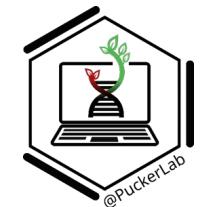
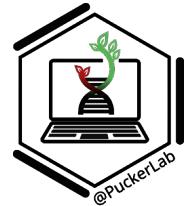
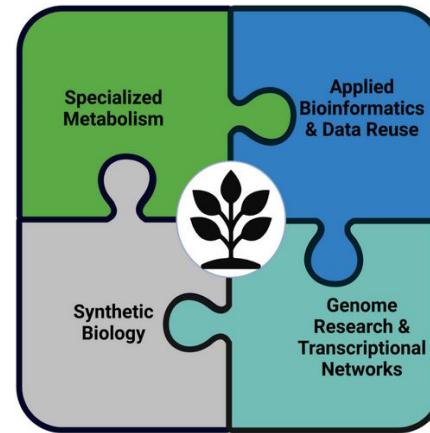


de.NBI course 1: Plant Genome Sequence Assembly and Annotation

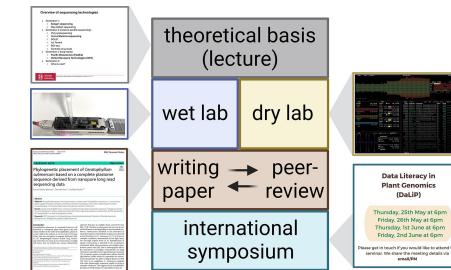




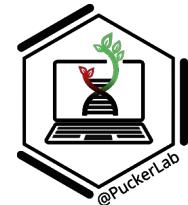
- Flavonoids: Anthocyanins, Flavonols, Flavones, Proanthocyanidins
- Withanolides
- Carotenoids
- ...



Tools: KIPES, NAVIP, MGSE, MYB/bHLH_annotator, DupyliCate...



Availability of Slides



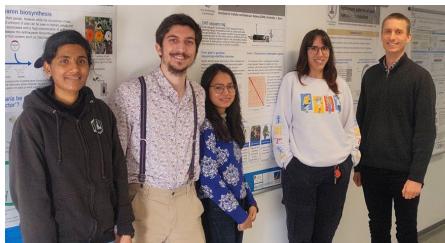
All materials will be shared with participants at the end of this course. Background information is available here:

<https://www.preprints.org/manuscript/202508.1176>

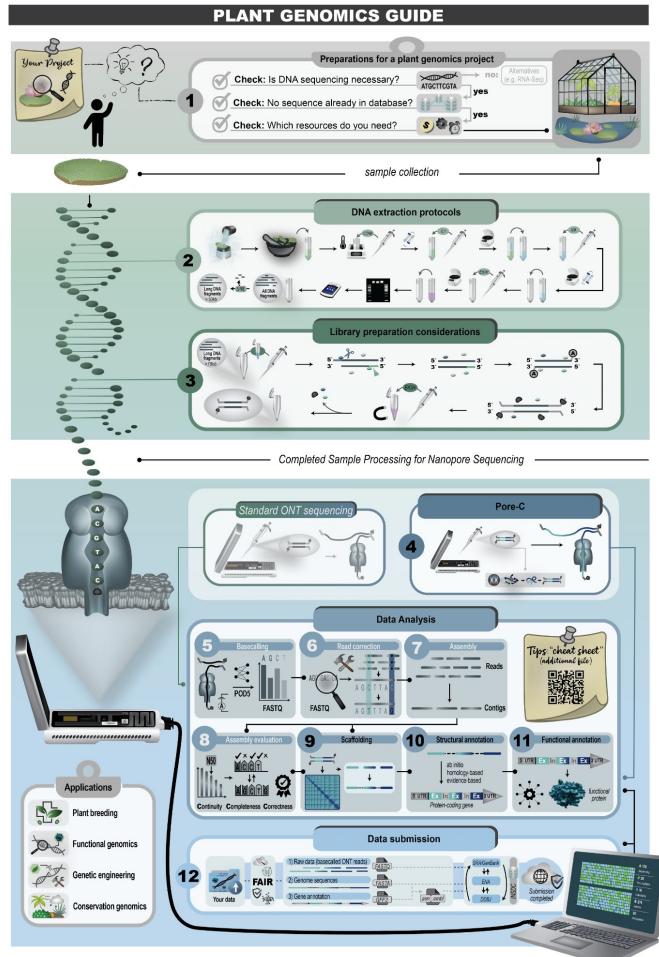


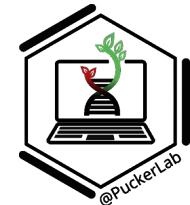
Plant Genome Sequence Assembly and Annotation

- Plant genomics
- Long read sequencing (ONT)
- Read acquisition
- Genome assembly & evaluation
- Structural annotation



From L to R: Shakunthala, Samuel, Nancy, Julie, and Boas
Picture credits: Jakob Horz



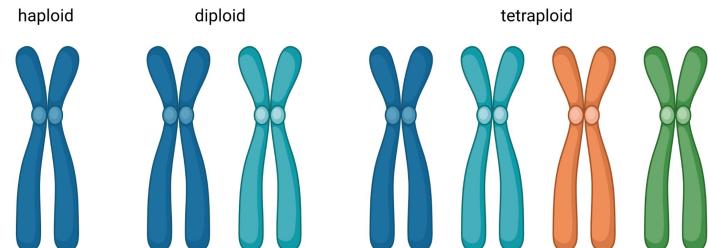


- Genome size: variation from 65 Mbp to 149 Gbp
- Ploidy: haploid/diploid genomes are much easier to analyze than polyploid genomes
- Computational costs increase with genome size

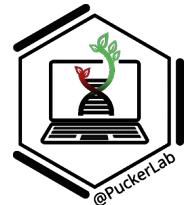


Genlisea violacea
Utricularia gibba
Arabidopsis thaliana
Dioscorea dumetorum
Beta vulgaris
Vitis vinifera
Sequoia giganteum
Paris japonica

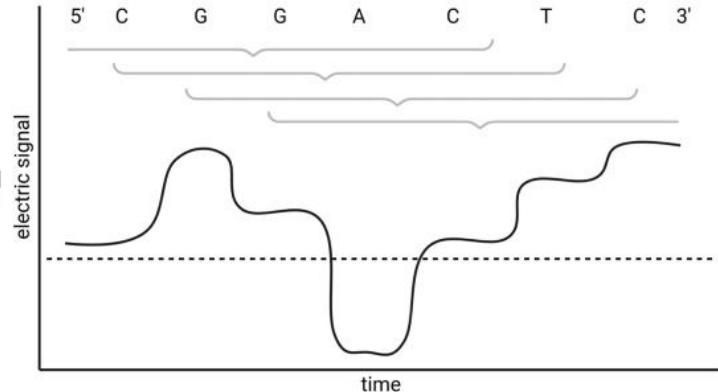
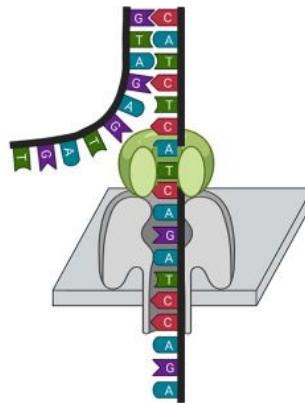
Genlisea violacea: https://commons.wikimedia.org/wiki/File:Genlisea_violacea_giant.jpg
Utricularia gibba: https://commons.wikimedia.org/wiki/File:Utricularia_gibba_flower_01.jpg
Arabidopsis thaliana: https://commons.wikimedia.org/wiki/File:Arabidopsis_thaliana_s10.jpg
Dioscorea dumetorum: https://commons.wikimedia.org/wiki/File:Yamsa_BrittonMarie_1.jpg
Beta vulgaris: <https://de.wikipedia.org/wiki/Zucker%C3%BCbe#/media/Datei:Zucker%C3%BCbe.jpg>
Vitis vinifera: https://commons.wikimedia.org/wiki/File:Vitis_vinifera_20120823.jpg
Sequoia giganteum: https://commons.wikimedia.org/wiki/File:Grizzly_Giant_Mariposa_Grove.jpg
Paris japonica: https://commons.wikimedia.org/wiki/File:Paris_japonica_Kinugasou_in_Hakusan_2003_7_27.jpg

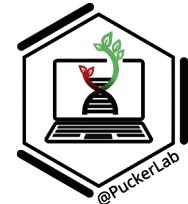


Long read sequencing

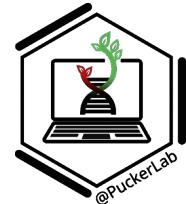
*Aquilegia vulgaris**Urtica dioica**Nymphaea spec.**Theobroma cacao**Tacca chantrieri**Corylus avellana*

Solanaceae species

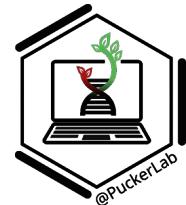




- high performance computing => server farm

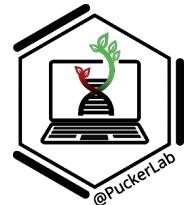


- high performance computing => server farm
- de.NBI offers VMs (VM = virtual machine)
 - running Linux
 - access through SSH
- de.NBI connected LifeScience AAI Account needed

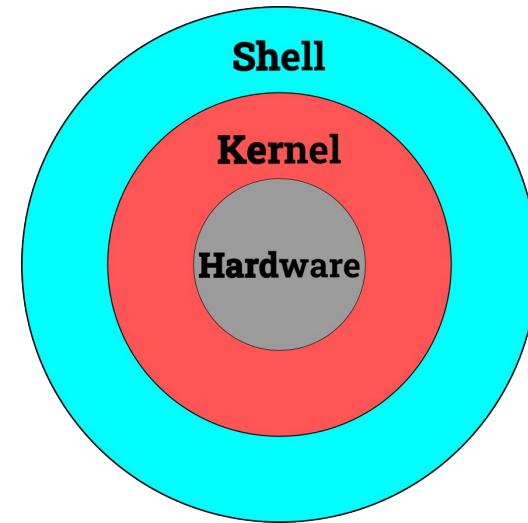


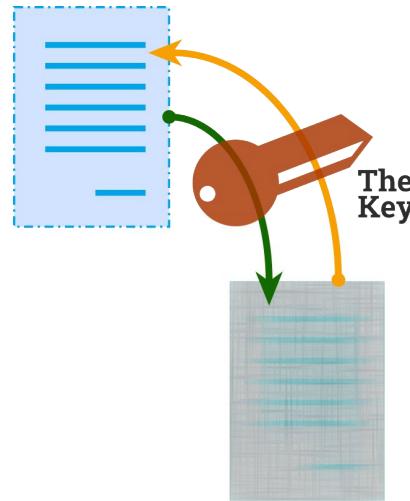
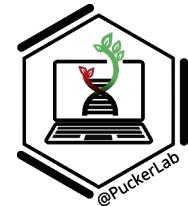
- high performance computing => server farm
- de.NBI offers VMs (VM = virtual machine)
 - running Linux
 - access through SSH
- de.NBI connected LifeScience AAI Account needed
- Tasks:
 1. Register: <https://cloud.denbi.de/wiki/registration/>
 2. Apply for the course de.NBI project:
<https://signup.aai.lifescience-ri.eu/fed/registrar/?vo=denbi&group=pbbcourse20251>



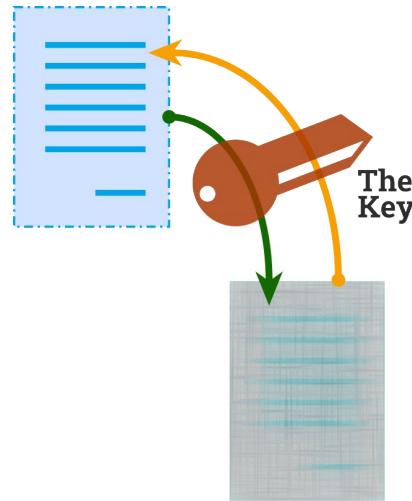
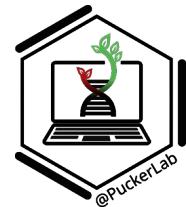


- SSH = Secure Shell
- Linux is a kernel
- shell \approx Interface to the Computer
- CLI = command line interface

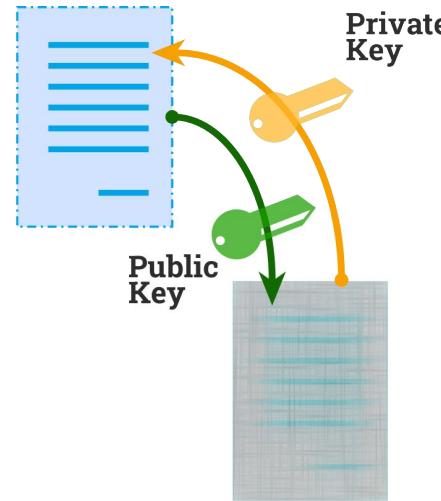




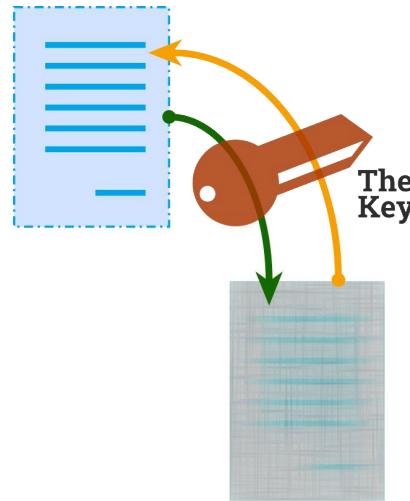
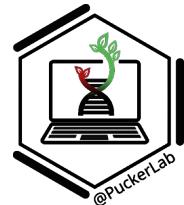
Symmetric encryption



Symmetric encryption



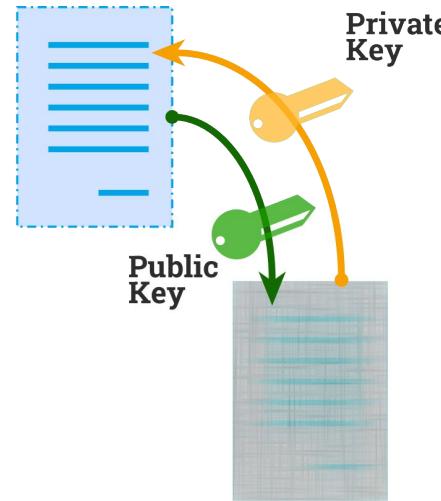
Asymmetric encryption



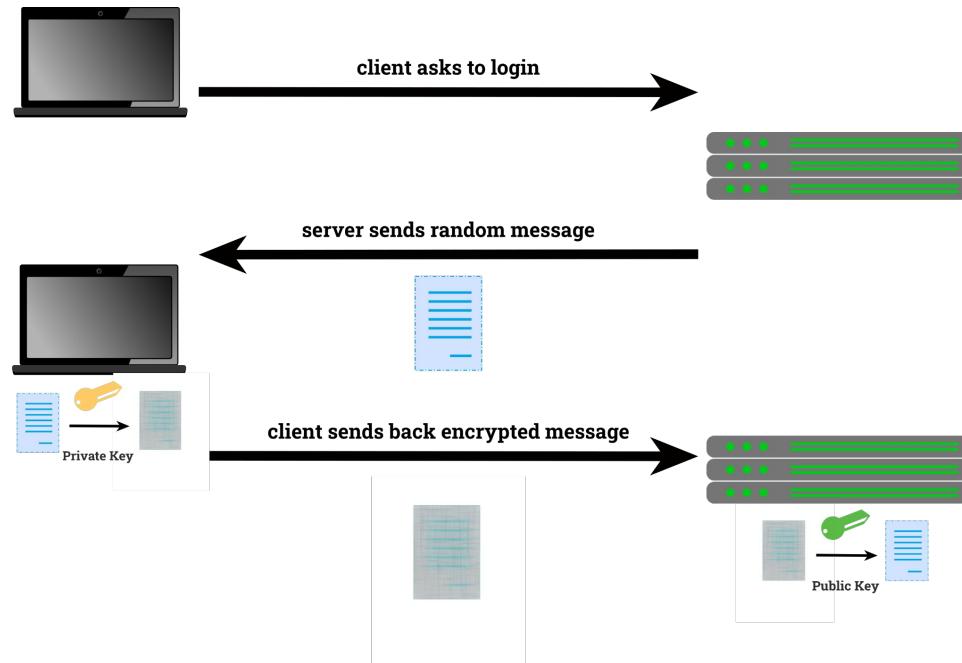
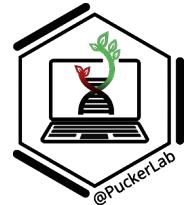
Symmetric encryption

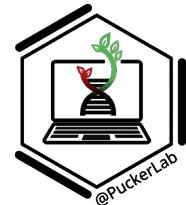
ecdsa-sha2-nistp256

AAAAAE2VjZHNhLXNoYTItbmlzdHAYNTYAAAAlbmlzdHAYNTYAAABBBJHT0aW+Q2bjkfVM7kN2ucaf8eltoalV6V0CDBgpWF+dKV1HKvGWWfjtYMOdLDUKIAzJc5Q6bKs6bkYiqqtUQ=



Asymmetric encryption

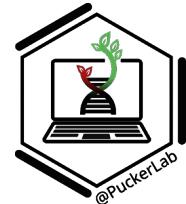




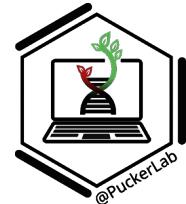
- high performance computing => server farm
- de.NBI offers VMs (VM = virtual machine)
 - running Linux
 - access through SSH
- de.NBI connected LifeScience AAI Account needed
- Tasks:
 1. Register: <https://cloud.denbi.de/wiki/registration/>
 2. Apply for the course de.NBI project:
<https://signup.aai.lifescience-ri.eu/fed/registrar/?vo=denbi&group=pbbcourse20251>
 3. create SSH key pair in de.NBI cloud portal and confirm unix user name



Groups of two for VM creation



- We will create the VMs
 - tell us your group members names
- Tasks:
 1. open a Terminal and login via ssh



- SSH key might need the correct permissions
- needed information:
 - username
 - ip address/domain
 - port number (usually optional but mandatory for de.NBI VMs)
 - ssh private key
 - exemplary commands:

```
ssh -i /path/to/key username@ip-address -p port-number
```

```
ssh -i C:\Users\User\de.NBI_course\key.txt ubuntu@134.176.27.78 -p 23457
```

CLI - basic commands

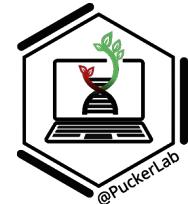
```
sam@dhcp068: ~$ ssh -i /home/TUBS_NC/13.DenBiCloud_access_scripts/meckoni_ecdsa.txt ubuntu@134.176.27.78 -p 30235
[...]
de.NBI cloud
http://cloud.denbi.de
cloud@denbi.de

Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-87-generic x86_64)

System information as of Sun Dec  7 14:28:58 UTC 2025

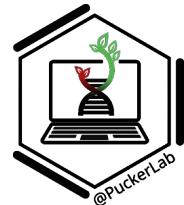
System load:  0.14      Processes:          422
Usage of /:  13.5% of 18.33GB  Users logged in:    0
Memory usage: 0%           IPv4 address for ens3: 192.168.0.235
Swap usage:  0%

Last login: Sun Dec  7 14:18:21 2025 from 192.168.0.143
ubuntu@snmtestdeletesoon-9b08e: ~$
```



```
ubuntu@snmtestdeletesoon-9b08e: ~$ cd /vol/data/
ubuntu@snmtestdeletesoon-9b08e:/vol/data$ █
```

CLI - basic commands



```
sam@dhcp068: ~$ ssh -i /home/TUBS_NC/13.DenBiCloud_access_scripts/meckoni_ecdsa.txt ubuntu@134.176.27.78 -p 30235
[de.NBI cloud
http://cloud.denbi.de
cloud@denbi.de

Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-87-generic x86_64)

System information as of Sun Dec 7 14:28:58 UTC 2025

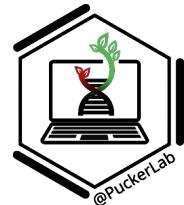
System load: 0.14      Processes:        422
Usage of /: 13.5% of 18.33GB  Users logged in:    0
Memory usage: 0%          IPv4 address for ens3: 192.168.0.235
Swap usage:  0%

Last login: Sun Dec 7 14:18:21 2025 from 192.168.0.143
ubuntu@snmtestdeletesoon-9b08e: $
```

```
ubuntu@snmtestdeletesoon-9b08e: ~$ cd /vol/data/
ubuntu@snmtestdeletesoon-9b08e:/vol/data$
```



[https://github.com/bpucker/teaching
tree/master/deNBI2025](https://github.com/bpucker/teaching/tree/master/deNBI2025)



```
sam@dhcp068: ~$ ssh -i /home/TUBS_NC/13.DenBiCloud_access_scripts/meckoni_ecdsa.txt ubuntu@134.176.27.78 -p 30235
[de.NBI cloud
http://cloud.denbi.de
cloud@denbi.de

Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-87-generic x86_64)

System information as of Sun Dec  7 14:28:58 UTC 2025

System load:  0.14      Processes:          422
Usage of /:   13.5% of 18.33GB  Users logged in:    0
Memory usage: 0%
IPv4 address for ens3: 192.168.0.235
Swap usage:   0%

Last login: Sun Dec  7 14:18:21 2025 from 192.168.0.143
ubuntu@snmtestdeletesoon-9b08e: $
```

```
ubuntu@snmtestdeletesoon-9b08e: ~$ cd /vol/data/
ubuntu@snmtestdeletesoon-9b08e:/vol/data$
```

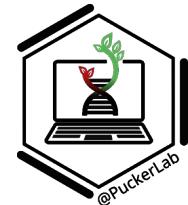
- Tasks:

1. create your own folder in the 1 TB volume



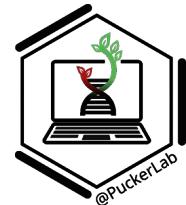
[https://github.com/bpucker/teaching
tree/master/deNBI2025](https://github.com/bpucker/teaching/tree/master/deNBI2025)

1. Downloading DNA reads



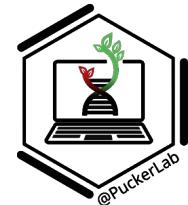
- SRA = Sequence Read Archive
- downloading tools:
 1. prefetch (compressed file)
 2. fasterq-dump (un- compresses the files)
- December 2, 2025: **SRA Toolkit** Release 3.3.0 -> **GitHub** repository (apt version is outdated)

1. Downloading DNA reads



- SRA = Sequence Read Archive
- downloading tools:
 1. prefetch (compressed file)
 2. fasterq-dump (un-compresses the files)
- December 2, 2025: **SRA Toolkit** Release 3.3.0 -> **GitHub** repository (apt version is outdated)
- Tasks:
 1. install the tools
 2. get the reads with the SRA/ENA ID: ERR4760286

1. Downloading DNA reads



- SRA = Sequence Read Archive
- downloading tools:
 1. prefetch (compressed file)
 2. fasterq-dump (un-compresses the files)
- December 2, 2025: **SRA Toolkit** Release 3.3.0 -> **github** repository (apt version is outdated)

SRA/ENA ID: **ERR4760286**

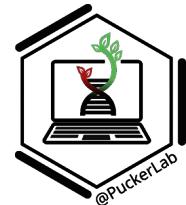
```
 wget https://ftp-trace.ncbi.nlm.nih.gov/sra/sdk/3.3.0/sratoolkit.3.3.0-ubuntu64.tar.gz
 tar -xvf sratoolkit.3.3.0-ubuntu64.tar.gz
```

```
 prefetch -p --max-size 250g --output-directory /vol/data/DNA_reads ERR4760286
```

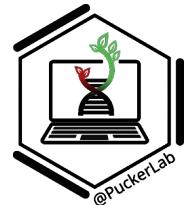
no. of threads (CPU)

SRA/ENA read ID

```
fasterq-dump -e 15 /vol/data/DNA_reads/ERR4760286 --outdir /vol/data/DNA_reads/raw_reads
```

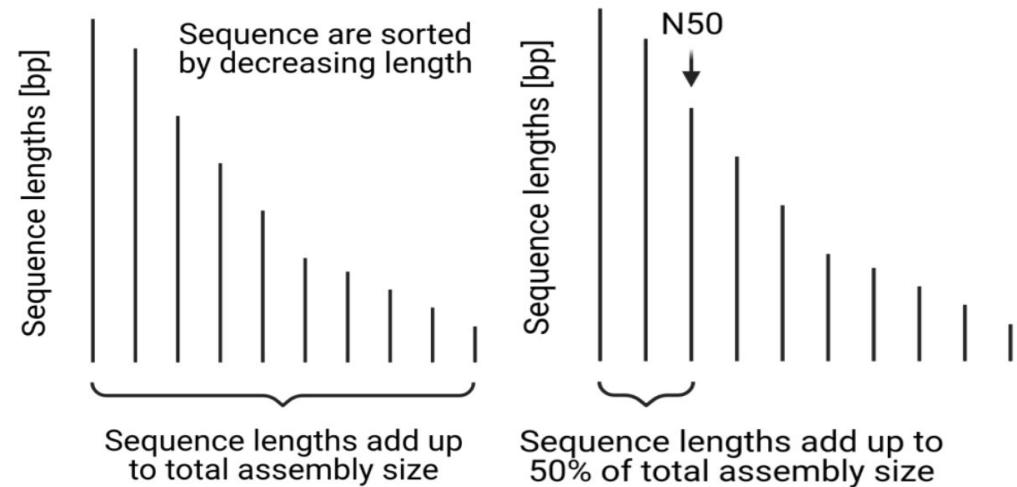


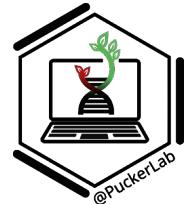
- Numerical summaries that describe the quality, length, composition, and coverage of your reads
 - Number of reads
 - Total number of bases
 - GC content
 - Read length
 - N50
-
- Important for downstream analysis



- The N50 of a set of reads is the length of the shortest read among the longest reads that together make up 50% of the total bases sequenced.

- Take all read lengths and sort them from longest to shortest;
- Sum the total number of bases across all reads to get the total yield;
- Starting from the longest read, cumulatively add read lengths until the sum reaches at least 50% of the total bases;
- The length of the last read added is the N50.

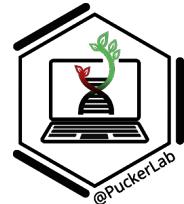




- FastQC/MultiQC: short reads
- NanoPlot, NanoComp: for ONT long reads
- Seqtk, Seqkit: generally for any sequence

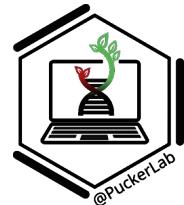
- Custom scripts:

⭐ FASTQ_stats3.py ⭐



<https://github.com/bpucker/PlantGenomicsGuide/>

 bpucker	reference updated	9eac6ed · 2 days ago	 23 Commits
 Arabidopsis_thaliana_araport11.anno.txt	Req. for construct_anno.py	4 months ago	
 Arabidopsis_thaliana_araport11.pep.fasta	Req. for construct_anno.py	4 months ago	
 FASTQ_stats3.py 	v0.42	2 days ago	
 LICENSE	Initial commit	5 months ago	
 README.md	Update README.md	last month	
 clean_fastq_headers.py	Update clean_fastq_headers.py	4 months ago	
 construct_anno.py	Create construct_anno.py	4 months ago	
 contig_stats3.py	reference updated	2 days ago	
 match_proteins.py	Create match_proteins.py	4 months ago	

[PlantGenomicsGuide / FASTQ_stats3.py](#)

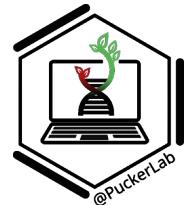
bpucker v0.42

9115e56 · 2 days ago

History

[Code](#) [Blame](#) 268 lines (214 loc) · 8.37 KB

```
1  ### Boas Pucker ###
2  ### pucker@uni-bonn.de ###
3
4  __version__ = "v0.42"
5
6
7  __usage__ = """
8      calculation of FASTQ statististics ("""+__version__+"""");
9
10     python3 FASTQ_stats3.py
11     --in <FULL_PATH_TO_FASTQ_FILE> |    --in_dir <FULL_PATH_TO_DIRECTORY>
12
13     optional:
14     --rfig <READ_LEN_HIST_PICTURE_FILE>
15     --cutoff <READ_LEN_CUTOFF_FOR_PLOT_IN_KB>[100]
16     --qfig <QUALITY_VS_READ_LEN_PICTURE_FILE>
17     --lenct <UPPER_READ_LENGTH_CUTOFF_FOR_PLOT_IN_KB>[200]
18     --qualct <UPPER_QUAL_CUTOFF_FOR_PLOT>[40]
19     --quality <ACTIVATES_QUALITY_ANALYSIS>
20
21     bug reports and feature requests: pucker@uni-bonn.de
22     """
```



```
wget https://raw.githubusercontent.com/bpucker/PlantGenomicsGuide/refs/heads/main/FASTQ_stats3.py
```

```
← → ⌂  https://raw.githubusercontent.com/bpucker/PlantGenomicsGuide/refs/heads/main/FASTQ_stats3.py
```

```
### Boas Pucker ###
### puckер@uni-bonn.de ###

__version__ = "v0.42"

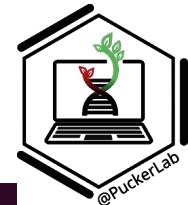
__usage__ = """
    Calculation of FASTQ statististics (""+__version__+""):

    python3 FASTQ_stats3.py
    -in <FULL_PATH_TO_FASTQ_FILE> | --in_dir <FULL_PATH_TO_DIRECTORY>

    optional:
    --rfig <READ_LEN_HIST FIGURE_FILE>
    --cutoff <READ_LEN_CUTOFF_FOR_PLOT_IN_KB>[100]
    -qfig <QUALITY_VS_READ_LEN FIGURE_FILE>
    --lencut <UPPER_READ_LENGTH_CUTOFF_FOR_PLOT_IN_KB>[200]
    --qualcut <UPPER_QUAL_CUTOFF_FOR_PLOT>[40]
    --quality <ACTIVATES_QUALITY_ANALYSIS>

    bug reports and feature requests: puckер@uni-bonn.de
"""
```

★ FASTQ_stats3.py ★



```
ubuntu@gossypiumhirsutum-ed98e:/vol/data/scripts$ wget https://raw.githubusercontent.com/bpucker/PlantGenomicsGuide/refs/heads/main/FASTQ_stats3.py
--2025-12-04 11:05:53-- https://raw.githubusercontent.com/bpucker/PlantGenomicsGuide/refs/heads/main/FASTQ_stats3.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.111.133, 185.199.108.133, 185.199.109.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 8569 (8.4K) [text/plain]
Saving to: 'FASTQ_stats3.py'

FASTQ_stats3.py          100%[=====]  8.37K  ---KB/s   in 0s

2025-12-04 11:05:53 (76.0 MB/s) - 'FASTQ_stats3.py' saved [8569/8569]

ubuntu@gossypiumhirsutum-ed98e:/vol/data/scripts$ ls
FASTQ_stats3.py  run_fastq_dump_parallel.sh
ubuntu@gossypiumhirsutum-ed98e:/vol/data/scripts$ python3 FASTQ_stats3.py -h
WARNING: matplotlib import failed - figure plotting not possible.

      Calculation of FASTQ statistics (v0.42):

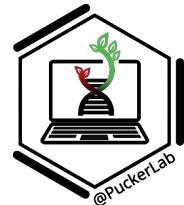
python3 FASTQ_stats3.py
--in <FULL_PATH_TO_FASTQ_FILE> |           --in_dir <FULL_PATH_TO_DIRECTORY>

optional:
--rfig <READ_LEN_HIST_PICTURE_FILE>
--cutoff <READ_LEN_CUTOFF_FOR_PLOT_IN_KB>[100]
--qfig <QUALITY_VS_READ_LEN_PICTURE_FILE>
--lencut <UPPER_READ_LENGTH_CUTOFF_FOR_PLOT_IN_KB>[200]
--qualcut <UPPER_QUAL_CUTOFF_FOR_PLOT>[40]
--quality <ACTIVATES_QUALITY_ANALYSIS>

      bug reports and feature requests: pucker@uni-bonn.de
```

Dataset: ERR4760286

```
ubuntu@gossypiumhirsutum-ed98e:/vol/data/Final_dataset_deNBI/Read$ cd /vol/data/scripts/
ubuntu@gossypiumhirsutum-ed98e:/vol/data/scripts$ python3 FASTQ_stats3.py --in /vol/data/Final_dataset_deNBI/Read/ERR4760286_pass.fastq
WARNING: matplotlib import failed - figure plotting not possible.
/vol/data/Final_dataset_deNBI/Read/ERR4760286_pass.fastq
total number of nucleotides:    8.812031342 Gbp
N50: 123311
number of reads: 69689
average read length:    126447.95221627517
GC content:      0.37114296001331676
```

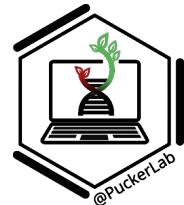


<https://github.com/nanoporetech/dorado>

- Haplotype-aware ERRor cOrrection
 - Highly accurate
 - Haplotype-aware
 - Deep-learning tool
 - Error correction of Nanopore Reads

> 64 CPU cores
> 256GB GB RAM
> 32GB GPU VRAM

Input: FASTQ(.gz)
Output: FASTA

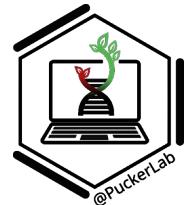


Step 1: merge all FASTQs of different runs from the same species

```
cat \  
/path/to/your/gzip/fastq/file/run1.mod.fastq.gz \  
/path/to/your/gzip/fastq/file/run2.mod.fastq.gz \  
/path/to/your/gzip/fastq/file/run3.mod.fastq.gz \  
> /path/to/your/file/thaliana.fastq.gz &
```

Step 2: CPU heavy part of HERRO

```
/vol/data/tools/dorado-1.3.0-linux-x64/bin/dorado \  
correct \  
/vol/data/your/folder/thaliana.fastq.gz \  
--to-paf \  
--threads 200 \  
> /vol/data/your/folder/thaliana.overlaps.paf \  
2> /vol/data/your/folder/thaliana.overlaps.doc.txt &
```

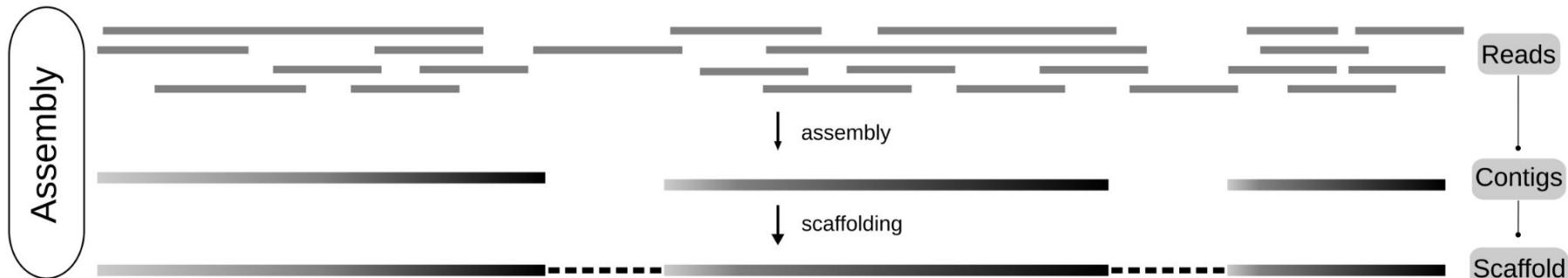
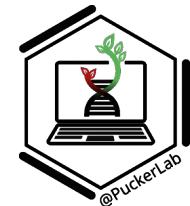


Step 3: Decompress the FASTQ file

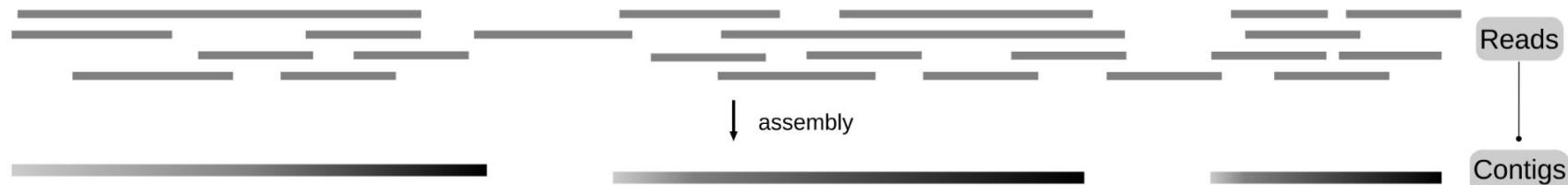
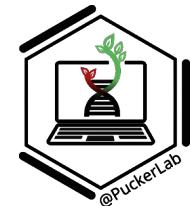
```
gunzip /vol/data/herro/thaliana.fastq.gz
```

Step 4: GPU heavy part of HERRO

```
/vol/data/tools/dorado-1.3.0-linux-x64/bin/dorado \
correct \
/vol/data/herro/thaliana.fastq \
--from-paf /vol/data/herro/thaliana.overlaps.paf \
> /vol/data/herro/thaliana.corrected_reads.fasta \
2> /vol/data/herro/thaliana.corrected_reads.errors.txt &
```

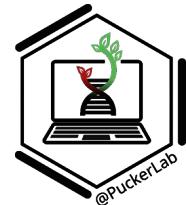


- Computational process to piece together reads to form a complete representation of an organism's genome
- Reads are shorter than the chromosome - even long reads
- Genome = DNA in a cell
- Genome sequence = representation of DNA in the cell



- Connecting reads - identify overlap between reads
- Reads -> Contigs (contiguous sequences)

Assemblers	Pros	Cons
Shasta	Specialized for ONT reads; Exceptional speed	Based on read quality, may produce fragmented or less contiguous sequences Smaller than expected genome sizes
NextDenovo	Superior error correction of raw reads (especially ONT); highly contiguous and accurate assemblies for ONT R9 data	Not optimized for recent, more accurate R10 data
Hifiasm	Leading assembler for diploid genomes using accurate reads (PacBio HiFi/ ONT R10 reads)	Does not work with ONT R9 data
Verkko	Works really well in hybrid approach - combining ONT reads with PacBio HiFi	Perform suboptimally on ONT-only moderate coverage reads



Before, the assembly process -

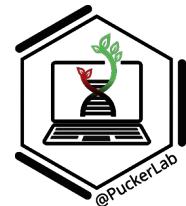
A. Genome size estimation

- Read file - ERR4760286.fastq → ***Arabidopsis thaliana* Col-0**
- Estimated genome size? <https://cvalues.science.kew.org/search/angiosperm>

Genus	Species	Subspecies	DNA Amount 1C (pg)	Original Reference
Arabidopsis	thaliana	ecotype Columbia	0.16	Bennett et al.,2003

- 1C = amount of DNA in a haploid set of chromosomes
- 1 picogram (pg) = 978 megabases (Mb)
- Estimated genome size for *A. thaliana* Col = $0.16 \text{ pg} \times 978 \approx 157 \text{ Mb}$

(What if C-value database does not have your species of interest?)



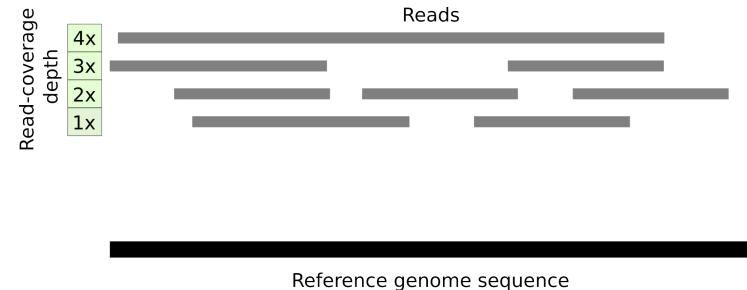
Before, the assembly process -

B. Selecting the right assembler

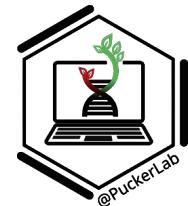
$$\text{Coverage} = \frac{N \times L}{G}, \text{ where } N = \text{number of reads}$$

L = average read length

G = estimated genome size



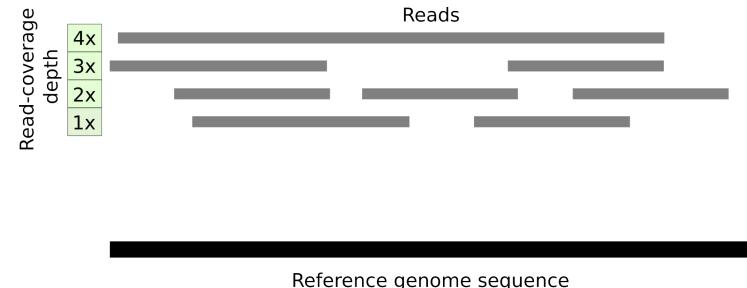
Assemblers	Pros	Cons
Shasta	Specialized for ONT reads; Exceptional speed	Based on read quality, may produce fragmented or less contiguous sequences Smaller than expected genome sizes
NextDenovo	Superior error correction of raw reads (especially ONT); highly contiguous and accurate assemblies for ONT R9 data	Not optimized for recent, more accurate R10 data
Hifiasm	Leading assembler for diploid genomes using accurate reads (PacBio HiFi/ ONT R10 reads)	Does not work with ONT R9 data
Verkko	Works really well in hybrid approach - combining ONT reads with PacBio HiFi	Perform suboptimally on ONT-only moderate coverage reads



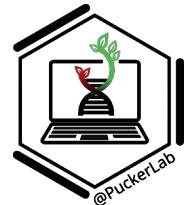
Before, the assembly process -

B. Selecting the right assembler

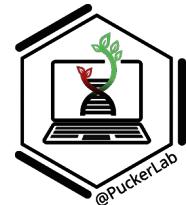
Coverage = **56 x**



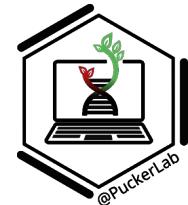
Assemblers	Pros	Cons
Shasta	Specialized for ONT reads; Exceptional speed	Based on read quality, may produce fragmented or less contiguous sequences Smaller than expected genome sizes
NextDenovo	Superior error correction of raw reads (especially ONT); highly contiguous and accurate assemblies for ONT R9 data	Not optimized for recent, more accurate R10 data
Hifiasm	Leading assembler for diploid genomes using accurate reads (PacBio HiFi/ ONT R10 reads)	Does not work with ONT R9 data
Verkko	Works really well in hybrid approach - combining ONT reads with PacBio HiFi	Perform suboptimally on ONT-only moderate coverage reads



	Shasta (DatasetA)	NextDenovo2 (DatasetB)
Documentation:	https://paoloshasta.github.io/shasta/	https://nextdenovo.readthedocs.io/en/latest/
Installation:		



	Shasta (DatasetA)	NextDenovo2 (DatasetB)
Documentation:	https://paoloshasta.github.io/shasta/	https://nextdenovo.readthedocs.io/en/latest/
Installation:	<pre>wget https://github.com/paoloshasta/shasta/releases/download /0.14.0/shasta-Linux-X.Y.Z chmod ugo+x shasta-Linux-X.Y.Z nano ~/.bashrc export PATH="/path/to/shasta_executable:\$PATH" source ~/.bashrc ./shasta-Linux-0.14.0 -h</pre>	<pre>sudo apt install pipx pipx ensurepath pipx install paraleltask pipx ensurepath pipx install paraleltask wget https://github.com/Nextomics/NextDenovo/releases/down load/2.5.2/NextDenovo.tgz tar -xvf NextDenovo.tgz && cd NextDenovo</pre>
Command:		



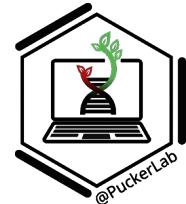
	Shasta (DatasetA)	NextDenovo2 (DatasetB)
Documentation:	https://paoloshasta.github.io/shasta/	https://nextdenovo.readthedocs.io/en/latest/
Installation:	<pre>wget https://github.com/paoloshasta/shasta/releases/download /0.14.0/shasta-Linux-X.Y.Z</pre> <pre>chmod ugo+x shasta-Linux-X.Y.Z</pre> <pre>nano ~/.bashrc export PATH="/path/to/shasta_executable:\$PATH" source ~/.bashrc</pre>	<pre>python3 -m venv mypython source mypython/bin/activate pip3 install paralletask</pre> <pre>wget https://github.com/Nextomics/NextDenovo/releases/down load/2.5.2/NextDenovo.tgz tar -xvf NextDenovo.tgz && cd NextDenovo</pre>
Command:	<pre>shasta-Linux-0.14.0 --config Nanopore-UL-May2022 --input Reads.fastq --AssemblyDirectory Shasta01/ --threads N >> shasta.log 2>&1 &</pre>	<pre>./nextDenovo run.cfg >>nextdenovo.log 2>&1 &</pre>

Calculating the right resources for running NextDenovo:

>run.cfg

```
[General]
job_type = local
job_prefix = nextDenovo
task = all
rewrite = yes
deltmp = yes
parallel_jobs = 4
input_type = raw
read_type = ont
input_fofn = input.fofn
workdir = 01_rundir
[correct_option]
read_cutoff = 1k
genome_size = 157m
sort_options = -m 20g -t 15
minimap2_options_raw = -t 8
pa_correction = 3
correction_options = -p 15

[assemble_option]
minimap2_options_cns = -t 8
nextgraph_options = -a 1
```

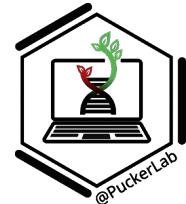


Today you learned:

- Accessing de.NBI VMs via SSH
- Working in a CLI environment
- Obtaining and evaluating sequencing reads
- Assembling a genome sequence

Tomorrow you will learn:

- Evaluating a genome assembly sequence
- Structural annotation of a genome assembly sequence

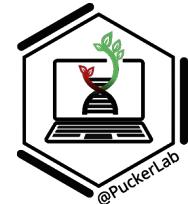


Yesterday you learned:

- Accessing de.NBI VMs via SSH
- Working in a CLI environment
- Obtaining and evaluating sequencing reads
- Assembling a genome sequence

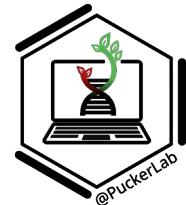
Today you will learn:

- Evaluating a genome assembly sequence
- Structural annotation of a genome assembly sequence



FASTA Format

```
>ctg000000 type:s:linear length:i:316896 node:i:15
cgccgtcctggtcacgaatctctggacccgtcgcatgaagttccctccagacctggtcatttatttcaacttga
>ctg000010 type:s:linear length:i:18291631 node:s:unknown
cctaaacctaaacCCTAACCTAAACCTAAACCTAAACCCCTAAACCCCTAAACCCCTAAACCCCTAAACCC
>ctg000020 type:s:linear length:i:18476100 node:s:unknown
TCCTGCTTCTTCGGCTTCGGAGGGCTGTCTTGGGCTTCCGAAAAGGTATCACATGCCAAGTTGGCCTACGGTCTAA
>ctg000030 type:s:linear length:i:32605930 node:i:1836
tgTAGGATCAATCATTGCATTTCCATAGGATAGATAGGCCGACAAGATCATGGGAAGAAGTCAAATCACATCCGAA
>ctg000040 type:s:linear length:i:11381128 node:i:703
cctaggcctaggcctaggcctgaaacctgaaaggcctaggcctaaggcctgaggcctaggcctgaggcctgaggcctgaa
>ctg000050 type:s:linear length:i:324797 node:s:unknown
aatacatttatttttattgacaagtactcgtaactaaaaagttaaaatttttattcatattttttagtatg
```



Print contigs in the assembly

```
grep ">" assembly.fasta
```

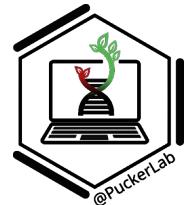
Count number of contigs in the assembly

```
grep -c ">" assembly.fasta
```

```
>ctg000000 type:s:linear length:i:316896 node:i:15  
cgccgtcctggtcacgaatctctggacccgtcgcatgaagttccctccagacctggtcgatttatttcaactttga  
>ctg000010 type:s:linear length:i:18291631 node:s:unknown  
cctaaacctaaacCCTAACCTAAACCTAAACCCCTAAACCCCTAAACCCCTAAACCCCTAAACCCCTAAACC
```

Shorten headers to avoid downstream analysis errors → [clean_genomic_fasta.py](#)

available at : <https://github.com/bpucker/GenomeAssembly>



Shorten headers to avoid downstream analysis errors → **clean_genomic_fasta.py**

available at : <https://github.com/bpucker/GenomeAssembly>

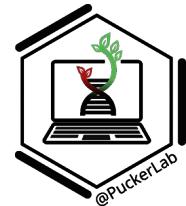
Commands:

```
wget https://raw.githubusercontent.com/bpucker/GenomeAssembly/main/clean\_genomic\_fasta.py
```

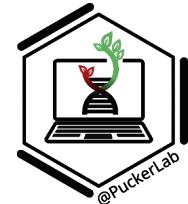
```
python3 clean_genomic_fasta.py
```

```
python3 clean_genomic_fasta.py --in genome.fasta --out cleaned_genome.fasta
```

All next steps using **cleaned_genome.fasta !!**



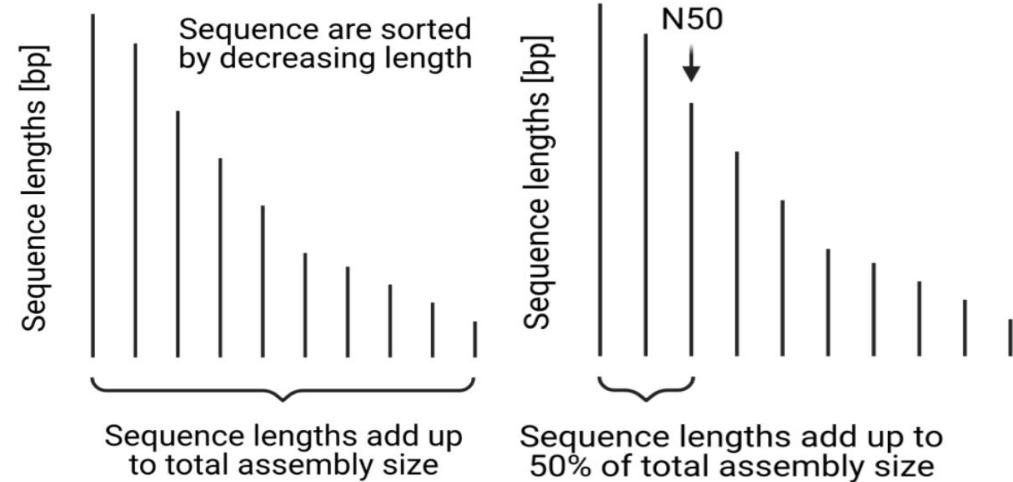
Parameters	DatasetA	DatasetB
No. of contigs		
Genome assembly size		
Minimum contig size		
Maximum contig size		
Contiguity (Contig N50)		
Contiguity (LAI)		
Completeness (BUSCO)		
Correctness (QV - Merqury)		

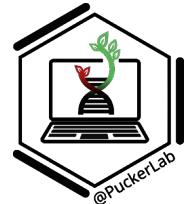


- Quantitative measures describing size, continuity, correctness, completeness of the assembled genome sequence
 - N50
 - Total assembly size
 - Number of contigs
 - GC content
 - Completeness
 - Error rate

- The length of the shortest sequence among the longest sequences that together make up at least 50% of the total assembly length

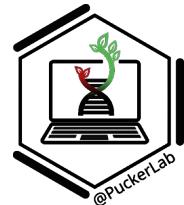
1. List all sequence lengths and sort them from longest to shortest;
2. Add up all sequence lengths to get the total assembly size;
3. Starting from the longest sequence, cumulatively add sequence lengths until the sum reaches at least 50% of the total assembly size;
4. The length of the last sequence added is the N50.





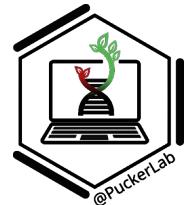
- BUSCO
- Merqury
- LAI

- Custom scripts:
 ★ contig_stats3.py ★



<https://github.com/bpucker/PlantGenomicsGuide/>

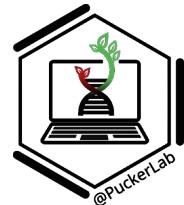
 bpucker	reference updated	9eac6ed · 2 days ago	 23 Commits
 Arabidopsis_thaliana_araport11.anno.txt	Req. for construct_anno.py	4 months ago	
 Arabidopsis_thaliana_araport11.pep.fasta	Req. for construct_anno.py	4 months ago	
 FASTQ_stats3.py	v0.42	2 days ago	
 LICENSE	Initial commit	5 months ago	
 README.md	Update README.md	last month	
 clean_fastq_headers.py	Update clean_fastq_headers.py	4 months ago	
 construct_anno.py	Create construct_anno.py	4 months ago	
 contig_stats3.py	 reference updated	2 days ago	
 match_proteins.py	Create match_proteins.py	4 months ago	

[PlantGenomicsGuide / contig_stats3.py](#) 

...

 bpucker reference updated9eac6ed · 2 days ago [Code](#) [Blame](#) 361 lines (303 loc) · 11.6 KB

```
1  ### Boas Pucker ###
2  ### puckер@uni-bonn.de ####
3
4  ## based on script contig_stats.py Pucker et al., 2016. doi:10.1371/journal.pone.0164321
5
6  import re, sys, os
7  from operator import itemgetter
8
9  # --- end of imports --- #
10
11 __version__ = "v3.1"
12
13 __citation__ = "de Oliveira, J. A. V. S.; Choudhary, N.; Meckoni, S. N.; Nowak, M. S.; Hagedorn, M.; Pucker, B. (2025). Cookbook for Plant Genome Sequences. doi: 10.20944/preprints202508.1176.v2"
14
15 __usage__ = """
16     Calculation of assembly statistics (""" + __version__ + """):
17     python3 contig_stats3.py
18     --in <INPUT_FASTA_FILENAME>
19
20     optional:
21     --min <MIN_CONTIG_LENGTH> [1000]
22     --out <FULL_PATH_TO_OUTPUT_DIRECTORY>
23     --exp <EXPRESSION_FILE(normalized)>
24
25     bug reports and feature requests: puckер@uni-bonn.de
26     Please cite: \n""" + __citation__ + """
27 """
```



```
wget https://raw.githubusercontent.com/bpucker/PlantGenomicsGuide/refs/heads/main/contig_stats3.py
```



```
## Boas Pucker ##
## puckер@uni-bonn.de ##

## based on script contig_stats.py Pucker et al., 2016. doi:10.1371/journal.pone.0164321

import re, sys, os
from operator import itemgetter

# --- end of imports ---

__version__ = "v3.1"

__citation__ = "de Oliveira, J. A. V. S.; Choudhary, N.; Meckoni, S. N.; Nowak, M. S.; Hagedorn, M.; Pucker, B. (2025). Cookbook for Plant Genome Sequences. doi: 10.20944/preprints202508.1176.v2"
```

```
__usage__ = """
    Calculation of assembly statistics (""" + __version__ + """):
    python3 contig_stats3.py
    --in <INPUT_FASTA_FILENAME>

    optional:
    --min <MIN_CONTIG_LENGTH> [1000]
    --out <FULL_PATH_TO_OUTPUT_DIRECTORY>
    --exp <EXPRESSION_FILE(normalized)>

    bug reports and feature requests: puckер@uni-bonn.de
    Please cite: \n""" + __citation__ + """
"""


```

Dataset A

```
assembly name: DatasetA.genome.fasta_trimmed

number of contigs:      22
average contig length: 5510209.590909091
minimal contig length: 105320
maximal contig length: 16222222

total number of bases: 121224611
total number of bases without Ns:      121224611
GC content:            0.36077104013144656

N25:      15538466
N50:      14706422
N75:      11234264
N90:      9390472

E90N25: False
E90N50: False
E90N75: False
E90N90: False
```

Dataset B

```
assembly name: DatasetB.genome.fasta_trimmed

number of contigs:      13
average contig length: 10369230.23076923
minimal contig length: 262562
maximal contig length: 32605930

total number of bases: 134799993
total number of bases without Ns:      134799993
GC content:            0.3637223853565037

N25:      26122726
N50:      22402157
N75:      18291631
N90:      11381128

E90N25: False
E90N50: False
E90N75: False
E90N90: False
```

Dataset A

```
assembly name: DatasetA.genome.fasta_trimmed

number of contigs: 22
average contig length: 5510209.590909091
minimal contig length: 105320
maximal contig length: 16222222

total number of bases: 121224611
total number of bases without Ns: 121224611
GC content: 0.36077104013144656

N25: 15538466
N50: 14706422
N75: 11234264
N90: 9390472

E90N25: False
E90N50: False
E90N75: False
E90N90: False
```

Dataset B

```
assembly name: DatasetB.genome.fasta_trimmed

number of contigs: 13
average contig length: 10369230.23076923
minimal contig length: 262562
maximal contig length: 32605930

total number of bases: 134799993
total number of bases without Ns: 134799993
GC content: 0.3637223853565037

N25: 26122726
N50: 22402157
N75: 18291631
N90: 11381128

E90N25: False
E90N50: False
E90N75: False
E90N90: False
```

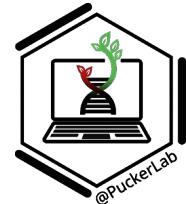
All next steps using **trimmed_genome.fasta** !!

<https://github.com/marbl/merqury>

<https://anaconda.org/bioconda/merqury>

- Evaluates genome assembly quality without a reference by comparing k-mers from high-accuracy reads to those in the assembly, estimating metrics like base-level accuracy (QV)

Phred Quality Score	Probability of incorrect base call	Base call accuracy
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1000	99.9%
40	1 in 10,000	99.99%
50	1 in 100,000	99.999%
60	1 in 1,000,000	99.9999%



Step 1: Calculate the right k size

```
bash /vol/data/tools/miniconda3/envs/merqury/bin/best_k.sh 134799993
```

genome: 134799993

18.4853

Step 2: Count k-mers from raw reads using meryl

```
meryl k=18 count /vol/data/julie/deNBI/ERR4760286_pass.fastq \  
output DatasetB.meryl >> /vol/data/julie/deNBI/merqury/DatasetB/meryl.log 2>&1 &
```

Step 3: Run Merqury to assess assembly quality

```
merqury.sh DatasetB.meryl /vol/data/julie/deNBI/DatasetB.genome.fasta output_DatasetB >>  
/vol/data/julie/deNBI/merqury/DatasetB/merqury.log 2>&1 &
```

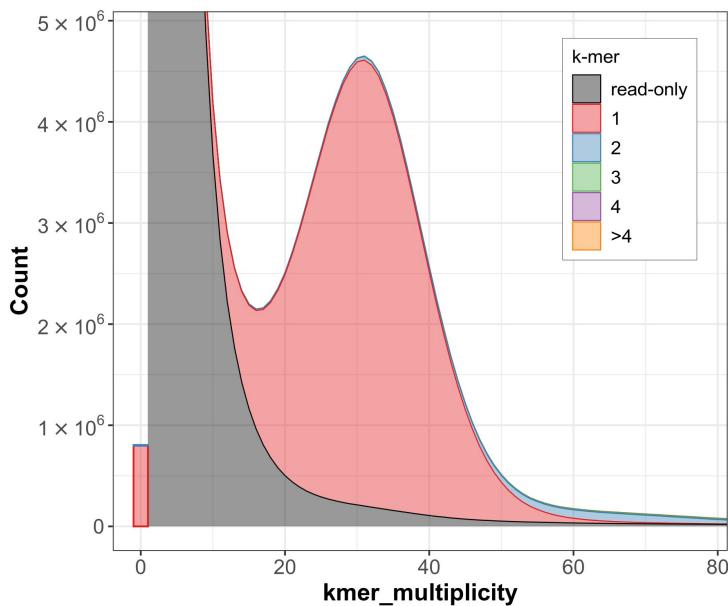
Assembly of interest: **DatasetB.genome**

Total (present) k-mers uniquely found only in the assembly: **134799772**

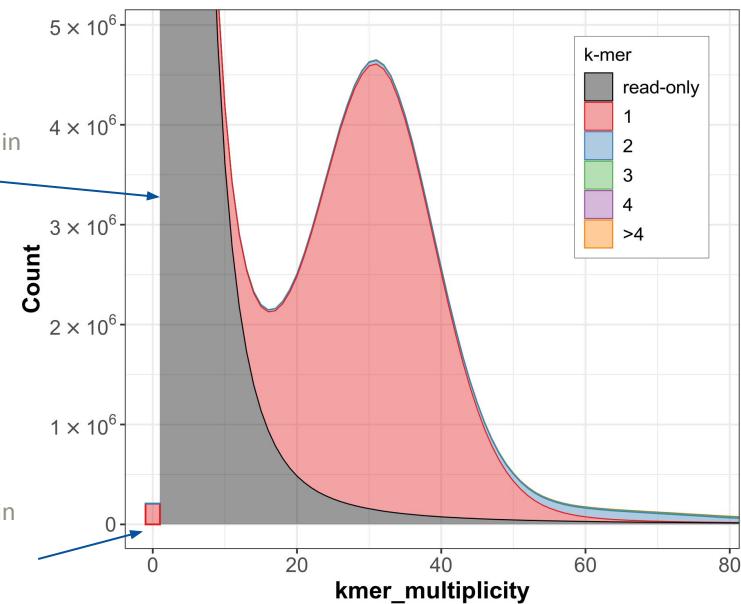
QV: **40.6544**

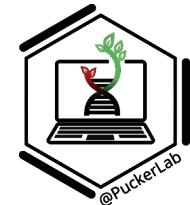
Error rate: **8.6013e-05**

Dataset A QV: 34.3181



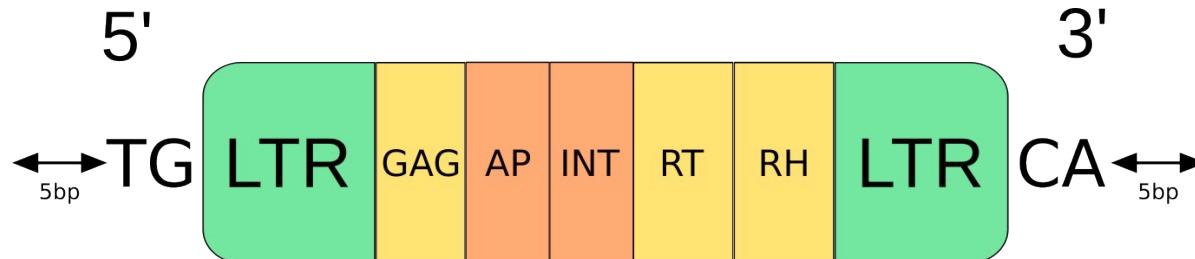
Dataset B QV: 40.6544





https://github.com/oushujun/LTR_retriever

- requires GenomeTools (<https://github.com/genometools/genometools>) for Step 1 and 2
- LTR_retriever can be used with conda
- The LTR Assembly Index (LAI) assesses assembly continuity by measuring intact LTR (Long-Term repeats) retrotransposons relative to total LTRs. Higher LAI (>10-15) indicates better assembly of repetitive regions

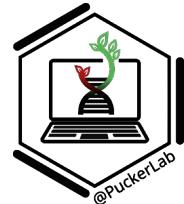


Step 1: Index the genome using GenomeTools (gt)

```
/vol/data/tools/genometools/bin/gt suffixerator \  
-db /vol/data/julie/deNBI/DatasetA.genome.fasta \  
-indexname DatasetA.genome.fa \  
-dna -tis -suf -lcp -des -ssp -sds &
```

Step 2: Identify candidate LTR elements using similarity searches

```
/vol/data/tools/genometools/bin/gt ltrharvest \  
-index DatasetA.genome.fa \  
-minlenltr 100 -maxlenltr 7000 \  
-mintsd 4 -maxtsd 6 \  
-motif TGCA -motifmis 1 \  
-similar 85 -vic 10 -seed 20 \  
-seqids yes > DatasetA.genome.fa.harvest.scn &
```



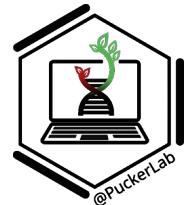
Step 3: Detects additional LTR candidates using structural features

```
conda activate ltr_env
```

```
LTR_FINDER_parallel \  
-seq /vol/data/julie/deNBI/DatasetA.genome.fasta \  
-threads 10 -harvest_out -size 1000000 -time 300 2>&1 &
```

Step 4: Combine Results

```
cat DatasetA.genome.fa.harvest.scn DatasetA.genome.fasta.finder.combine.scn \  
> DatasetA.genome.fa.rawLTR.scn
```



Step 5: LTR_retriever calculates LAI and filters high confidence LTR-RTs

```
LTR_retriever \
-genome DatasetA.genome.fasta \
-inharvest /vol/data/julie/deNBI/LAI/DatasetA/DatasetA.genome.fa.rawLTR.scn \
-threads 10 &
```

To see LAI results

```
nano DatasetA.genome.fasta.out.LAI
```

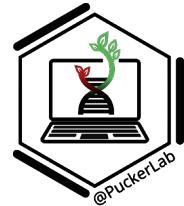
Chr	From	To	Intact	Total	raw_LAI	LAI
whole_genome	1	121224611	0.0089	0.0981	9.08	14.01

0 to 10	draft quality
10-20	reference quality
> 20	gold quality

- BUSCO = Benchmarking Universal Single-Copy Orthologs
- Universal genes -> expected to be in the genome
- Single-Copy -> expected to be present only once per haplophase
- if all BUSCO genes are present in the assembly => good assembly
- some might be natively missing due to gene loss in evolution

https://busco.ezlab.org/busco_userguide.html

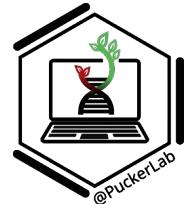
- Installation with docker 
- Installation with conda
- Manual installation - not recommended



https://busco.ezlab.org/busco_userguide.html

- Installation with docker 
- Installation with conda
- Manual installation - not recommended

- Tasks:
 1. Install Docker (<https://docs.docker.com/engine/install/ubuntu/>)
 2. install and run BUSCO on your genome assembly
 - a. choose a lineage dataset
 - b. choose the correct mode



- exemplary commands:

list datasets:

```
sudo docker run --rm -v /vol/data:/vol/data ezlabgva/busco:v6.0.0_cv1 busco --list-datasets
```

docker container name (will be automatically downloaded)

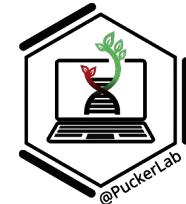
command that will be run in the
docker container starts here

start the docker container
input file
mode and lineage-dataset
output file

important for the docker container to access files

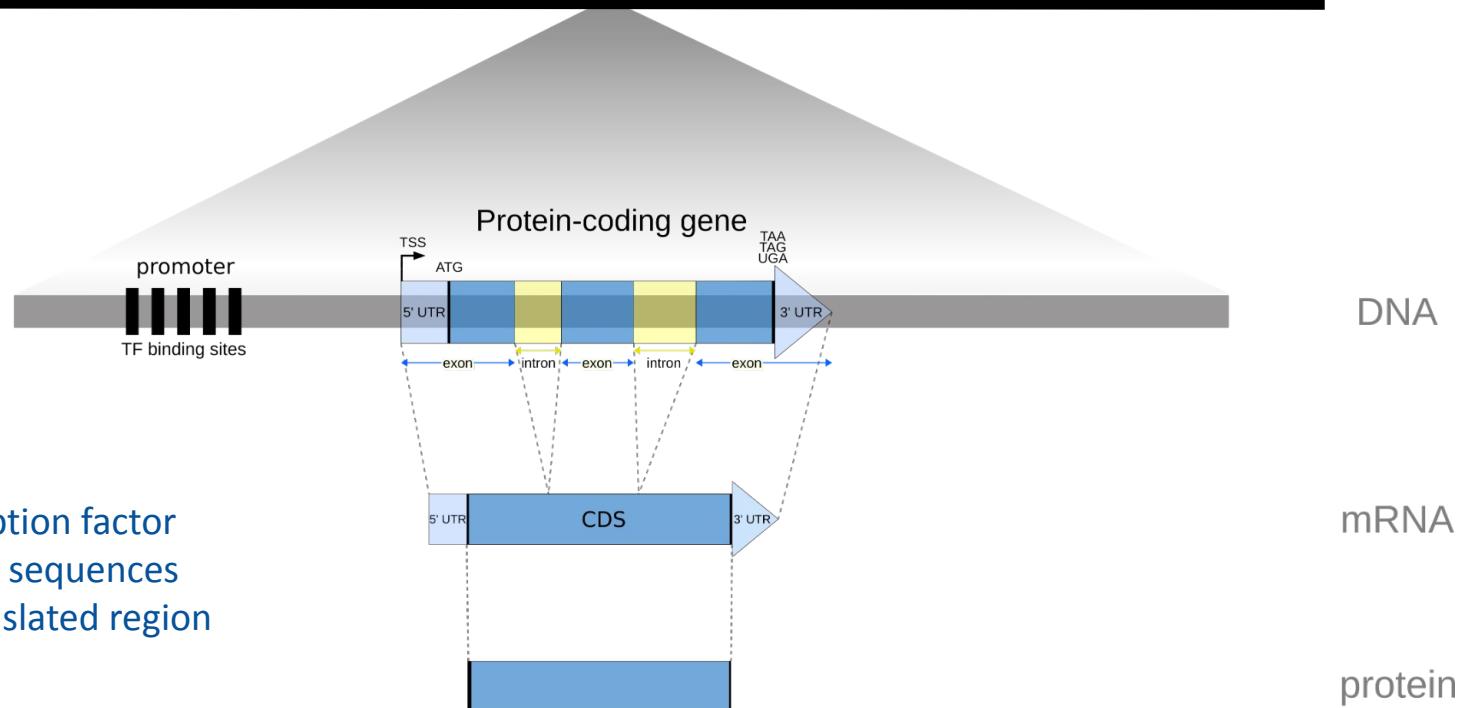
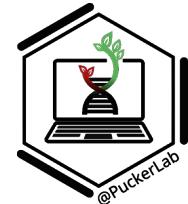
```
sudo docker run --rm -v /vol/data:/vol/data ezlabgva/busco:v6.0.0_cv1 \  
busco -i /vol/data/user/assembly.fasta \  
-m genome --cpu 13 -l brassicales_odb12 \  
--out_path /vol/data/user/BUSCO/ -o busco_run_id --opt-out-run-stats
```

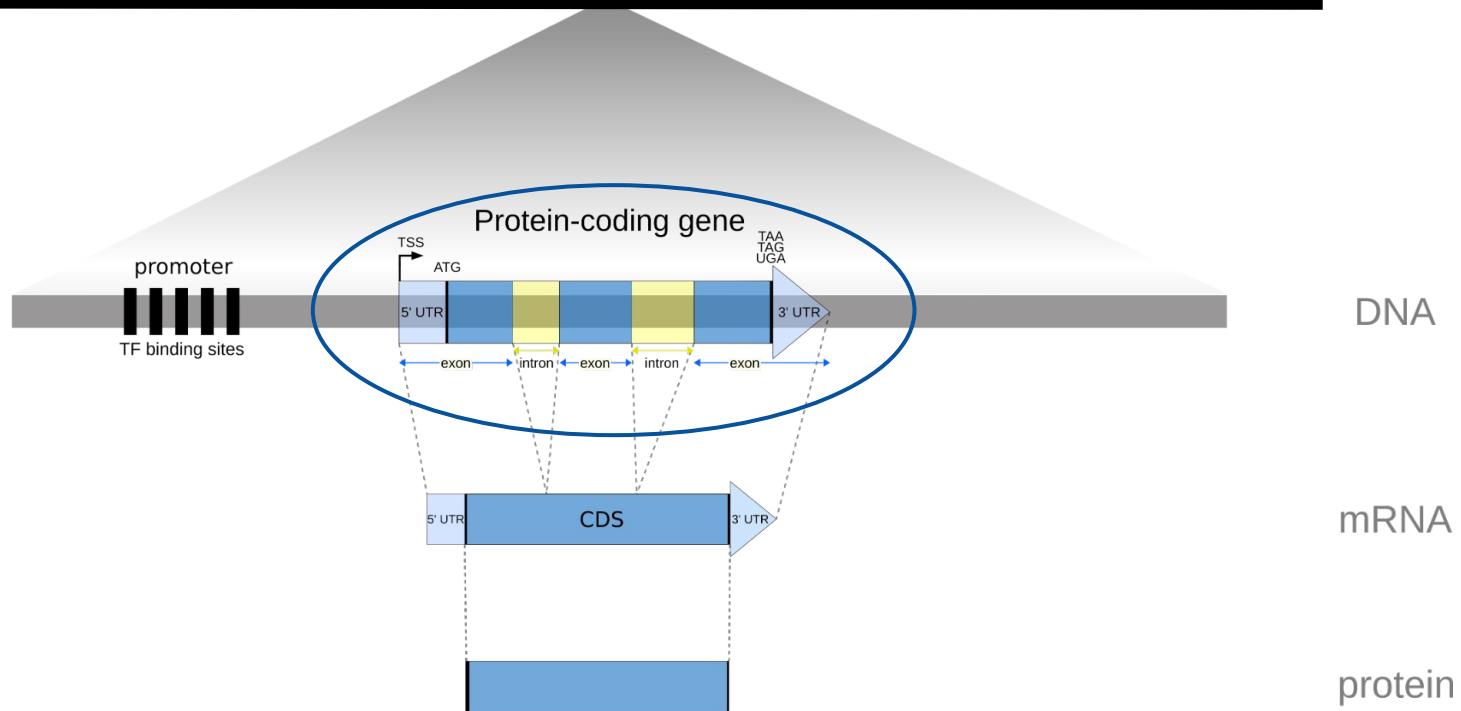
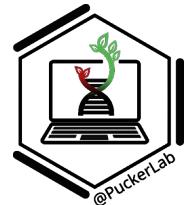
Parameters	DatasetA (Shasta)	DatasetB (NextDenovo)
No. of contigs		
Genome assembly size		
Minimum contig size		
Maximum contig size		
Contiguity (Contig N50)		
Contiguity (LAI)		
Completeness (BUSCO)		
Correctness (QV - Merqury)		

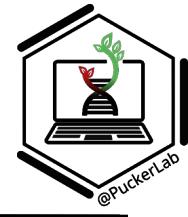


- Process of finding genes in a genome sequence

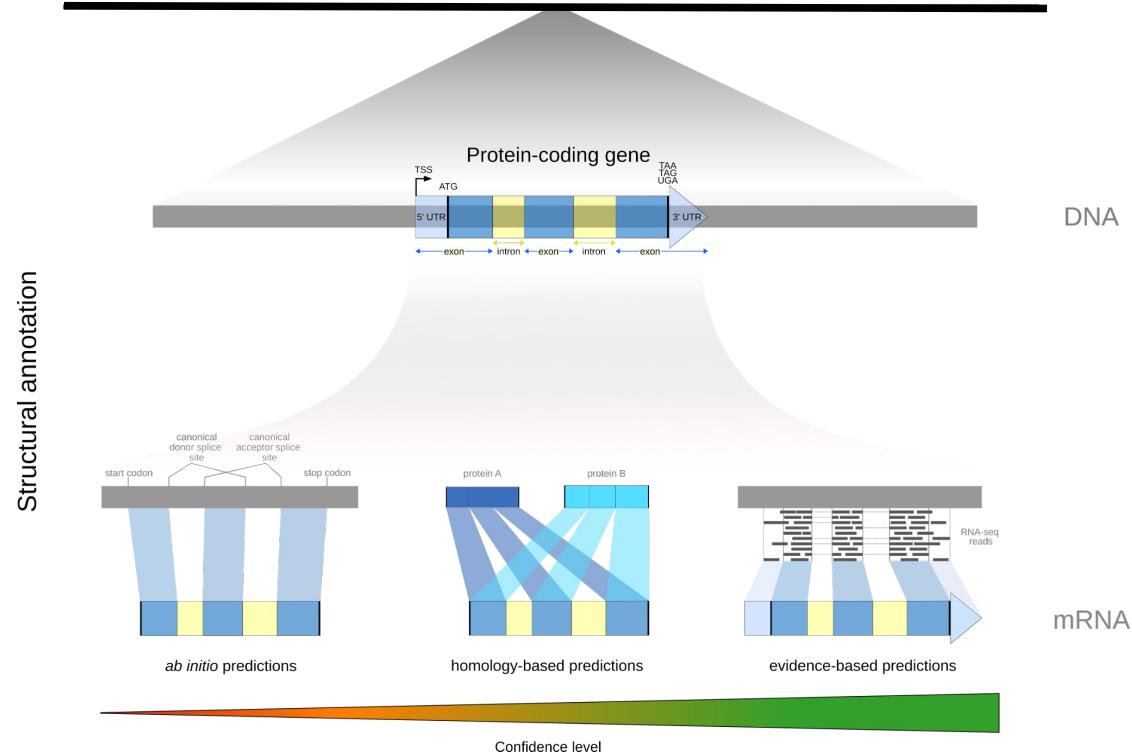
```
TTTACAGTTCTTGATGTATTGGAAACCGAAAAGTTGGATATTAAACGTTCAATATAAAATAGTTGGAGTTTGTGGAAATTTCACTAGT  
GTTACAAAATGTGATTAAGATCGTTTACTCGTTATTTAGGTTAGTCTTGGTATAAGTATGTATTTAAAAAACGTAACGAGGTACATTTAAGTGT  
GTTATTTACACTGAAAACAAGCTAATTAGAGGTGGTCGAGAGCAGATATCATGATTTTTTAATGTATGTAATCCGTTCATTGTGTATCATTATTAT  
TATGATTTCAGAAATGTATGTTGTATCTTCAATTACCAAAATTCAATGTCCTACGCATATGATGGAATTAAACCAAATACAAGTAGCTAGTTAGTGA  
CTAAACTATCAAACATATTAAAAACAATTATGAAGACTATTGACAACTACGCTACTATAAAATTAAATCAGAAGAGAAAAATGTAGATCTAAACTATCAAACAT  
AAGTCCTAAAATATGTTAAAAGGCAAGGAGGGCCAACCATTCCATCTCGCCCACTTTGTCTAATATGTCACTCTCTCATATCCTAACCTCCAATAACATT  
CAAATCATTATTATATCATTATTCTGAATATACCATTTCATAACACAGAAATTAAACCTAAACGAAATCGAAACGAATCAGAACCGAAATCCG  
TTGTGAGTCACAGTTGATTGGTTCAACTGGATTGGATAATATTCAAGTGGAAAGAAATAAATAAAAGTTGGTTGGAAAATTATTCCAATAAATCTCGTGTG  
TTAATAAGAGATAAATATTTCAAACTGTAACCTAACAAATAAATAAAAGAGAAATTAAATGCAAATATTGCTATATTGGTTATTACTTAGAGAC  
AGATAAGTGACCAAAGGGTTCGATTCTGAAATAGTTGAATAAAAAGTAATTATTGTAATTGGCAAAGAAAAGTTTAGTTTTAGTTGTGAG  
TCTAGGTGGTGGCTCAAGAACAGTTGGCTCAGCTGATTCTCTTCTAGAAGTCTTGTACTTGTCTTACCAAGAGATCTACTGATAGTGTGTTCTCTTC  
ATAAGATGATCTGAAATGATAAAGTTGATCTTTCTGAACTGATCCATTGTCTTGGTGTGATCACCAGAGCTAAAGATGTTGACGGCGCAAGGCATAGCGGTT  
CTTGTGGGTTTCGTTTCAAATCTGTATCATTCAATAATGTTCTTTATGTTGCAAGGGGATGTGTCTAAATGTGTTGGGATCCATTGGTGTGAGGATG  
GTGAGTGTGAGAAAAGTTTCATCTGTTGGATCAAACAGGTGCGTGGCAACGAAGGAAACGAGGCCGGAAAGAAAGCGTGAATCTTCTGAAATCTAT  
CTACTGGTTGGTTGATGATCGCATAAGAGTAGTTGTTCAAGCTTCAAGCTTAAAGTGCTTCATTACCATGGAGATGGGGATGAAGTATTCTGGGACACACT  
GGATACATCATCGTGGAACGCACCTGAGTCGTTCTCAAGCTGCTGGTATCGAGCTGGATCAAACCTACCAATTGTAGGATTAGACGAAGCAAAGC  
GGAGATATGAGGATCAGATGGTCCAAGCATTACTCTTCTTGATCAGACCTGAAGAAGACGAAGAGTCGTCCTTAATTGAGAAATTCACTAGTAGGAGATAGCAG  
GCATAGTCCCCGAGTGGTCTCCAGGGTACTCAGAGCAGTCCCTAGTGAAGAAAATGGATTGGAGGGAGAAGTACAACGTCAGCTACTTGCAAGTACGACCATGA  
GCACACTGTCTAGCCAAGTAGTATCAGTTAAATTGAATTAGTGAAGAAGTTGAACTGTTCAAATCCAAGTGTTGCAATTACAGTAGACGACACAGTAA  
TGTAAACCGAGTACATTGGTACGGTTAATGTAATTCCGGTATGGGCTGGAGAGAAACTATGTAGGAGTTGTCTGATGTACATTCTTATCTCTGGTT
```

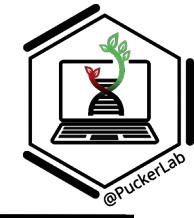






- (a) ***ab initio* prediction:**
uses intrinsic sequence features
- (b) **homology-based:**
using gene sequences from closely related species
- (c) **evidence-based:**
using transcriptomics and/or proteomic hints from the species itself



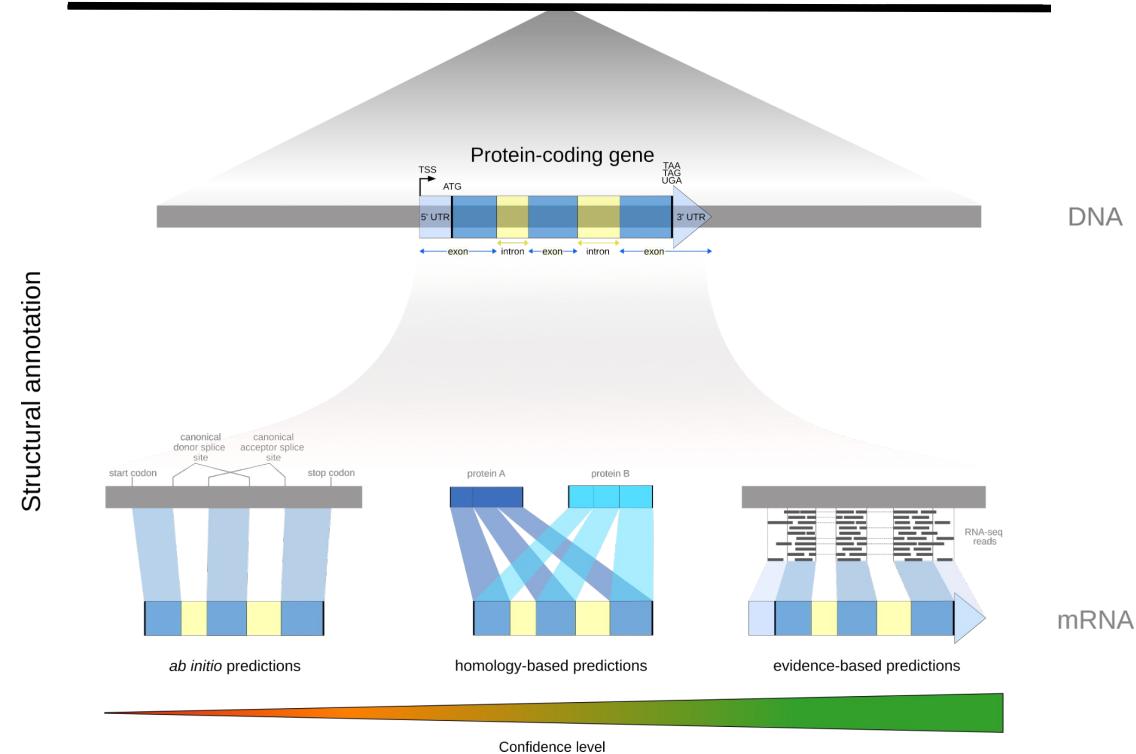


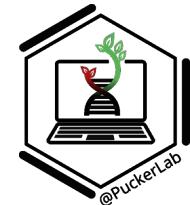
(a) ***ab initio*** prediction:
Augustus, Helixer, Tiberius

(b) **homology-based:**
GeMoMa

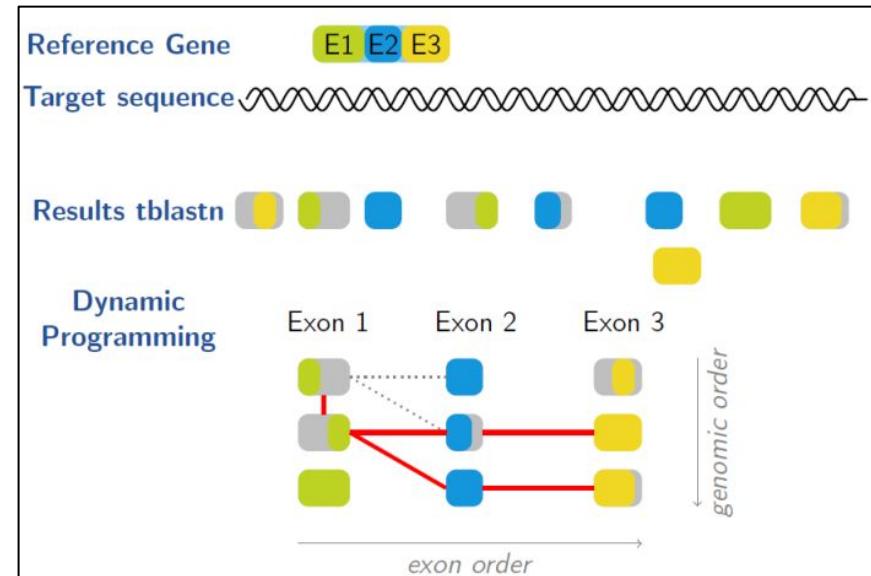
(c) **evidence-based:**
BRAKER3

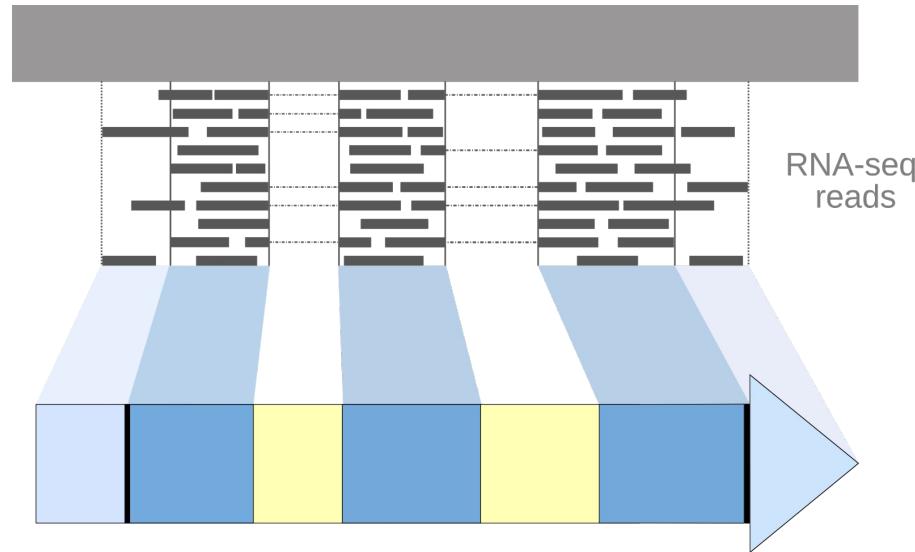
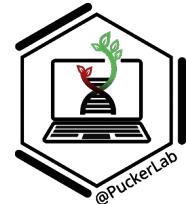
(d) **Integrated pipelines:**
Helixer + GeMoMa +
BRAKER3





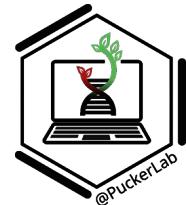
- GeMoMa → Gene Model Mapper
- Homology-based gene prediction
- Requirements for GeMoMa:
 - (a) Protein-coding genes from closely-related reference genomes
 - (b) RNA-seq evidence to refine borders of exons (optional)





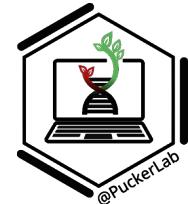
1. RNA-seq reads
2. read mapping of RNA-seq reads to the genome sequence assembly

1. Downloading RNA-seq reads



- SRA = Sequence Read Archive
- downloading tools:
 1. prefetch (compressed file)
 2. fasterq-dump (un- compresses the files)
- December 2, 2025: **SRA Toolkit** Release 3.3.0 -> **GitHub** repository (apt version is outdated)
- Tasks:
 1. search for suitable two RNA-seq read files
 - a. paired-end Illumina
 - b. max. 6 Gb total size
 2. download them

1. Downloading RNA-seq reads



- SRA = Sequence Read Archive
- downloading tools:
 1. prefetch (compressed file)
 2. fasterq-dump (un-compresses the files)
- December 2, 2025: **SRA Toolkit** Release 3.3.0 -> **GitHub** repository (apt version is outdated)

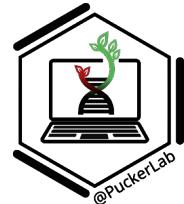
```
/vol/data/tools/prefetch -p --max-size 250g --output-directory /vol/data/DNA_reads XXX
```

no. of threads (CPU)

SRA/ENA read ID

```
/vol/data/tools/fasterq-dump -e 15 /vol/data/DNA_reads/XXX --outdir /vol/data/DNA_reads/raw_reads
```

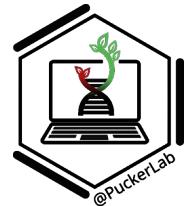
2. Mapping RNA-seq reads



- RNA-seq reads alignment programs: HISAT2, STAR
- HISAT2 easily installable via apt
- samtools useful tool for SAM/BAM files (mapping files)

- Tasks:
 1. install HISAT2 (apt package manager recommended)
 2. map the RNA-seq reads to your genome assembly
 3. sort the mapping file

2. Mapping RNA-seq reads



1. Installation: `sudo apt install hisat2 samtools`

2. Indexing the genome assembly:

```
hisat2-build -p 15 /vol/data/genome/assembly.fasta /vol/data/genome/assembly.fasta.hisat2.index
```

output index name (will create multiple files!)

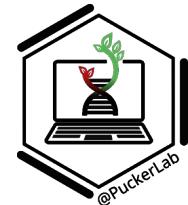
3. Starting the mapping:

```
hisat2 -p 15 -x \
```

input files `/vol/data/genome/assembly.fasta.hisat2.index -1 /vol/data/read_1.fq.gz -2 /vol/data/read_2.fq.gz \`

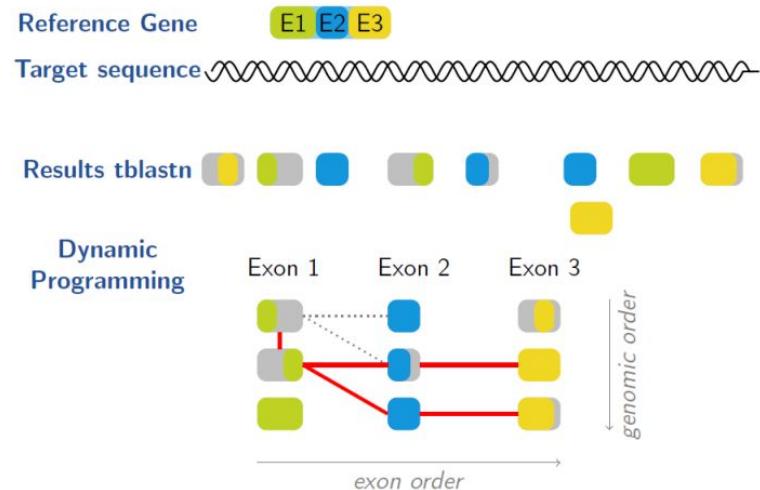
output file `| samtools sort --O BAM -o /vol/data/assembly.fasta.RNA_reads.sorted.bam`

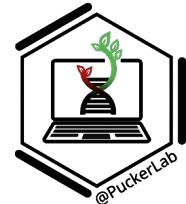
very important



- Requirements for GeMoMa:

- (a) Protein-coding genes from closely-related reference genomes
- (b) RNA-seq evidence to refine borders of exons (optional) 





- Requirements for GeMoMa:

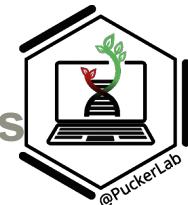
- (a) Protein-coding genes from closely-related reference genomes

- Genome sequence (fasta format)

- Gene annotation (gff3 format)

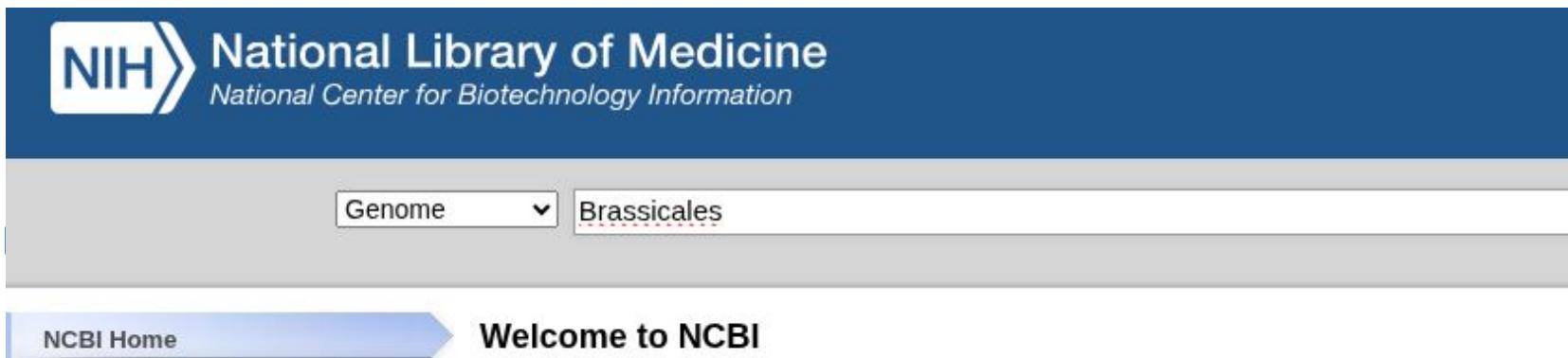
GFF3 format

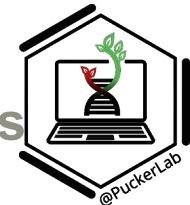
seqid	source	type	start	end	score	strand	phase	attributes
ctg1	GAF	gene	7031	7666	.	+	.	ID=gene001
ctg1	GeMoMa	mRNA	7031	7666	.	+	.	ID=gene001.1;Parent=gene001
ctg1	GeMoMa	CDS	7031	7097	.	+	0	Parent=gene001.1
ctg2	GeMoMa	CDS	7401	7666	.	+	2	Parent=gene001.1



Tasks:

- A. Search NCBI for nearly-complete annotated Brassicales genome sequences
- NCBI Genome (<https://www.ncbi.nlm.nih.gov/>)
 - Search for Brassicales





Tasks:

A. Search NCBI for nearly-complete annotated Brassicales genome sequences

- NCBI Genome
- Search for Brassicales
- Choose annotated from filters
- NCBI RefSeq assemblies recommended

Genome

Selected taxa
Brassicales  Enter one or more taxonomic names 

 Filters 

INCLUDE ONLY

- Reference genomes
- Annotated genomes
 - Annotated by NCBI RefSeq
 - Annotated by GenBank submitter
 - Metagenome-assembled genomes (MAGs)
 - From type material
 - From ICTV exemplar

SEARCH WITHIN RESULTS



ASSEMBLY LEVEL

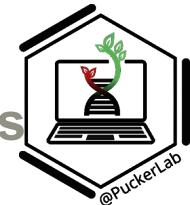
contig scaffold chromosome complete

YEAR RELEASED

1980 2025

B. Download any 4 using wget command (using FTP - file transfer protocol)

- Genome file (format ?)
- Gene annotation (format ?)



Tasks:

B. Download any 4 using wget command (using FTP - file transfer protocol)

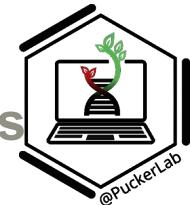
- Genome file (format ?)
- Gene annotation (format ?)

Genome assembly TAIR10.1

[reference](#)

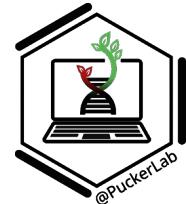
NCBI RefSeq assembly GCF_000001735.4

Submitted GenBank assembly GCA_000001735.2



Tasks:

- A. Download any 4 using wget command (using FTP - file transfer protocol)
 - Genome file (format ?) (_genomic.fna.gz)
 - Gene annotation (format ?) (_genomic.gff.gz)
 - right-click on the link → copy link → In the VM,
wget (paste copied link)



1. Installing GeMoMa: <https://www.jstacs.de/index.php/GeMoMa>

```
wget http://www.jstacs.de/download.php?which=GeMoMa
```

```
mv download.php\?which\=GeMoMa GeMoMa
```

```
unzip GeMoMa
```

2. Installing requirements

- mmseqs2: <https://github.com/soedinglab/MMseqs2>

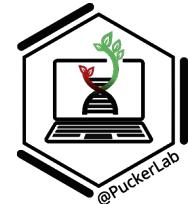
```
wget https://mmseqs.com/latest/mmseqs-linux-gpu.tar.gz; tar xvfz mmseqs-linux-gpu.tar.gz; export  
PATH=$(pwd)/mmseqs/bin/:$PATH
```

- v15< java >v1.8: To check if available, java -version

If java not found,

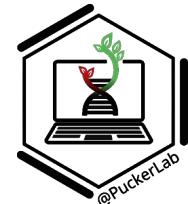
```
sudo apt install openjdk-11-jdk or sudo update-alternatives --config java
```

3. Running GeMoMa pipeline

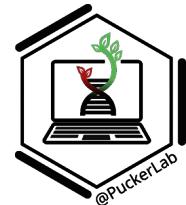


Command:

```
java -jar /path/to/GeMoMa-1.9.jar CLI GeMoMaPipeline \
t=/path/to/genome.fasta \
i=SG1 a=/path/to/SG1.gff3.gz g=/path/to/SG1.genome.fasta.gz \
i=SG2 a=/path/to/SG2.gff3.gz g=/path/to/SG2.genome.fasta.gz \
i=SG3 a=/path/to/SG3.gff3.gz g=/path/to/SG3.genome.fasta.gz \
i=SG4 a=/path/to/SG4.gff3.gz g=/path/to/SG4.genome.fasta.gz \
r=MAPPED ERE.m=/path/to/assembly.fasta.RNA_reads.sorted.bam \
outdir=/path/to/Gemoma_output/ \
pc=true pgr=true p=true o=true \
GeMoMa.c=0.4 GeMoMa.Score=ReAlign Extractor.r=true \
GAF.f="start=='M' and stop=='*' and (isNaN(score) or score/aa>='0.75')" \
AnnotationFinalizer.r=NO threads=13 >> gemoma.log 2>&1 &
```



```
java -jar /vol/data/denbi2025/tools/GeM  
oMa.1/GeMoMa-1.9.jar CLI GeMoMaPipeline t=/vol/data/denbi2025/DatasetB.genome.fasta s=own i=AT a=/vol/  
da ta/denbi2025/references/GCF_000001735.4_TAIR10.1_genomic.gff.gz g=/vol/data/denbi2025/references/GCF_00  
01735.4_TAIR10.1_genomic.fna.gz s=own i=CS a=/vol/data/denbi2025/references/GCF_000633955.1_Cs_genomic.  
gff.gz g=/vol/data/denbi2025/references/GCF_000633955.1_Cs_genomic.fna.gz s=own i=BN a=/vol/data/denbi20  
25/references/GCF_020379485.1_Da-Ae_genomic.gff.gz g=/vol/data/denbi2025/references/GCF_020379485.1_Da-A  
e_genomic.fna.gz r=MAPPED ERE.m=/vol/data/denbi2025/hisat_DatasetB_sorted.bam outdir=/vol/data/denbi2025  
/GeMoMa/ pc=true pgr=true p=true o=true GeMoMa.c=0.4 GeMoMa.Score=ReAlign Extractor.r=true GAF.f="start=  
='M' and stop=='*' and (isNaN(score) or score/aa>='0.75')" AnnotationFinalizer.r=NO threads=25 >> gemoma  
.log 2>&1 &
```

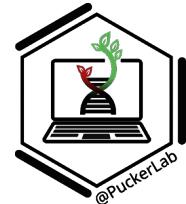


Today you learned:

- Evaluating a genome assembly sequence
- Structural annotation of a genome assembly sequence

Tomorrow you will learn:

- Evaluating your structural annotation
- Finalizing your structural annotation (Filtering)

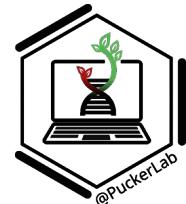


Yesterday you learned:

- Evaluating a genome assembly sequence
- Structural annotation of a genome assembly sequence

Today you will learn:

- Evaluating your structural annotation
- Finalizing your structural annotation (Filtering)



<https://github.com/bpucker/teaching/tree/master/deNBI2025>

Background information:

<https://www.preprints.org/manuscript/202508.1176>



 Preprints.org

Cookbook for Plant Genome Sequences

Julie Anne Vieira Salgado de Oliveira , Nancy Choudhary , Samuel Nestor Meckoni , Melina Sophie Nowak , Marie Hagedorn , Boas Pucker * 

Abstract

Access to genome sequences is crucial to investigate and engineer traits in plants, explore biodiversity, and support phylogenetic studies. During the last decade, affordable sequencing devices have substantially increased the size of the genomics community by enabling individual research groups to sequence the genomes of their favourite plants. There has also been a quick development of novel tools for sequencing data analysis. Here, we share experiences with plant long-read genomics methods for Oxford Nanopore Technologies (ONT) and provide hands-on guidelines to support newcomers' dive into plant genomics. The full workflow from planning a plant genome sequencing project to public release of associated data sets is covered.

- exemplary commands:

list datasets:

```
sudo docker run --rm -v /vol/data:/vol/data ezlabgva/busco:v6.0.0_cv1 busco --list-datasets
```

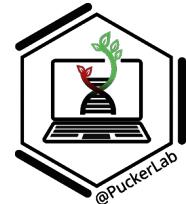
docker container name (will be automatically downloaded)

command that will be run in the
docker container starts here

important for the docker container to access files

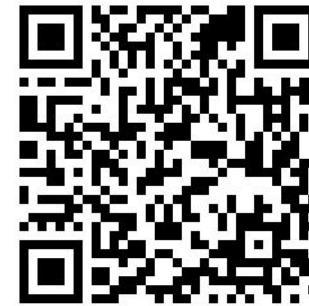
start the docker container
input file
mode and lineage-dataset
output file

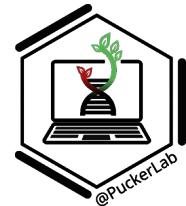
```
sudo docker run --rm -v /vol/data:/vol/data ezlabgva/busco:v6.0.0_cv1 \  
busco -i /vol/data/user/protein.fasta \  
-m proteins --cpu 13 -l brassicales_odb12 \  
--out_path /vol/data/user/BUSCO_pep/ -o busco_run_id --opt-out-run-stats
```



https://busco.ezlab.org/busco_userguide.html

- Tasks:
 1. run BUSCO on the results of the structural annotation
 - a. choose a lineage dataset
 - b. choose the correct mode
 2. Compare the results to the one from the genome assembly





1. BUSCO >95%

2. Number of predicted genes and proteins : 21,000> genes <40,000
Average number of genes in plants : 27,200 genes

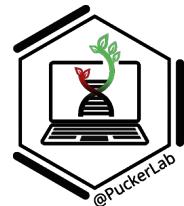
Calculate number of genes:

```
grep -c $'\tgene\t' final_annotation.gff3
```

Calculate number of protein sequences:

```
grep -c $'\tmRNA\t' final_annotation.gff3
```

Sample gff3 file

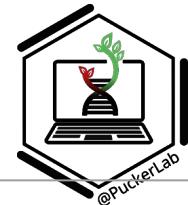


ctg000000	AUGUSTUS	gene	210589	210771	.	+	.	ID=g55;
ctg000000	AUGUSTUS	mRNA	210589	210771	0.72	+	.	ID=g55.t1;Parent=g55;
ctg000000	AUGUSTUS	start_codon	210589	210591	.	+	0	ID=g55.t1.start1;Parent=g55.t1;
ctg000000	AUGUSTUS	CDS	210589	210771	0.72	+	0	ID=g55.t1.CDS1;Parent=g55.t1;
ctg000000	AUGUSTUS	exon	210589	210771	.	+	.	ID=g55.t1.exon1;Parent=g55.t1;
ctg000000	AUGUSTUS	stop_codon	210769	210771	.	+	0	ID=g55.t1.stop1;Parent=g55.t1;
ctg000010	AUGUSTUS	gene	499686	506363	.	+	.	ID=g215;
ctg000010	AUGUSTUS	mRNA	499686	506363	0.32	+	.	ID=g215.t1;Parent=g215;
ctg000010	AUGUSTUS	start_codon	499686	499688	.	+	0	ID=g215.t1.start1;Parent=g215.t1;
ctg000010	AUGUSTUS	CDS	499686	499741	1	+	0	ID=g215.t1.CDS1;Parent=g215.t1;

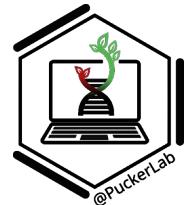


```
ctg000000      GAF      gene     7031     7132     .         +         .         ID=gene_0;transcrip
ts=1;complete=1;maxTie=NA;maxEvidence=1;combinedEvidence=1
ctg000000      GeMoMa   mRNA     7031     7132     .         +         .         ID=AT_rna-NM_102213
.1_R1;ref-gene=AT_gene-AT1G23650;aa=34;raa=29;score=100;prediction=1;bestScore=150;ce=1;rce
=1;tae=NA;tde=NA;tie=NA;minSplitReads=NA;tpc=0.049;minCov=0;avgCov=0.049;pAA=0.7647;iAA=0.6
765;lpm=12;maxScore=150;maxGap=5;nps=0;start=M;stop=*&;evidence=1;Parent=gene_0;sumWeight=1.
0;
ctg000000      GeMoMa   CDS      7031     7132     .         +         0         Parent=AT_rna-NM_10
2213.1_R1;pc=0.049;minCov=0
ctg000000      GAF      gene     25621    25740    .         +         .         ID=gene_1;transcrip
ts=1;complete=1;maxTie=NA;maxEvidence=1;combinedEvidence=1
ctg000000      GeMoMa   mRNA     25621    25740    .         +         .         ID=AT_rna-NM_001124
772.1_R0;ref-gene=AT_gene-AT2G01023;aa=40;raa=40;score=205;prediction=0;bestScore=205;ce=1;
rce=1;tae=NA;tde=NA;tie=NA;minSplitReads=NA;tpc=0;minCov=0;avgCov=0;pAA=1;iAA=1;lpm=40;maxS
core=205;maxGap=0;nps=0;start=M;stop=*&;evidence=1;Parent=gene_1;sumWeight=1.0;
ctg000000      GeMoMa   CDS      25621    25740    .         +         0         Parent=AT_rna-NM_00
1124772.1_R0;pc=0;minCov=0
ctg000000      GAF      gene     32910    34053    .         -         .         ID=gene_4160;transc
ripts=1;complete=1;maxTie=0.0;maxEvidence=1;combinedEvidence=1
```

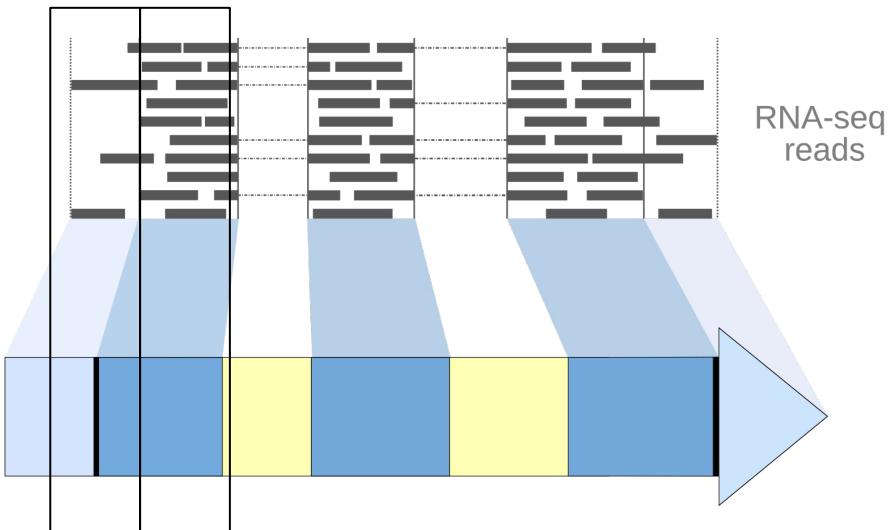
Understanding GeMoMa's GFF attributes



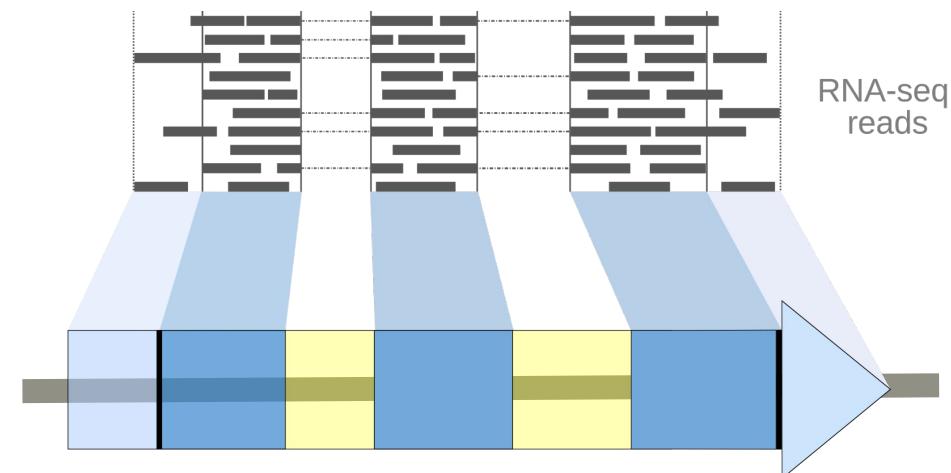
Attribute	Long Name	Feature	Description
aa	amino acids	mRNA	number of amino acids in predicted protein
raa	reference amino acids	mRNA	number of amino acids in reference protein
score	GeMoMa score	mRNA	score computed by GeMoMa using substitution matrix, gap costs and additional penalties
avgCov	average coverage	mRNA	average coverage of all bases of the prediction given RNA-seq evidence
tpc	transcript percentage coverage	mRNA	percentage of covered bases per predicted transcript given RNA-seq evidence
tie	transcript intron evidence	mRNA	percentage of predicted introns per predicted transcript with RNA-seq evidence
evidence		mRNA	number of reference organisms that have a transcript yielding this prediction

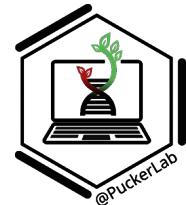


average coverage = sum of coverage per base/ total bases



tie (transcript intron evidence)





```
cd /vol/data/GeMoMa/
```

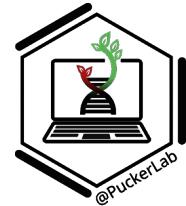
First filter

```
java -jar /vol/data/tools/GeMoMa/GeMoMa-1.9.jar CLI GAF
g=unfiltered_predictions_from_species_0.gff g=unfiltered_predictions_from_species_1.gff
g=unfiltered_predictions_from_species_2.gff f="start=='M' and stop=='*' and aa>20 and "
atf="tie==1 and evidence>1" outdir=filter1/
```

```
cd filter1/
```

Evaluation

```
java -jar /vol/data/tools/GeMoMa/GeMoMa-1.9.jar CLI Extractor a=filtered_predictions.gff
g=DatasetB.genome.fasta p=true
```



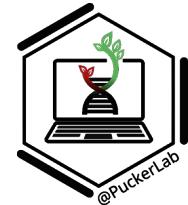
Run protein busco on filtered protein file

```
sudo docker run --rm -v /vol/data:/vol/data ezlabgva/busco:v6.0.0_cv1 busco -i proteins.fasta -m proteins --cpu 13 -l brassicales_odb12 -o busco_pep
```

Comparison (Better or worse?)

	Unfiltered	Filtered
BUSCO		
Number of genes		
Number of proteins		

Finalizing annotation



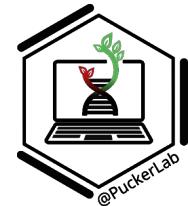
Once you are happy with annotation, rename headers using this command

```
java -jar /vol/data/tools/GeMoMa/GeMoMa-1.9.jar CLI AnnotationFinalizer g=/vol/data/DatasetB.genome.fasta  
a=filtered_predictions.gff tf=true rename=SIMPLE p=Athal n=false outdir=final/
```

Extract protein and cds sequences

```
java -jar /vol/data/tools/GeMoMa/GeMoMa-1.9.jar CLI Extractor a=final_annotation.gff  
g=/vol/data/DatasetB.genome.fasta p=true c=true
```

Availability of materials

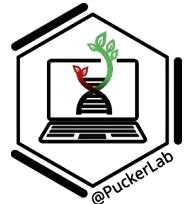
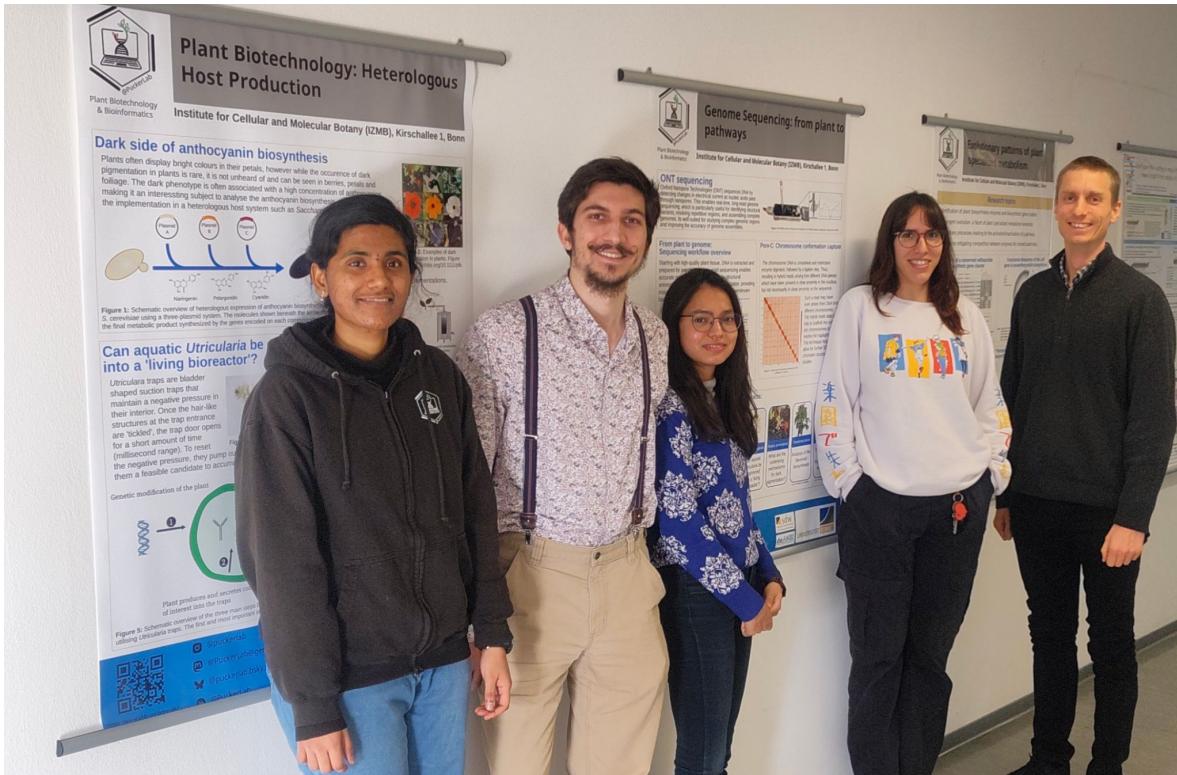


Please find all course materials here:

<https://github.com/bpucker/teaching/tree/master/deNBI2025>



Acknowledgements



Email: pucker@uni-bonn.de
 Web: pbb.uni-bonn.de

