq with BWA

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/bwa-mem.sh

Base Code:

bwa mem refernce.fasta filtered\_chrom.fastq -t 50 | samtools sort -o sorted\_filtered\_chrom.bam

* Identify single mapping SVs with BWA mem alignment

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/filt\_bams\_for\_blast.py

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/filt\_bams\_for\_blast.sh

Base Code:

filt\_bams\_for\_blast.py -b sorted\_filtered\_chrom.bam -f filtered\_chrom.fastq -o BWA\_filtered\_chrom.fasta

* Align filtered SV’s fasta with BLASTn

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/blastn\_database.sh

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/blastn.sh

Base Code:

makeblastdb -in reference.fasta -parse\_seqids -dbtype nucl

blastn -db reference.fasta

-query BWA\_filtered\_chrom.fasta \

-num\_threads 50 \

-outfmt "6 qseqid sseqid qstart qend sstart send evalue pident length qseq sseq" \

-out blast\_bwa\_filtered.txt

* Identify single mapping SVs with BLASTn alignments

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/FilterVCFwithBLASTn.py

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/ FilterVCFwithBLASTn.sh

Base Code:

FilterVCFwithBLASTn.py \

-b blast\_bwa\_filtered.txt \

-f filtered\_chrom.fastq \

-v filtered\_chrom.vcf \

-k single\_mapping\_chrom.txt \

-fv single\_mapping\_chrom.vcf

**Variant Calling – Transcript Pipeline**

* Variant calling with Freebayes

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/Freebayes.sh

Base Code:

freebayes -f reference.fa -b chrom\_start\_stop.bam --cnv-map ploidy.txt > transcript.vcf

* Filter VCF file calls based on agreed requirements (Quality >= 20, MAF >= 10% Max depth +2 SD, each allele supported by 5 reads if present, missing allele frequency less than 20%)

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/bcftools\_filt\_vcf.sh

Base Code:

cat transcipt.vcf | bcftools view -i '%QUAL>=20 && MIN(INFO/AF)>=.10 && MAX(INFO/DP)<=7146' --threads 50 \

bcftools filter -e '(FORMAT/RO)=1 | (FORMAT/RO)=2 | (FORMAT/RO)=3 | (FORMAT/RO)=4' -S . --threads 50 \

| bcftools filter -e '(FORMAT/AO)=1 | (FORMAT/AO)=2 | (FORMAT/AO)=3 | (FORMAT/AO)=4' -S . --threads 50 \

| bcftools view -i 'F\_MISSING<=0.2' --threads 50 \

> filtered\_transcript.vcf

* Extract SV flanking sequences and create fastq to identify single mapping SVs with BWA mem

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/extract\_flanking\_sequences\_for\_SV.py

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/extract\_flanking\_sequences\_for\_SV.sh

Base Code:

extract\_flanking\_sequences\_for\_SV.py -f refercence.fasta -v filtered\_transcipt.vcf -o filtered\_transcript.fastq

* Align filtered SV’s fastq with BWA

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/bwa-mem.sh

Base Code:

bwa mem refernce.fasta filtered\_transcript.fastq -t 50 | samtools sort -o sorted\_transcript..bam

* Identify single mapping SVs with BWA mem alignment

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/filt\_bams\_for\_blast.py

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/filt\_bams\_for\_blast.sh

Base Code:

filt\_bams\_for\_blast.py -b sorted\_transcript.bam -f filtered\_ transcript.fastq -o BWA\_filtered\_transcript.fasta

* Align filtered SV’s fasta with BLASTn

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/blastn\_database.sh

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/blastn.sh

Base Code:

makeblastdb -in reference.fasta -parse\_seqids -dbtype nucl

blastn -db reference.fasta

-query BWA\_filtered\_transcript.fasta \

-num\_threads 50 \

-outfmt "6 qseqid sseqid qstart qend sstart send evalue pident length qseq sseq" \

-out blast\_bwa\_filtered\_transcript.txt

* Identify single mapping SVs with BLASTn alignments

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/FilterVCFwithBLASTn.py

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/ FilterVCFwithBLASTn.sh

Base Code:

FilterVCFwithBLASTn.py \

-b blast\_bwa\_filtered.txt \

-f filtered\_chrom.fastq \

-v filtered\_chrom.vcf \

-k single\_mapping\_transcript.txt \

-fv single\_mapping\_transcript.vcf

**Existing SNP Pipeline**

* Collected all existing collaborator SVs and supplementary information.

<https://docs.google.com/spreadsheets/d/1e08Ly6o2hmGCZuZUTGFTy-TDXaXFJI3G/edit?usp=sharing&ouid=104879950178892330058&rtpof=true&sd=true>

* To find their positions on the W85 P0, we extracted the flanking sequences from the indicated reference or were provided the flanking sequences. The flanking sequences with the SV were aligned to the W85 P0 reference with BWA mem. Actual SV chom, pos, and supplementary alignment information was determined with a custom Python script. This information was added to the above google sheet.

Script usage and examples:

See BWA mem alignment above for source code.

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/Find\_SV\_Positions.py

**Creating Tables per Chromosome– Genomic (de novo), transcriptomic, and existing**

* Created an excel sheet and converted to text file that had all the chromosome and positions for each collaborator and transcript project.

File access:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/VCF\_files/VacCAP/ All\_collaborator\_Transcript\_SNP\_POS.txt

* Identify and created the shared SV tables

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/Final\_Table\_shared.py

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/snakemake-Final\_Table\_shared.yaml

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/snakemake-Final\_Table\_shared.smk

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/snakemake-Final\_Table\_shared.sh

Base Code:

Final\_Table\_shared.py -hc W85\_Hard\_Core\_P0\_4BB.txt -sc W85\_Soft\_Core\_P0\_4BB.txt -g vcae1.4.gff3 -f single\_mapping\_chrom.txt -c All\_collaborator\_Transcript\_SNP\_POS.txt -o Table1\_files\_chrom.txt

* Identify and created the de novo SV tables

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/Final\_Table\_denovo.py

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/snakemake-Final\_Table\_denovo.yaml

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/snakemake-Final\_Table\_denovo.smk

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/snakemake-Final\_Table\_denovo.sh

Base Code:

Final\_Table\_denovo.py -hc W85\_Hard\_Core\_P0\_4BB.txt -sc W85\_Soft\_Core\_P0\_4BB.txt -g vcae1.4.gff3 -f single\_mapping\_chrom.txt -t Table1\_file\_chrom.txt -o Table2\_file\_chrom.txt

* Created an excel file and converted to text file that has all the chromosome, position, priority, QTL trait, and QTL citation information for collaborators that contributed this information

File access:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/VCF\_files/VacCAP/Collab\_priority\_trait\_citation\_data.txt

* Combine the shared table (Table 1) and de novo table (Table 2)

Script usage and examples:

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/Final\_Table\_combined.py

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/snakemake-Final\_Table\_combined.yaml

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/snakemake-Final\_Table\_combined.smk

/nfs3/HORT/Bassil\_Lab/mandie/Blueberry\_VacCap\_Project/scripts/snakemake-Final\_Table\_combined.sh

Base Code:

Final\_Table\_combined.py -t1 Table1\_file\_chrom.txt -t2 Table2\_file\_chrom.txt -c Collab\_priority\_trait\_citation\_data.txt -o MasterTable\_chrom.txt