Genomic features of *Agrobacterium divergnes* Azo12 and wheat responses to salinity stress

Agnieszka Kalwasińska[1,2*], Sweta Binod Kumar[1], Marcin Gołębiewski[2,3,4], Marcin Sikora[2,3,4], Edyta Deja-Sikora[2,5], Eman Abdalhady[6]

[1]Department of Environmental Microbiology and Biotechnology, Faculty of Biological and Veterinary Sciences, Nicolaus Copernicus University in Toruń, Lwowska 1, 87-100 Toruń

[2]Institute of Advanced Studies, Nicolaus Copernicus University in Toruń, Gagarina 11, 87-100 Toruń

[3]Department of Systems Biology, Faculty of Biological and Veterinary Sciences, Nicolaus Copernicus University in Toruń, Lwowska 1, 87-100 Toruń

[4]Centre for Modern Interdisciplinary Technologies, Nicolaus Copernicus University in Toruń, Wileńska 4, 87-100 Toruń

[5]Department of Microbiology, Nicolaus Copernicus University in Toruń, Lwowska 1, 87-100 Toruń

[6]Department of Botany and Microbiology, Faculty of Science, Gamaa St. Cairo University, 12613, Giza, Egypt

*Corresponding author: Agnieszka Kalwasinska kala@umk.pl

Table S1. Probe sequences used to deplete chloroplast, mitochondrial, and nuclear rRNA from total RNA samples

Probe name	**Sequence**	**Target rRNA**
mit1	GAGCCTCTTTTCTTTCTGCCTAGCTCCC	Mitochondrial 16S rRNA
mit2	ACGCACCCGTTCGCCACTTTGTTCTC	Mitochondrial 16S rRNA
mit3	GTCCTGTCATGATCGCGCACTCAACG	Mitochondrial 16S rRNA
mit4	CGACTTTCACTTTCAACCCGATTCACCG	Mitochondrial 16S rRNA
mit5	GATCCGTGTAGACCAAGGGCGAACAC	Mitochondrial 16S rRNA
mit6	GGCTCCTTGGCTCACTTCGGTTGC	Mitochondrial 16S rRNA
mit7	CCATTGTAGCACGTGTGTGGCCCAG	Mitochondrial 16S rRNA
chl1	GGATTCCTCCTTTTGCTTCTCAGCCT	Chloroplast 16S rRNA
chl2	GATAGTTTCCACCGCCTGTCCAGGG	Chloroplast 16S rRNA
chl3	ACAACACTGCACGGGTCGATACGCACAGC	Chloroplast 16S rRNA
chl4	CTGTTCAGGGTTCCAAACTCAACGT	Chloroplast 16S rRNA
chl5	GAACTGAGGACGGGTTTTTGGGGTTAG	Chloroplast 16S rRNA
chl6	GTCACTAGCCCTGCCTCCGGCATCC	Chloroplast 16S rRNA
nuc1	GAGCCATTCGCAGTTTTACAGTCTGA	Nuclear 18S rRNA
nuc2	CCTCCAATGGATCCTCGTTAAGGGAT	Nuclear 18S rRNA
nuc3	CCCATGCTAATGTATACAGAGCGTAGGC	Nuclear 18S rRNA
nuc4	AGTTTCAGCCTTGCGACCATACTCCC	Nuclear 18S rRNA
nuc5	CTAAGAACGGCCATGCACCACCACC	Nuclear 18S rRNA
nuc6	CCGTGGCCTAAAAGGCCATAGTCCCTC	Nuclear 18S rRNA
nuc7	TCACCGGACCATTCAATGGGTAGGAG	Nuclear 18S rRNA
nuc8	CCCTCGCGGTACTTGTTCGCTATCGG	Nuclear 25S rRNA
nuc9	GACTCGCACACATGTCAGACTCCTTG	Nuclear 25S rRNA
nuc10	CATCCCGCATCGCCAGTTCTGCTTAC	Nuclear 25S rRNA
nuc11	GCAAGTGCCGTTCACATGGAACCTTTCC	Nuclear 25S rRNA
nuc12	GGATTCCCCTTGTCCGTACCAGTTCTGA	Nuclear 25S rRNA
nuc13	AACTCCCCACCTGACAATGTCCTCCG	Nuclear 25S rRNA
nuc14	TCTAAACCCAGCTCACGTTCCCTATTGGT	Nuclear 25S rRNA

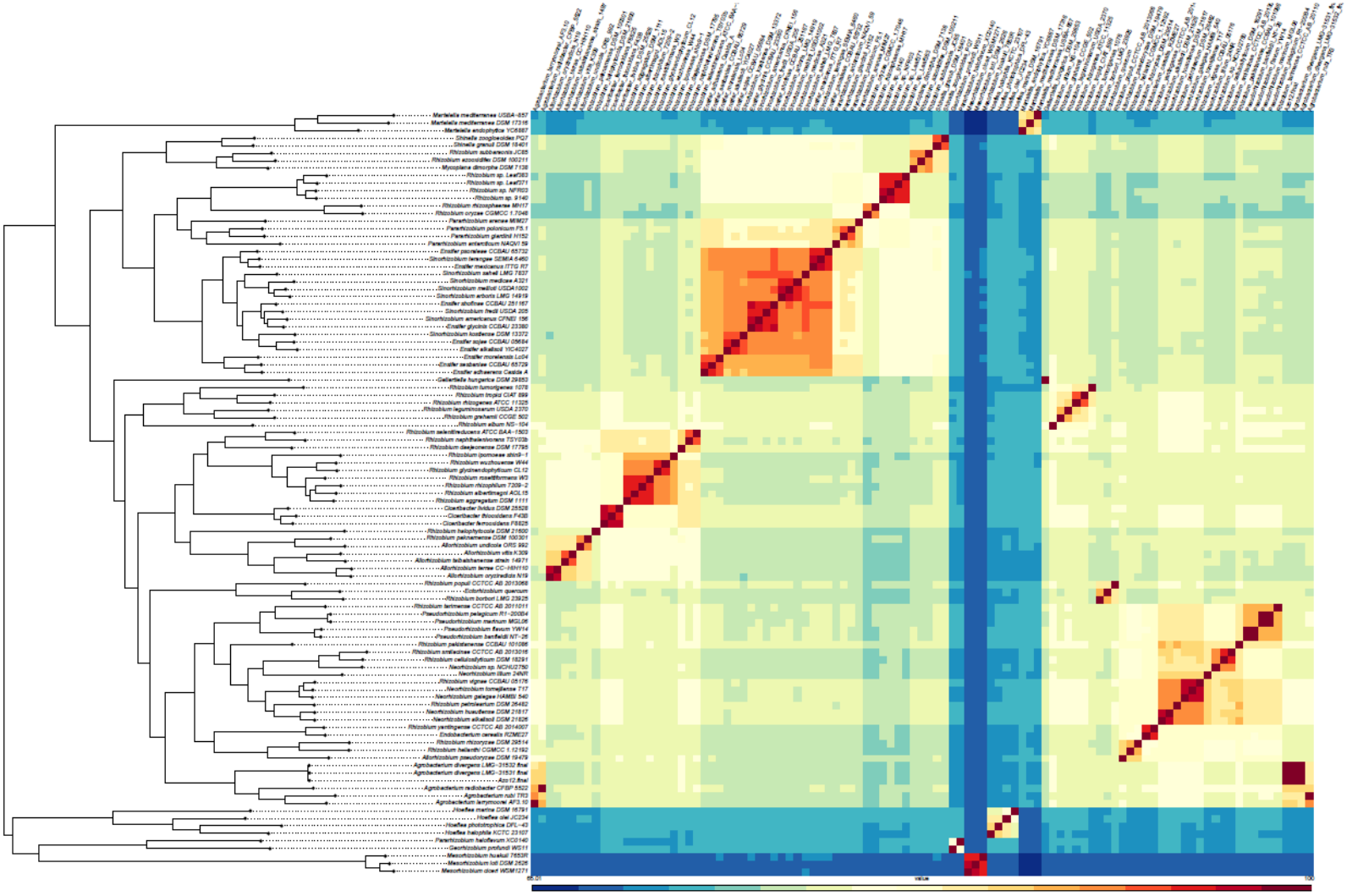

Fig. S1. Phylogenetic placement of *Agrobacterium* sp. Azo12 (Azo12 final) based on core-proteome average amino acid identity (cpAAI) analysis.

Genome assembly and annotation

```
#### Data downloaded from NCBI BioProject ...

cd /home/mgoleb/Dokumenty/Azo12_Agnieszka/Agrobacterium_divergens

~/software/sratoolkit/... fasterq-dump -3 -O . accession

conda activate ont_assembly

 mv SRR21755518.fastq.gz SRR21755518_minion.fastq.gz

 mv SRR21755519.fastq.gz SRR21755519_illumina_interleaved.fastq.gz

 mv SRR21755520.fastq.gz SRR21755520_minion.fastq.gz

 mv SRR21755521.fastq.gz SRR21755521_illumina_interleaved.fastq.gz

filtlong --min_length 1000 --keep_percent 90
SRR21755518_minion.fastq.gz > minion1.fastq
filtlong --min_length 1000 --keep_percent 90
SRR21755520_minion.fastq.gz > minion2.fastq


mkdir LMG-31531
mkdir LMG-31532

mv SRR21755518* LMG-31532
mv SRR21755519* LMG-31532
mv SRR21755520* LMG-31531
mv SRR21755521* LMG-31531
mv minion1.fastq LMG-31532
mv minion2.fastq LMG-31531

cd LMG-31531

trycycler subsample --genome_size 5.8m --count 36 --reads
minion2.fastq.gz --out_dir read_subsets

threads=16  # change as appropriate for your system
mkdir assemblies

for n in 01 02 03 04 05 06 07 08 09 10 11 12; do flye --nano-hq
read_subsets/sample_$n.fastq --threads "$threads" --out-dir assembly_$n
&& cp assembly_$n/assembly.fasta assemblies/assembly_$n.fasta && cp
assembly_$n/assembly_graph.gfa assemblies/assembly_$n.gfa && rm -r
assembly_$n; done

for n in 13 14 15 16 17 18 19 20 21 22 23 24; do
miniasm_and_minipolish.sh read_subsets/sample_$n.fastq "$threads" >
assemblies/assembly_$n.gfa && any2fasta assemblies/assembly_$n.gfa >
assemblies/assembly_$n.fasta; done

for n in 25 26 27 28 29 30 31 32 33 34 35 36; do raven --threads
"$threads" --disable-checkpoints --graphical-fragment-assembly
assemblies/assembly_$n.gfa read_subsets/sample_$n.fastq >
assemblies/assembly_$n.fasta; done > raven.log 2>&1 &
```

```
trycycler cluster --assemblies assemblies/*.fasta --reads
minion2.fastq.gz --out_dir trycycler

rm -rf trycycler/cluster_005

trycycler reconcile --reads minion2.fastq.gz --cluster_dir
trycycler/cluster_001 --max_add_seq 10000 #
trycycler reconcile --reads minion2.fastq.gz --cluster_dir
trycycler/cluster_002 --max_add_seq 10000
trycycler reconcile --reads minion2.fastq.gz --cluster_dir
trycycler/cluster_003 --max_add_seq 10000
trycycler reconcile --reads minion2.fastq.gz --cluster_dir
trycycler/cluster_004 --max_add_seq 10000

trycycler msa -c trycycler/cluster_001
trycycler msa -c trycycler/cluster_002
trycycler msa -c trycycler/cluster_003
trycycler msa -c trycycler/cluster_004

trycycler partition --reads minion2.fastq.gz --cluster_dirs
trycycler/cluster_*

trycycler consensus --cluster_dir trycycler/cluster_001
trycycler consensus --cluster_dir trycycler/cluster_002
trycycler consensus --cluster_dir trycycler/cluster_003
trycycler consensus --cluster_dir trycycler/cluster_004

for c in trycycler/cluster_*; do
    medaka_consensus -i "$c"/4_reads.fastq -d
"$c"/7_final_consensus.fasta -o "$c"/medaka -m
r1041_e82_400bps_sup_v4.2.0 -t 12
    mv "$c"/medaka/consensus.fasta "$c"/8_medaka.fasta
    rm -r "$c"/medaka "$c"/*.fai "$c"/*.mmi  # clean up
done

cat trycycler/cluster_00*/8_medaka.fasta > trycycler/consensus.fasta

ln -s trycycler/consensus.fasta ./consensus.fasta

fastp --in1 SRR21755521_1.fastq.gz --in2 SRR21755521_2.fastq.gz --out1
1.fastq.gz --out2 2.fastq.gz --unpaired1 u.fastq.gz --unpaired2
u.fastq.gz

bwa index consensus.fasta

bwa mem -t 16 -a consensus.fasta 1.fastq.gz > alignments_1.sam

bwa mem -t 16 -a consensus.fasta 2.fastq.gz > alignments_2.sam

polypolish consensus.fasta alignments_1.sam alignments_2.sam >
polypolish.fasta

polca.sh -a polypolish.fasta -r "1.fastq.gz 2.fastq.gz" -t 16 -m 1G

mv polypolish.fasta.PolcaCorrected.fa Agrobacterium_divergens_LMG-
31531_final.fasta
```

```
cd LMG-31532

trycycler subsample --genome_size 5.8m --count 36 --reads
minion1.fastq.gz --out_dir read_subsets

threads=16  # change as appropriate for your system
mkdir assemblies

for n in 01 02 03 04 05 06 07 08 09 10 11 12; do flye --nano-hq
read_subsets/sample_$n.fastq --threads "$threads" --out-dir assembly_$n
&& cp assembly_$n/assembly.fasta assemblies/assembly_$n.fasta && cp
assembly_$n/assembly_graph.gfa assemblies/assembly_$n.gfa && rm -r
assembly_$n; done

for n in 13 14 15 16 17 18 19 20 21 22 23 24; do
miniasm_and_minipolish.sh read_subsets/sample_$n.fastq "$threads" >
assemblies/assembly_$n.gfa && any2fasta assemblies/assembly_$n.gfa >
assemblies/assembly_$n.fasta; done

for n in 25 26 27 28 29 30 31 32 33 34 35 36; do raven --threads
"$threads" --disable-checkpoints --graphical-fragment-assembly
assemblies/assembly_$n.gfa read_subsets/sample_$n.fastq >
assemblies/assembly_$n.fasta; done > raven.log 2>&1 &

trycycler cluster --assemblies assemblies/*.fasta --reads
minion1.fastq.gz --out_dir trycycler

rm -rf trycycler/cluster_004/
rm -rf trycycler/cluster_005/
rm -rf trycycler/cluster_006/

trycycler reconcile --reads minion1.fastq.gz --cluster_dir
trycycler/cluster_001 --max_add_seq 10000 #
trycycler reconcile --reads minion1.fastq.gz --cluster_dir
trycycler/cluster_002 --max_add_seq 10000
trycycler reconcile --reads minion1.fastq.gz --cluster_dir
trycycler/cluster_003 --max_add_seq 10000

trycycler msa -c trycycler/cluster_001
trycycler msa -c trycycler/cluster_002
trycycler msa -c trycycler/cluster_003

trycycler partition --reads minion1.fastq.gz --cluster_dirs
trycycler/cluster_*

trycycler consensus --cluster_dir trycycler/cluster_001
trycycler consensus --cluster_dir trycycler/cluster_002
trycycler consensus --cluster_dir trycycler/cluster_003

for c in trycycler/cluster_*; do
    medaka_consensus -i "$c"/4_reads.fastq -d
"$c"/7_final_consensus.fasta -o "$c"/medaka -m
r1041_e82_400bps_sup_v4.2.0 -t 12
    mv "$c"/medaka/consensus.fasta "$c"/8_medaka.fasta
    rm -r "$c"/medaka "$c"/*.fai "$c"/*.mmi  # clean up
done

cat trycycler/cluster_00*/8_medaka.fasta > trycycler/consensus.fasta
```

```
ln -s trycycler/consensus.fasta ./consensus.fasta

fastp --in1 SRR21755519_1.fastq.gz --in2 SRR21755519_2.fastq.gz --out1
1.fastq.gz --out2 2.fastq.gz --unpaired1 u.fastq.gz --unpaired2
u.fastq.gz

bwa index consensus.fasta

bwa mem -t 16 -a consensus.fasta 1.fastq.gz > alignments_1.sam

bwa mem -t 16 -a consensus.fasta 2.fastq.gz > alignments_2.sam

polypolish consensus.fasta alignments_1.sam alignments_2.sam >
polypolish.fasta

polca.sh -a polypolish.fasta -r "1.fastq.gz 2.fastq.gz" -t 16 -m 1G #
fails as no variants are found

mv polypolish.fasta Agrobacterium_divergens_LMG-31532_final.fasta
```

Triticum aestivum reads preprocessing and mapping

```
##########################################################################

        All computations on athena.cyfronet.pl

##########################################################################


# Reads correction with rcorrector
cd ~/storage/RSWH/WH


mkdir done # directory to store processed files
### WH_rcorrector_athena.bash
#!/bin/bash -l
## Job name
#SBATCH -J larix_WH_rcorrector
## Number of allocated nodes
#SBATCH -N 1
## Number of tasks per node (by default this corresponds to the number
of cores allocated per node)
#SBATCH --ntasks-per-node=96
## Memory allocated per core (default is 5GB)
## Max task execution time (format is HH:MM:SS)
#SBATCH --time=12:00:00
## Name of grant to which resource usage will be charged
#SBATCH -A plglarix-gpu-a100
#SBATCH --mem=300GB
## Name of partition
#SBATCH -p plgrid-gpu-a100
## Name of file to which standard output will be redirected
#SBATCH --output="WH_rcorrector_07022023.out"
```

```
## Name of file to which the standard error stream will be redirected
#SBATCH --error="WH_rcorrector_07022023.err"

## change to sbatch working directory
cd $SLURM_SUBMIT_DIR

module load Java gzip

for r1 in *_R1_001.fastq.gz; do r2=`echo $r1 | sed
"s/\_R1_001\./_R2_001./"`;
/net/people/plgrid/plgmgoleb/storage/bin/run_rcorrector.pl -t 96 -1 $r1
-2 $r2 >> rcorrector.log 2>&1; mv $r1 done; mv $r2 done; done

### end RS_rcorrector_athena.bash


mkdir corrected

mv *.cor.fq.gz corrected


cd corrected


# Removing uncorrectable pairs

### run_FilterUncorrectablePEfastq.bash
#!/usr/bin/bash
      r1=$1;
      r2=`echo $r1 | sed "s/\_R1_/_R2_/"`;
      sample=`echo $r1 | sed "s/\_L00._R1_001.cor.fq.gz//"`;
      logfile="rmunfixable_$sample.log";
      srun -n 1 -N 1 /net/pr2/projects/plgrid/plgglarix/bin/python2.7
/net/pr2/projects/plgrid/plgglarix/bin/FilterUncorrectabledPEfastq.py -
1 $r1 -2 $r2 -s $sample >> FilterUncorrectablePEfastq.log 2>&1;
### end run_FilterUncorrectablePEfastq.bash

### run_gzip.bash
#!/usr/bin/bash
      file=$1;
      srun -n 1 -N 1 gzip $file;
### end run_gzip.bash

### WH_removeuncorrectable_athena.bash
#!/bin/bash -l
## Job name
#SBATCH -J larix_WH_removeuncorrectable
## Number of allocated nodes
#SBATCH -N 1
## Number of tasks per node (by default this corresponds to the number
of cores allocated per node)
#SBATCH --ntasks-per-node=96
## Memory allocated per core (default is 5GB)
## Max task execution time (format is HH:MM:SS)
#SBATCH --time=12:00:00
## Name of grant to which resource usage will be charged
#SBATCH -A plglarix-gpu-a100
#SBATCH --mem=300GB
```

```
## Name of partition
#SBATCH -p plgrid-gpu-a100
## Name of file to which standard output will be redirected
#SBATCH --output="WH_removeuncorrectable_07022023.out"
## Name of file to which the standard error stream will be redirected
#SBATCH --error="WH_removeuncorrectable_07022023.err"

## change to sbatch working directory
cd $SLURM_SUBMIT_DIR

module load gzip Perl



ls *_R1_001.cor.fq.gz | xargs -t -d "\n" -P 53 -n 1
./run_FilterUncorrectablePEfastq.bash

ls unfixrm_*.cor.fq | xargs -t -d "\n" -P 53 -n 1 ./run_gzip.bash

### end RS_removeuncorrectable_athena.bash

sbatch WH_removeuncorrectable_athena.bash


# trimming with trim_galore
mkdir trimmed_reads

### WH_trimgalore_athena.bash
#!/bin/bash -l
## Job name
#SBATCH -J larix_WH_trimgalore
## Number of allocated nodes
#SBATCH -N 20
## Number of tasks per node (by default this corresponds to the number
of cores allocated per node)
#SBATCH --ntasks-per-node=2
## Memory allocated per core (default is 5GB)
## Max task execution time (format is HH:MM:SS)
#SBATCH --time=12:00:00
## Name of grant to which resource usage will be charged
#SBATCH -A plglarix-gpu-a100
#SBATCH --mem=12GB
## Name of partition
#SBATCH -p plgrid-gpu-a100
## Name of file to which standard output will be redirected
#SBATCH --output="WH_trimgalore.out"
## Name of file to which the standard error stream will be redirected
#SBATCH --error="RS_trimgalore.err"

## change to sbatch working directory
cd $SLURM_SUBMIT_DIR

module load gzip Perl bzip2

conda activate cutadapt

for r1 in unfixrm_*_R1_001.cor.fq.gz; do
        r2=$( echo $r1 | sed "s/\_R1_/_R2_/" );
```

```
        sample=$( echo $r1 | sed "s/unfixrm\_//" | sed
"s/\_S.*\_L00.\_R1\_001\.cor\.fq\.gz//");
        while [ "$(jobs -p | wc -l)" -ge "$SLURM_NTASKS" ]; do
                sleep 30;
        done

        srun --ntasks 1 --cpus-per-task  1 trim_galore --paired --
phred33 --retain_unpaired --output_dir trimmed_reads --length 36 -q 5 -
-stringency 1 -e 0.1 $r1 $r2 > $sample\.trim_galore.log &

done

wait
### end WH_trimgalore_athena.bash

sbatch WH_trimgalore_athena.bash

cd trimmed_reads


# removing ribosomal RNA reads
mkdir nonribosomal
mkdir done

### WH_bowtie2_athena.bash
#!/bin/bash -l
## Job name
#SBATCH -J larix_WH_bowtie2
## Number of allocated nodes
#SBATCH -N 24
## Number of tasks per node (by default this corresponds to the number
of cores allocated per node)
#SBATCH --ntasks-per-node=64
## Memory allocated per core (default is 5GB)
## Max task execution time (format is HH:MM:SS)
#SBATCH --time=12:00:00
## Name of grant to which resource usage will be charged
#SBATCH -A plglarix-gpu-a100
#SBATCH --mem=48GB
## Name of partition
#SBATCH -p plgrid-gpu-a100
## Name of file to which standard output will be redirected
#SBATCH --output="WH_bowtie2.out"
## Name of file to which the standard error stream will be redirected
#SBATCH --error="WH_bowtie.err"

## change to sbatch working directory
cd $SLURM_SUBMIT_DIR

module load gzip Perl bzip2


for r1 in unfixrm_*_R1_001.cor_val_1.fq.gz; do
        r2=`echo $r1 | sed
"s/\_R1\_001\.cor\_val\_1\./_R2_001.cor_val_2./g"`;
        sample=`echo $r1 | sed "s/unfixrm\_//" | sed
"s/\_S.*\_L00.\_R1\_001\.cor\_val\_1\.fq\.gz//"`;
        echo -n $sample;
```

```
        while [ "$(jobs -p | wc -l)" -ge "$SLURM_NTASKS" ]; do
               sleep 30
        done


        srun --ntasks 1 --cpus-per-task 64
/net/pr2/projects/plgrid/plgglarix/bin/bowtie2 --quiet --nofw --very-
sensitive-local --phred33  -x
/net/pr2/projects/plgrid/plgglarix/db/silva/silva_v138 -1 $r1 -2 $r2 --
threads 64 --met-file $sample\_bowtie2_metrics.txt --al-conc-gz
ribosomal_$sample\.fq.gz --un-conc-gz
./nonribosomal/nonribosomal_$sample\.fq.gz  --al-gz
ribosomal_unpaired_$sample\.fq.gz --un-gz
./nonribosomal/nonribosomal_unpaired_$sample\.fq.gz >
$sample\.bowtie2.log 2>&1 &

        echo " done";
done

wait
### end WH_bowtie2_athena.bash

sbatch WH_bowtie2_athena.bash

# mapping to wheat genome on Athena

mkdir ~/storage/RSWH/WH/all_clean
cd ~/storage/RSWH/WH/all_clean
ln -s ~/storage/RSWH/WH/corrected/trimmed_reads/nonribosomal/* .
mkdir mapped
mkdir unmapped
mkdir done

### WH_hisat2_athena.bash
#!/bin/bash -l
## Job name
#SBATCH -J larix_WH_hisat2
## Number of allocated nodes
#SBATCH -N 24
## Number of tasks per node (by default this corresponds to the number
of cores allocated per node)
#SBATCH --ntasks-per-node=80
## Memory allocated per core (default is 5GB)
## Max task execution time (format is HH:MM:SS)
#SBATCH --time=12:00:00
## Name of grant to which resource usage will be charged
#SBATCH -A plglarix-gpu-a100
#SBATCH --mem=48GB
## Name of partition
#SBATCH -p plgrid-gpu-a100
## Name of file to which standard output will be redirected
#SBATCH --output="WH_hisat2.out"
## Name of file to which the standard error stream will be redirected
#SBATCH --error="WH_hisat2.err"

## change to sbatch working directory
cd $SLURM_SUBMIT_DIR
```

```
module load gzip Perl bzip2


for r1 in nonribosomal_*\.fq.1.gz; do
        r2=$(echo $r1 | sed "s/\.1/.2/g");
        sample=$(echo $r1 | sed "s/nonribosomal_//" | sed
"s/\.fq\.1\.gz//");
        bam=$sample\_hisat2.bam;
        echo -n $sample;
        while [ "$(jobs -p | wc -l)" -ge "$SLURM_NTASKS" ]; do
               sleep 30
        done


        srun --nodes 1 --ntasks 1 --cpus-per-task 80
/net/pr2/projects/plgrid/plgglarix/bin/hisat2 -p 72 --rna-strandness RF
--phred33 --dta -x
/net/people/plgrid/plgmgoleb/storage/db/triticum_aestivum/Triticum_aest
ivum -1 $r1 -2 $r2 --un-gz ./unmapped/$sample\_single.fq.gz --un-conc-
gz ./unmapped/$sample\.fq.gz |
/net/pr2/projects/plgrid/plgglarix/bin/samtools view -b -@ 16 -o $bam -
&

        echo " done";
done

wait
### end WH_hisat2_athena.bash

sbatch WH_hisat2_athena.bash

mv *.bam mapped/
cd mapped

# sorting alignments according to their position in the genome
### WH_samtools_sort_athena.bash
#!/bin/bash -l
## Job name
#SBATCH -J larix_WH_hisat2
## Number of allocated nodes
#SBATCH -N 12
## Number of tasks per node (by default this corresponds to the number
of cores allocated per node)
#SBATCH --ntasks-per-node=36
## Memory allocated per core (default is 5GB)
## Max task execution time (format is HH:MM:SS)
#SBATCH --time=12:00:00
## Name of grant to which resource usage will be charged
#SBATCH -A plglarix-gpu-a100
#SBATCH --mem=48GB
## Name of partition
#SBATCH -p plgrid-gpu-a100
## Name of file to which standard output will be redirected
#SBATCH --output="WH_samtools_sort.out"
## Name of file to which the standard error stream will be redirected
#SBATCH --error="WH_samtools_sort.err"

## change to sbatch working directory
```

```
cd $SLURM_SUBMIT_DIR

module load gzip Perl bzip2


for r1 in nonribosomal_*\.fq.1.gz; do
        sample=$(echo $bam | sed "s/\_hisat2\.bam//");
        bamsorted=$sample\_hisat2_sorted.bam;
        while [ "$(jobs -p | wc -l)" -ge "$SLURM_NTASKS" ]; do
               sleep 30
        done

      srun --nodes 1 --ntasks 1 --cpus-per-task 36
/net/pr2/projects/plgrid/plgglarix/bin/samtools sort -@ 35 -o
$bamsorted $bam  &

done

wait

### end WH_samtools_sort_athena.bash

sbatch WH_samtools_sort_athena.bash



# first stringtie run - assembles individual samples guided by wheat
genome

### WH_stringtie1_athena.bash
#!/bin/bash -l
## Job name
#SBATCH -J larix_WH_hisat2
## Number of allocated nodes
#SBATCH -N 12
## Number of tasks per node (by default this corresponds to the number
of cores allocated per node)
#SBATCH --ntasks-per-node=16
## Memory allocated per core (default is 5GB)
## Max task execution time (format is HH:MM:SS)
#SBATCH --time=6:00:00
## Name of grant to which resource usage will be charged
#SBATCH -A plglarix-gpu-a100
#SBATCH --mem=48GB
## Name of partition
#SBATCH -p plgrid-gpu-a100
## Name of file to which standard output will be redirected
#SBATCH --output="WH_stringtie1.out"
## Name of file to which the standard error stream will be redirected
#SBATCH --error="WH_stringtie1.err"
## change to sbatch working directory
cd $SLURM_SUBMIT_DIR

module load gzip Perl bzip2


for bam in *_hisat2\_sorted\.bam; do
        sample=$(echo $bam | sed "s/\_hisat2\_sorted\.bam//");
```

```
        gtf=$sample\_stringtie.gtf;
        while [ "$(jobs -p | wc -l)" -ge "$SLURM_NTASKS" ]; do
               sleep 30
        done

      srun --nodes 1 --ntasks 1 --cpus-per-task 16
/net/pr2/projects/plgrid/plgglarix/bin/stringtie $bam -p 16 -o $gtf --
rf -G
/net/people/plgrid/plgmgoleb/storage/db/triticum_aestivum/Triticum_aest
ivum.IWGSC.56.gff3 -m 150 > $sample\_stringtie.log 2>&1  &

done

wait


### end WH_stringtie1_athena.bash


# merging all assemblies from all samples
ls *.gtf > assemblies_list.txt

### WH_stringtie_merge_athena.bash
#!/bin/bash -l
## Job name
#SBATCH -J larix_WH_stringie_merge
## Number of allocated nodes
#SBATCH -N 1
## Number of tasks per node (by default this corresponds to the number
of cores allocated per node)
#SBATCH --ntasks-per-node=16
## Memory allocated per core (default is 5GB)
## Max task execution time (format is HH:MM:SS)
#SBATCH --time=2:00:00
## Name of grant to which resource usage will be charged
#SBATCH -A plglarix-gpu-a100
#SBATCH --mem=128GB
## Name of partition
#SBATCH -p plgrid-gpu-a100
## Name of file to which standard output will be redirected
#SBATCH --output="WH_stringtie_merge.out"
## Name of file to which the standard error stream will be redirected
#SBATCH --error="WH_stringtie_merge.err"
## change to sbatch working directory
cd $SLURM_SUBMIT_DIR

module load gzip Perl bzip2


srun --nodes 1 --ntasks 1 --cpus-per-task 16
/net/pr2/projects/plgrid/plgglarix/bin/stringtie --merge -p 16 -i -G
/net/people/plgrid/plgmgoleb/storage/db/triticum_aestivum/Triticum_aest
ivum.IWGSC.56.gff3 -o WH_TaeIWGSC_stringtie.gtf assemblies_list.txt

### end WH_stringie_merge_athena.bash

sbatch WH_stringtie_merge_athena.bash
```

```
# generating input tables for Ballgown

### WH_stringtie2_athena.bash
#!/bin/bash -l
## Job name
#SBATCH -J larix_WH_stringtie2
## Number of allocated nodes
#SBATCH -N 24
## Number of tasks per node (by default this corresponds to the number
of cores allocated per node)
#SBATCH --ntasks-per-node=24
## Memory allocated per core (default is 5GB)
## Max task execution time (format is HH:MM:SS)
#SBATCH --time=2:00:00
## Name of grant to which resource usage will be charged
#SBATCH -A plglarix-gpu-a100
#SBATCH --mem=48GB
## Name of partition
#SBATCH -p plgrid-gpu-a100
## Name of file to which standard output will be redirected
#SBATCH --output="WH_stringtie2.out"
## Name of file to which the standard error stream will be redirected
#SBATCH --error="WH_stringtie2.err"
## change to sbatch working directory
cd $SLURM_SUBMIT_DIR

module load gzip Perl bzip2


for bam in *_hisat2\_sorted\.bam; do
        sample=$(echo $bam | sed "s/\_hisat2\_sorted\.bam//");
        gtf=$sample\_stringtie.gtf;
        while [ "$(jobs -p | wc -l)" -ge "$SLURM_NTASKS" ]; do
               sleep 30
        done

      srun --nodes 1 --ntasks 1 --cpus-per-task 24
/net/pr2/projects/plgrid/plgglarix/bin/stringtie $bam -p 24 -B -e -o
ballgown\_$sample/$gtf --rf -G WH_TaeIWGSC_stringtie.gtf -m 150 >
$sample\_stringtie.log 2>&1  &

done

wait

### end WH_stringtie2_athena.bash

sbatch WH_stringtie2_athena.bash


######################################################################


         Annotation of stringtie-assembled transcripts


######################################################################
```

```
gffread -g
~/db/triticum_aestivum/Triticum_aestivum.IWGSC.dna_sm.toplevel.fa -w
stringtie_transcripts.fasta WH_TaeIWGSC_stringtie.gtf

gffread  --table  @geneid,@id stringtie_transcripts.gff3 >
gene_trans_map.tsv

diamond blastx -p 30 --db ~/db/uniprot/uniprot.dmnd --out
stringtie_transcripts.diamond.outfmt6 --outfmt 6 --query
stringtie_transcripts.fasta --evalue 0.000001 --max-target-seqs 1 >
diamond.log 2>&1 &


# TransDecoder

TransDecoder.LongOrfs -t stringtie_transcripts.fasta

hmmscan --cpu 4 --domtblout stringtie_transcripts.pfamout ~/db/Pfam-
A/Pfam-A.hmm stringtie_transcripts.longorfs.pep > hmmscan.log 2>&1 &

TransDecoder.Predict -t stringtie_transcripts.fasta --single_best_only
-T 1000 --retain_pfam_hits stringtie_transcripts.pfamout

diamond blastx -p 30 --db ~/db/ncbi_nr/ncbi_nr.dmnd --out
stringtie_transcripts.diamond_ncbi.outfmt6.blastx --outfmt 6 --query
stringtie_transcripts.fasta --evalue 0.000001 --max-target-seqs 1 >
diamond1.log 2>&1 &

diamond blastp -p 30 --db ~/db/ncbi_nr/ncbi_nr.dmnd --out
stringtie_transcripts.fasta.transdecoder.pep.diamond_ncbi.outfmt6.blast
p --outfmt 6 --query stringtie_transcripts.fasta.transdecoder.pep --
evalue 0.000001 --max-target-seqs 1 > diamond2.log 2>&1 &

diamond blastp -p 30 --db ~/db/uniprot/uniprot.dmnd --out
stringtie_transcripts.fasta.transdecoder.pep.sprot.outfmt6.blastp --
outfmt 6 --query stringtie_transcripts.fasta.transdecoder.pep --evalue
0.000001 --max-target-seqs 1 > diamond3.log 2>&1 &


######################################################################


           Differential expression analysis with Ballgown


######################################################################

# Data were copied to /home/mgoleb/Dokumenty/pszenica_Agnieszka on
158.75.106.101

cd /home/mgoleb/Dokumenty/pszenica_Agnieszka

### preparation of annotation data: files were downloaded from
EnsemblPlants Biomart and placed in /home/mgoleb/db/triticum_aestivum
```

```
# GO terms (columns 'Gene stable ID', 'Transcript stable ID' and 'GO
term accession' were chosen in BioMart, gzipped tsv file was downloaded
to /home/mgoleb/db/triticum_aestivum and unpacked)
go_terms <- read.table("Triticum_aestivum.IWGSC.GO_terms.tsv",
sep="\t", header=T)
go_terms_final <- data.frame(
                        transcript_id=aggregate(GO.term.accession ~
Transcript.stable.ID, data=go_terms, FUN=function(x) paste(x,
collapse=','))$Transcript.stable.ID,
                        go_terms=sapply(sapply(strsplit(gsub("^,", "",
aggregate(GO.term.accession ~ Transcript.stable.ID, data=go_terms,
FUN=function(x) paste(x, collapse=','))$GO.term.accession), "\\,"),
unique), function(x) paste(x, collapse=","))
                        )
write.table(go_terms_final,
"Triticum_aestivum.IWGSC.GO_terms_final.tsv", sep="\t")

# annotation (columns 'Gene stable ID', 'Pfam ID', 'UniProtKB/Swiss-
Prot ID', 'NCBI gene (formerly Entrezgene) ID', 'Interpro ID',
'Interpro Description' and 'Gene description' were chosen in BioMart,
gzipped tsv file was downloaded, unpacked and "'" were removed in vi
(:%s/\'//g))
annot <- read.table("Triticum_aestivum.IWGSC.annot.tsv", sep="\t",
header=T)
annot_final <- data.frame(
                        gene_id=aggregate(Pfam.ID ~ Gene.stable.ID,
data=annot, FUN=function(x) paste(x, collapse=','))$Gene.stable.ID,
                        pfam_id=sapply(sapply(strsplit(gsub("^,", "",
aggregate(Pfam.ID ~ Gene.stable.ID, data=annot, FUN=function(x)
paste(x, collapse=','))$Pfam.ID), "\\,"), unique), function(x) paste(x,
collapse=',')),
                        interpro_id=sapply(sapply(strsplit(gsub("^,", "",
aggregate(Interpro.ID ~ Gene.stable.ID, data=annot, FUN=function(x)
paste(x, collapse=','))$Interpro.ID), "\\,"), unique), function(x)
paste(x, collapse=',')),

     gene_description=sapply(sapply(strsplit(gsub("^;", "",
aggregate(Gene.description ~ Gene.stable.ID, data=annot,
FUN=function(x) paste(x, collapse=';'))$Gene.description), "\\;"),
unique), function(x) paste(x, collapse=';')),

     interpro_description=sapply(sapply(strsplit(gsub("^;", "",
aggregate(Interpro.Description ~ Gene.stable.ID, data=annot,
FUN=function(x) paste(x, collapse=';'))$Interpro.Description), "\\;"),
unique), function(x) paste(x, collapse=';'))
)
write.table(annot_final, "Triticum_aestivum.IWGSC.annot_final.tsv",
sep="\t")

# Conversion of GFF3 to GTF3 with AGAT (conda activate base - it is
installed with conda install -c bioconda agat)
agat_convert_sp_gff2gtf.pl --gff Triticum_aestivum.IWGSC.56.gff3 -o
triticum_ref.gtf

R

# Analysis in: pszenica_triticum_Agnieszka.R (mapping to wheat genome)
and pszenica_Azo12_Agnieszka.R (mapping to Azo12 genome)
```

```
      hisat2 -p 24 --rna-strandness RF --phred33 --dta -x
/home/mgoleb/db/Azo12/Azo12 -1 $r1 -2 $r2 -S
./mapped/$sample\_hisat2.sam --un-gz ./unmapped/$sample\_single.fq.gz -
-un-conc-gz ./unmapped/$sample\.fq.gz > $sample\_hs2.log 2>&1;

######################################################################


           Mapping to Azo12 genome      on 158.75.106.101


######################################################################

cd /home/mgoleb/db/Azo12

conda activate base
agat_convert_sp_gff2gtf.pl --gff Azo12_final_PROKKA.gff -o
Azo12_final_PROKKA.gtf
conda deactivate

cd /home/mgoleb/Dokumenty/pszenica_Agnieszka/all_clean/Azo12
mv ../nonribosomal_*.fq.gz .

for r1 in nonribosomal_*\.fq.1.gz; do
        r2=$(echo $r1 | sed "s/\.1/.2/g");
        sample=$(echo $r1 | sed "s/nonribosomal_//" | sed
"s/\.fq\.1\.gz//");
        bam=$sample\_hisat2.bam;
        echo $sample;
        echo $r1 $r2 $bam;

      hisat2 -p 24 --rna-strandness RF --phred33 --dta -x
/home/mgoleb/db/Azo12/Azo12_final -1 $r1 -2 $r2 -S
./mapped/$sample\_hisat2.sam --un-gz ./unmapped/$sample\_single.fq.gz -
-un-conc-gz ./unmapped/$sample\.fq.gz > $sample\_hs2.log 2>&1;

        echo " done";
done &

for sam in *_hisat2.sam; do echo $sam; bam=`echo $sam | sed
"s/\.sam/.bam/"`; if [[ ! -f "$bam" ]]; then samtools view -@ 5 -Su
$sam | samtools sort -@ 7 -o $bam -; echo $bam; fi; done



for bam in *_hisat2\.bam; do
        sample=$(echo $bam | sed "s/\_hisat2\.bam//");
        gtf=$sample\_stringtie.gtf;

      stringtie $bam -p 24 -B -e -o ballgown\_$sample/$gtf --rf -G
/home/mgoleb/db/Azo12/Azo12_final_PROKKA.gtf -m 150 >
$sample\_stringtie.log 2>&1  &

done
```

Differentially expressed genes (DEGs) and transcripts (DETs) identification

```
library(ballgown)
library(genefilter)
library(dplyr)
library(RSkittleBrewer)
library(rtracklayer)
library(trinotateR)
library(thacklr)
library(mixOmics)
library(vegan)


# importing samples data

pszenica_design <- read.table("pszenica_design.csv", header=T, sep=",")
rownames(pszenica_design) <- pszenica_design$id


sampledata <- pszenica_design
rownames(sampledata) <- sub("ballgown_", "", rownames(sampledata))
sampledata <- sampledata[ sampledata$sterility == FALSE, ]
sampledata <- sampledata[ sampledata$strain != 'W', ]


### producing ballgown object

pszenica_bg <- ballgown( dataDir="./ballgown", samplePattern="ballgown_",
pData=pszenica_design )
pszenica_wlasciwa_bg <- subset(pszenica_bg, "sterility == FALSE",
genomesubset=F)
pszenica_wlasciwa_bg <- subset(pszenica_wlasciwa_bg, "strain != 'W'",
genomesubset=F)

pszenica_texpr <- texpr(pszenica_wlasciwa_bg)
pszenica_texpr <- as.data.frame(t(pszenica_texpr))
rownames(pszenica_texpr) <- sub("FPKM.ballgown_", "",
rownames(pszenica_texpr))
pszenica_texpr_nonzero <- pszenica_texpr[, colSums(pszenica_texpr) > 0 ]
pszenica_texpr_nonzero.nzv <- nearZeroVar( pszenica_texpr_nonzero )
pszenica_texpr_nonzero.lv <- pszenica_texpr_nonzero[ , -
pszenica_texpr_nonzero.nzv$Position ]


### Importing annotation

# genes descriptions


# GO terms

set.seed(667)
# Beta-diversity on Morisita-Horn and Bray-Curtis distances
for( organ in c('S', 'R') ){
        for( NaCl in c(TRUE, FALSE) ){
                message(paste(organ, as.character(NaCl)));
                otutable <- paste0('pszenica.', organ, '.',
as.character(NaCl));
                envdata <- paste0('sdata.', organ, '.', as.character(NaCl));
```

```
                assign( otutable, pszenica_texpr_nonzero.lv[
sampledata$organ == organ & sampledata$NaCl == NaCl, ] );
                assign( otutable, get(otutable)[, colSums(get(otutable)) >
0] )
                message("exprtable");
                assign( envdata, sampledata[ sampledata$organ == organ &
sampledata$NaCl == NaCl, ] );
                message("envdata")
                for( distance in c('horn', 'bray') ){

                                ordnmds <- paste0(otutable, '.nmds');
                                orddbrda <- paste0(otutable, '.', distance,
'.dbrda');
                                filenmds <- paste0(otutable, '.', distance,
'.nmds.svg');
                                filedbrda <- paste0(otutable, '.', distance,
'.dbrda.svg');
                                ordadonis <- paste0(otutable, '.', distance,
'.adonis');
                                ordbetadisper <- paste0(otutable, '.',
distance, '.betadisper');
                                orddbrdaanova <- paste0(orddbrda, '.anova');
                                ordbetadisperanova <- paste0(ordbetadisper,
'.anova');


                                if( length(rownames(get(envdata))) > 3 ){
                                        assign( ordnmds,
metaMDS(get(otutable), dist=distance, k=2, try=100, trymax=200) );
                                        print("NDMS");
                                        assign( ordadonis,
adonis2(get(otutable) ~ strain, data=get(envdata), permu=999) );
                                        print("PERMANOVA");
                                        assign( ordbetadisper,
betadisper(vegdist(get(otutable), method=distance), get(envdata)$strain,
bias.adjust=T) );
                                        print("betadisper");
                                        assign( ordbetadisperanova,
anova(get(ordbetadisper), permu=999) );
                                        print("betadisper anova");
                                        assign( orddbrda, dbrda(get(otutable)
~ strain, data=get(envdata), dist=distance) );
                                        print("dbRDA")
                                        assign( orddbrdaanova,
anova(get(orddbrda), permu=999) );
                                        print( "dbRDA anova");

#                                       explvar_genotype <-
paste0(sprintf("%.2f", 100 * get(ordvarpart)$part$indfract[1,3]), '%');
#                                       explvar_inoculation <-
paste0(sprintf("%.2f", 100 * get(ordvarpart)$part$indfract[3,3]), '%');
                                        betadisperf <- sprintf( "%.2f",
get(ordbetadisperanova)$"F value"[1] );
                                        betadisperp <- sprintf( "%.2e",
get(ordbetadisperanova)$"Pr(>F)"[1] );
                                        adonisf <- sprintf( "%.2f",
get(ordadonis)$F[1] );
                                        adonisp <- sprintf( "%.3f",
get(ordadonis)$"Pr(>F)"[1] );
```

```
                                        title <- paste0( "PERMANOVA F = ",
adonisf, ", p = ", adonisp, "\nbetadisper F = ", betadisperf, ", p = ",
betadisperp);

                                        palette("default");
                                        svg( filenmds, width=3.5, height=3.5,
pointsize=6);
                                        par(mar=c(4,4,2,4)+0.1, mfrow=c(1,1),
las=2);
                                        ordiplot(get(ordnmds), type='none',
main=title);
                                        points( get(ordnmds),
pch=20+as.numeric(as.factor(get(envdata)$strain)), col='black',
bg=as.numeric(as.factor(get(envdata)$strain)) );
                                        ordiellipse(get(ordnmds),
display='sites', groups=as.factor(get(envdata)$strain), label=T);
                                        legend('topright', title="Strain",
legend=levels(as.factor(get(envdata)$strain)),
pch=20+sort(unique(as.numeric(as.factor(get(envdata)$strain)))),
pt.bg=sort(unique(as.numeric(as.factor(get(envdata)$strain)))));
                                        dev.off();

                                        betadisperf <- sprintf( "%.2f",
get(ordbetadisperanova)$"F value"[1] );
                                        betadisperp <- sprintf( "%.2e",
get(ordbetadisperanova)$"Pr(>F)"[1] );
                                        dbrdaanovaf <- sprintf( "%.2f",
get(orddbrdaanova)$F[1] );
                                        dbrdaanovap <- sprintf( "%.3f",
get(orddbrdaanova)$"Pr(>F)"[1] );

                                        title <- paste0( "ANOVA F = ",
dbrdaanovaf, ", p = ", dbrdaanovap, "\nbetadisper F = ", betadisperf, ", p
= ", betadisperp);

                                        svg( filedbrda, width=3.5,
height=3.5, pointsize=6);
                                        par(mar=c(4,4,2,4)+0.1, mfrow=c(1,1),
las=2);
                                        ordiplot(get(orddbrda), type='none',
main=title);
                                        points( get(orddbrda),
pch=20+as.numeric(as.factor(get(envdata)$strain)), col='black',
bg=as.numeric(as.factor(get(envdata)$strain)) );
                                        ordiellipse(get(orddbrda),
display='sites', groups=as.factor(get(envdata)$strain), label=T);
                                        legend('topright', title="Strain",
legend=levels(as.factor(get(envdata)$strain)),
pch=20+sort(unique(as.numeric(as.factor(get(envdata)$strain)))),
pt.bg=sort(unique(as.numeric(as.factor(get(envdata)$strain)))));
                                        dev.off();
                                } else {
                                        message("Mniej niz 3 probki");
                        }
                }
        }
}


### DEGs and DETs identification
```

```
pszenica_bg_highvar <- subset(pszenica_wlasciwa_bg,
"rowVars(texpr(pszenica_wlasciwa_bg)) > 1", genomesubset=T)
result <- stattest(pszenica_bg_highvar, feature='transcript', meas='FPKM',
getFC=T, covariate='organ', adjustvars=c("NaCl", "strain"))


for( salinity in c(TRUE, FALSE) ){
        for( organ in c('S', 'R') ){
                for( comp in c( 'C_A' ) ){
                        cult1 <- strsplit(comp, '_')[[1]][1];
                        cult2 <- strsplit(comp, '_')[[1]][2];
                        bg <- paste0('pszenica_bg_highvar.salinity',
as.character(salinity), '.organ', organ,  '_', comp);
                        file_t <- paste0(bg,
".Inoculation.DEtranscripts.significant.csv");
                        file_g <- paste0(bg,
".Inoculation.DEgenes.significant.csv");
                        transDE <- paste0(bg, ".Inoculation.DEtranscripts");
                        string <- paste0('NaCl == ', salinity, ' & organ ==
"', organ, '" & (strain == "', cult1, '" | strain == "', cult2, '")')
                        message(string)
                        assign(bg, subset(pszenica_bg_highvar, string,
genomesubset=F) )
                        result <- stattest(get(bg), feature='transcript',
meas='FPKM', getFC=T, covariate="strain");
                        message( transDE );
#                       result <- merge( result,
soltu_stringtie_annot_transcripts, by.x='id', by.y='t_id', all.x=T)
#                       assign(transDE, merge( result,
soltu_stringtie_annot_transcripts, by.x='id', by.y='t_id', all.x=T));
                        resultsignificant <- result[ !(is.na(result$qval)) &
result$qval < 0.05 & (!is.na(result$fc) & (result$fc > 2 | result$fc <
0.5)), ];
                        write.table(resultsignificant, file_t, sep="\t",
col.names=NA);
                        message("Transcripts done");
                        result <- stattest(get(bg), feature='gene',
meas='FPKM', getFC=T, covariate="strain");
                        resultsignificant <- result[ !(is.na(result$qval)) &
result$qval < 0.05 & (!is.na(result$fc) & (result$fc > 2 | result$fc <
0.5)), ];
                        write.table(resultsignificant, file_g, sep="\t",
col.names=NA);
                        message( "Genes done" );
                        }
                }

        }
```