Vaccinium floribundum Genome Assembly and Annotation Script

Martina Albuja-Quintana[1], Gabriela Pozo[1], Milton Gordillo-Romero[1], Carolina E. Armijos[1], Maria de Lourdes Torres[1]

[1]Universidad San Francisco de Quito

Martina Albuja-Quintana
Universidad San Francisco de Quito

APR 03, 2024

DOI:
dx.doi.org/10.17504/protocols.io.n92ldmo4nl5b/v1

Protocol Citation: Martina Albuja-Quintana, Gabriela Pozo, Milton Gordillo-Romero, Carolina E. Armijos, Maria de Lourdes Torres 2024. Vaccinium floribundum Genome Assembly and Annotation Script. **protocols.io** https://dx.doi.org/10.17504/protocols.io.n92ldmo4nl5b/v1

ABSTRACT

Oxford Nanopore long reads and Illumina short reads obtained from sequencing the DNA of a *Vaccinum floribundum* specimen from the Paramo region in Ecuador, were used to assemble and annotate the whole genome of this species. ONT long reads were filtered and trimmed in Nanofilt and Porechop. Sequencing statistics were visualized in Nanoplot. Illumina short reads were evaluated with fastqc. Different assemblers and polishers were used with both long and short reads. The resulting Flye assembly polished with Polca and Medaka was then analyzed for genome completeness and quality with Quast, BUSCO, LAI Index, and Coverage Graph. The assembly was later annotated in Maker in 3 consecutive rounds using the *ab initio* gene predictor SNAP.

Protocol status: Working
We use this protocol and it's working

Created: Mar 14, 2024

Last Modified: Apr 03, 2024

PROTOCOL integer ID: 96735

Funders Acknowledgement:

Fondos COCIBA USFQ

Oxford Nanopore Sequencing - Raw Reads Filtering, Trimming, and Statist…

1 **Raw Read Adapter Filtering**

Porechop v0.2.4 (RRID:SCR_016967)

```
porechop -i Mortino_ONT_RawReads.fastq.gz -o
Porechop_Mortino_ONT_RawReads.fastq.gz
```

2 **Raw Read Quality and Length Trimming**

Nanofilt v2.8.0 (RRID:SCR_016966)

```
NanoFilt -q 7 < Porechop_Mortino_ONT_RawReads.fastq >
Porechop_Mortino_ONT_q7.fastq

NanoFilt -l 1000 < Porechop_Mortino_ONT_q7.fastq >
Porechop_Mortino_ONT_q7_1000.fastq
```

3 **Raw Read Dataset Statistics**

LongQC v1.2.0c

```
python longQC.py sampleqc -x ont-ligation -o longQC_Mortino_ONT
Porechop_Mortino_ONT_q7_1000.fastq
```

Nanoplot v1.33.0 (RRID:SCR_024128)

```
NanoPlot --fastq Porechop_Mortino_ONT_q7_1000.fastq --readtype 1D -t 4 --
title "Nanoplot_Mortino_ONT" -o Nanoplot_Mortino_ONT
```

Illumina Sequencing - Raw Reads Statistics

4 **Raw Read Dataset Statistics**

FastQC (RRID:SCR_014583)

```
fastqc FC225MJ3LT3_L1_Mort-Illumina_1.fq.gz FC225MJ3LT3_L1_Mort-
Illumina_2.fq.gz -o FastQC_Mortino_Illumina
```

Genome Size Estimation

5 **k-mer based analysis**

Jellyfish v2.3.0 (RRID:SCR_005491)

```
zcat FC225MJ3LT3_L1_Mort-Illumina_1.fq.gz FC225MJ3LT3_L1_Mort-
Illumina_2.fq.gz > Mortino-Illumina_Combined1y2.fq.gz

jellyfish count -t 8 -m 21 -o Mortino_Illumina.jf -c Mortino-
Illumina_Combined1y2.fq.gz

jellyfish histo -t 10 Mortino_Illumina.jf > Mortino_df.histo
```

k-mer profile visualization

GenomeScope v2.0 (RRID:SCR_017014)

```
http://qb.cshl.edu/genomescope/
```

De novo Genome Assembly and Polishing

6 **Assembly**

SMARTdenovo v1.0.0 (RRID:SCR_017622)

```
smartdenovo.pl -p SdN_Assembly_Mortino -c 1
Porechop_Mortino_ONT_q7_1000.fastq > SdN_Assembly_Mortino.mak

make -f SdN_Assembly_Mortino.mak
```

Flye v2.9.2 (RRID:SCR_017016)

```
flye --nano-raw Porechop_Mortino_ONT_q7_1000.fastq --out-dir
Flye_Mortino_Assembly -g 0.5g
```

MaSuRCA v.4.1.0 (RRID:SCR_010691)

```
masurca -i FC225MJ3LT3_L1_Mort-Illumina_1.fq.gz,FC225MJ3LT3_L1_Mort-
Illumina_2.fq.gz -r Porechop_Mortino_ONT_q7_1000.fastq -t 5
```

7 **Polishing**

Medaka v1.11.1

```
medaka_consensus -i Assembly.fasta -o Medaka_Assembly -t 4  -m
r941_min_fast_g507
```

POLCA (MaSuRCA, v4.1.0 RRID:SCR_010691)

```
polca.sh -a Assembly.fasta -r FC225MJ3LT3_L1_Mort-Illumina_1.fq.gz -r
FC225MJ3LT3_L1_Mort-Illumina_2.fq.gz -t 5
```

Genome assembly quality, continuity, and completeness assessment

8 **Quast v5.2.0 (RRID:SCR_001228)**

```
quast.py Assembly.fasta -r Vaccinium_myrtillus_GCA_016920895.1_VacMyr_v1.0_genomic.fna.gz' -o Quast_ref_Vmyrtillus
```

9 **BUSCO v5.4.7 (RRID:SCR_015008)**

```
busco -i Assembly.fasta -l eudicots_odb10 -o BUSCO_Assembly -m genome
```

10 **Long Terminal Repeat (LTR) Assembly Index (LAI)**

LTRharvest v1.6.2 (RRID:SCR_018970)

1. Create an "Enhanced Suffix Array"

```
gt suffixerator -db Assembly.fasta -indexname Assembly_indextable.fsa -tis -suf -lcp -des -ssp -sds -dna
```

2. Run LTRHARVEST

```
gt ltrharvest -index Assembly_indextable.fsa -minlenltr 100 -maxlenltr 7000 -mintsd 4 -maxtsd 6 -motif TGCA -motifmis 1 -similar 85 -vic 10 -seed 20 -seqids yes > Assembly.fa.harvest.scn
```

LTR_FINDER v1.0.7 (RRID:SCR_015247)

```
ltr_finder assembly.fasta > Assembly_LTRfinder.txt
```

Combine both documents from LTRFinder and LTRHarvest

```
cat Assembly_LTRfinder.txt Assembly.fa.harvest.scn > Assembly.fa.rawLTR.scn
```

LTR_retriever v2.8.7 (RRID:SCR_017623)

```
LTR_retriever -genome Assembly.fasta -inharvest Assembly.fa.rawLTR.scn -threads 10
```

11 **Coverage Graphs of ONT and Illumina Reads**

1. Change contig names from assembly file

```
awk '/^>/{print ">Mortino" ++i; next}{print}' < Assembly.fasta > Assembly_nom.fasta
```

2. Identify longest contig

```
cat Assembly_nom.fasta | bioawk -c fastx '{ print length($seq), $name }' | sort -k1,1rn | head -1
```

3. Extract the longest contig from the assembly
Samtools package v1.18 (RRID:SCR_002105)

```
samtools faidx Assembly_nom.fasta Name_Longest_Contig > Longest_contig.fasta
```

4. Extract 10% of ONT and Illumina reads (total reads were obtained from previously run Nanoplot)
BWA (RRID:SCR_010910)

```
reformat.sh in1=FC225MJ3LT3_L1_Mort-Illumina_1.fq.gz
in2=FC225MJ3LT3_L1_Mort-Illumina_2.fq.gz out1=FC225MJ3LT3_L1_Mort-
Illumina_1_subset_10.fq.gz out2=FC225MJ3LT3_L1_Mort-
Illumina_2_subset_10.fq.gz reads=# of total reads samplereadstarget=# of 10%
reads
```

5. Map ONT reads and illumina reads, seperately, to the assembly

ONT reads: minimap2 v2.26 (RRID:SCR_018550)

```
minimap2 -ax map-ont Longest_contig.fa Porechop_Mortino_ONT_q7_1000.fastq >
ONT_Assembly.sam
```

Illumina reads: BWA (RRID:SCR_010910)

```
bwa index Longest_contig.fa

bwa mem Longest_contig.fa FC225MJ3LT3_L1_Mort-Illumina_1_subset_10.fq
FC225MJ3LT3_L1_Mort-Illumina_2_subset_10.fq > Illumina_assembly.sam
```

6. Convert sam files to bam

```
samtools view -bS ONT_Assembly.sam > ONT_Assembly.bam

samtools view -bS Illumina_Assembly.sam > Ilumina_Assembly.bam
```

7. Sort bam files

```
samtools sort ONT_Assembly.bam > ONT_Assembly_sorted.bam

samtools sort Illumina_Assembly.bam > Illumina_Assembly_sorted.bam
```

8. Coverage depth

Samtools package v1.18 (RRID:SCR_002105)

```
samtools depth ONT_Assembly_sorted.bam > depth_Assembly_ONT

samtools depth Illumina_Assembly_sorted.bam > depth_Assembly_Illumina
```

5. Graphs: script developed by López et al., 2023

```
python 3

import matplotlib.pyplot as plt
import pandas as pd
df = pd.read_csv('depth_Assembly_sorted', sep="\t", names=["Sequence",
"Base_position", "Depth"], index_col=0)
fig=df.plot(x='Base_position', y='Depth', kind='line', figsize=(16,
4)).get_figure()
fig.savefig("Coverage_plot.pdf")
```

Genome scaffolding

12 ntLink v1.3.9

```
ntLink scaffold gap_fill target=Assembly.fasta
reads=Porechop_Mortino_ONT_q7_1000.fast overlap=true
```

Removing Foreign Contamination

13 python3 ./fcs.py screen Assembly.fasta –fasta ./fcsgx_test.fa.gz –out-dir ./gx_out/ –gx-db "$GXDB_LOC/gxdb" –tax-id 6973

Genome Annotation

14 **RepeatModeler v2.0.3 (RRID:SCR_015027)**

```
BuildDatabase -name Assembly.fasta RepeatModeler -threads 32 -database
Mortino_genome -LTRStruct >& repeatmodeler.log
```

15 **Change Contig Names in Assembly File**

```
 awk '/^>/{print ">Mortino" ++i; next}{print}' Assembly_nom.fasta
```

16 **Generate Executable Files**

```
maker -CTL
```

17 **1. Modify Maker_opts.ctl file**

2. Run 10 iterations of the first run of maker

```
maker
```

3. Run and train SNAP script

```
sh Snap_Pult_Creator.sh
```

4. Run 5 iterations of the second run of maker

```
maker
```

5. Rerun and train SNAP script

```
sh Snap_Pult_Creator.sh
```

4. Run 5 iterations of the third run of maker

```
maker
```

18 **Generating and Filtering Final Files**

1. Generate single gff and fasta protein and transcript files from all 3 maker rounds

```
gff3_merge -d Assembly_master_datastore_index.log -o Mortino_All.gff

fasta_merge -d Assembly_master_datastore_index.log -o Mortino_All
```

2. Identify conserved protein regions in predicted gene models
InterProScan v5.64-9.60 (RRID:SCR_005829)

```
interproscan.sh -appl PfamA -iprlookup -goterms -f tsv -i
Mortino.all.maker.proteins.fasta
```

3. Modify the original gff3 file by identifying gene models with conserved protein domains

```
ipr_update_gff Mortino_All.gff Mortino.all.maker.proteins.fasta.tsv >
Mortino_genomic_updated.all.gff
```

4. Eliminate gene models with AED <1

```
quality_filter -s Mortino_updated.all.gff > Mortino_FINAL.all.gff
```

5. Calculate annotation statistics with AGAT

AGAT (Another Gff Analysis Toolkit) v1.2.0

```
agat_sp_statistics.pl –gff Mortino_FINAL.all.gff -o Mortino_Stats
```

6. Filter out gene models with no conserved protein regions and AED <1 from protein and transcript fasta files

```
perl -lne 'print $1 if /\tmRNA\t.+ID=([^;]+).+_AED=(.+?);/'
Mortino_FINAL.all.gff > genes_from_gff.aed-1.0.ids

fastaqual_select.pl -f Mortino.all.maker.proteins.fasta -inc
genes_from_gff.aed-1.0.ids > Mortino_Proteins_Final.fasta

fastaqual_select.pl -f Mono_Anotado_All.all.maker.transcripts.fasta -inc
genes_from_gff.aed-1.0.ids > Mono_Anotado_All_Transcripts_Final.fasta
```