Guest Lecture

Bodo Linz

09/13/18

Linz *et al. BMC Genomics* (2016) 17:767
DOI 10.1186/s12864-016-3112-5

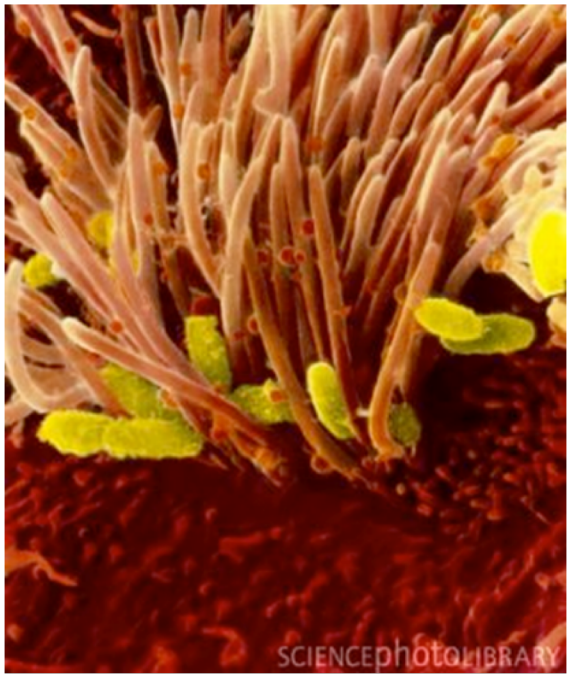
BMC Genomics

RESEARCH ARTICLE Open Access

Acquisition and loss of virulence-associated factors during genome evolution and speciation in three clades of *Bordetella* species

Bodo Linz[1*†], Yury V. Ivanov[1†], Andrew Preston[2], Lauren Brinkac[3], Julian Parkhill[4], Maria Kim[3], Simon R. Harris[4], Laura L. Goodfield[1], Norman K. Fry[5], Andrew R. Gorringe[6], Tracy L. Nicholson[7], Karen B. Register[7], Liliana Losada[3] and Eric T. Harvill[1,8,9*]

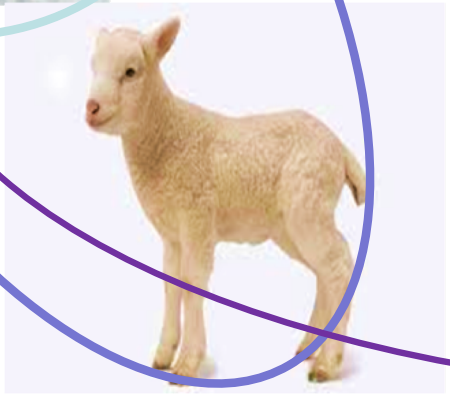
The Bordetellae

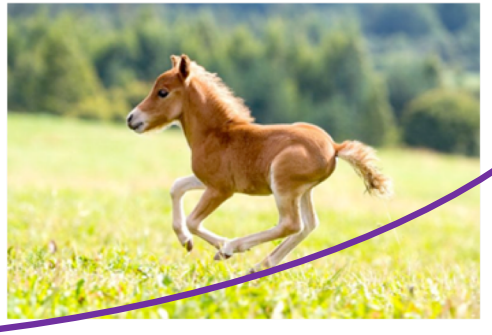


Beta-Proteobacteria
Include the classical bordetellae:
- *B. bronchiseptica*
- *B. parapertussis*
- *B. pertussis*

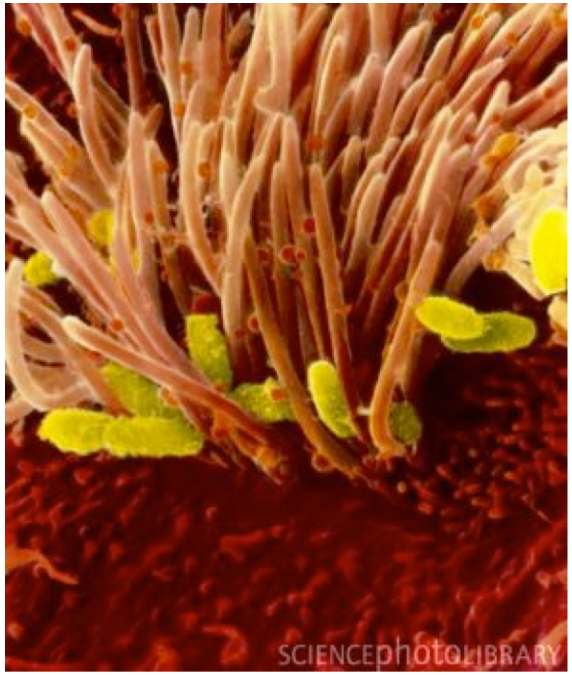






The Bordetellae



- Include the classical bordetellae:
  - *B. bronchiseptica*
  - *B. parapertussis*
  - *B. pertussis*

- Non-classical:
  - *B. holmesii*
  - *B. hinzii*
  - *B. avium*

  respiratory pathogens in animals and in immuno-compromized humans

  - *B. trematum*
  - *B. ansorpii*

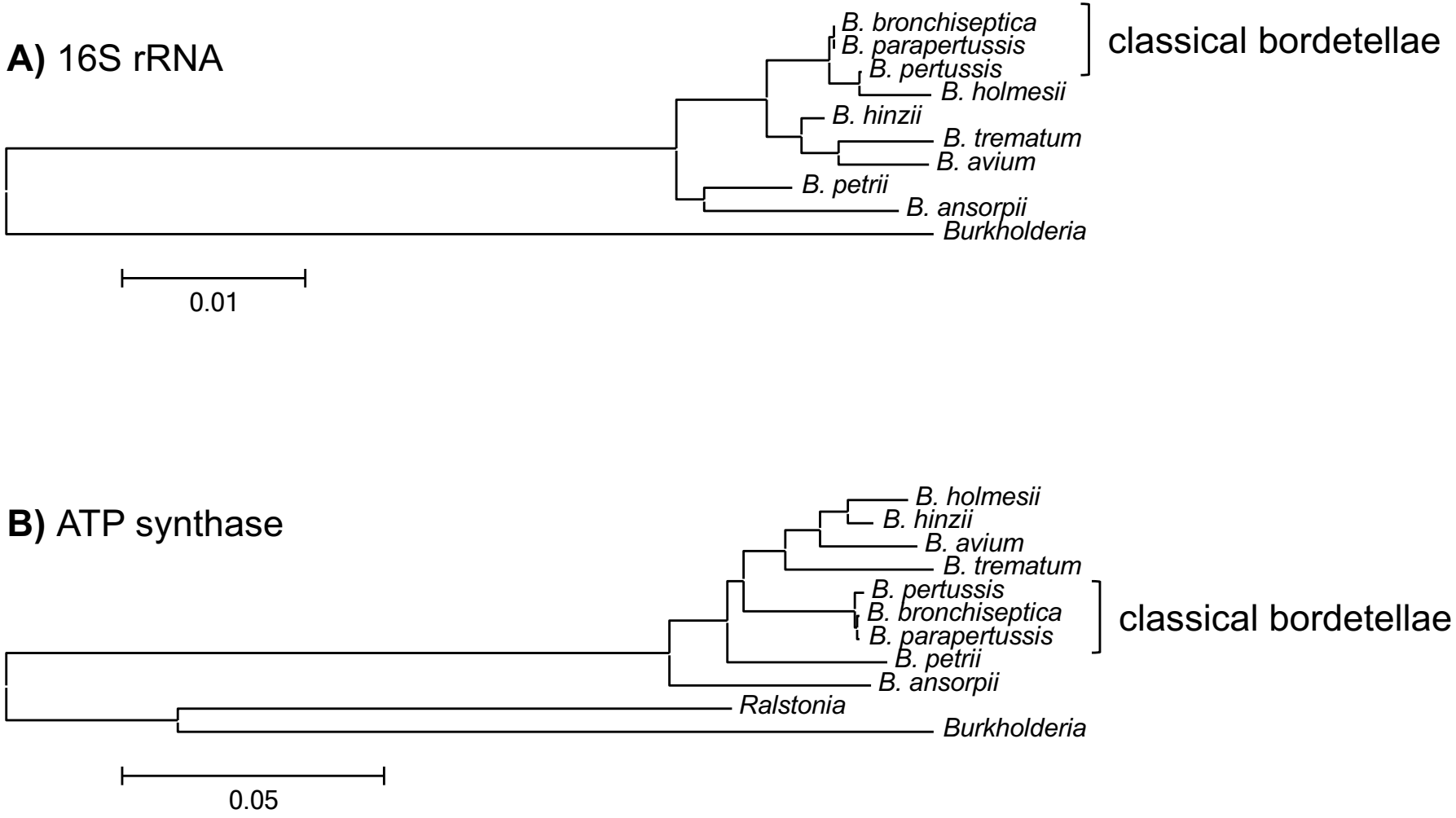
  wound and ear infection in humans

  - *B. petrii*

  environmental / ear infection in humans

+ several other recently described species

Neighbor-joining trees of 16S rRNA gene sequences and 8 concatenated ATP synthase proteins from *Bordetella*



128 Bordetella genomes

95 classical bordetellae:

- 58 *B. bronchiseptica*
- 2 *B. parapertussis*
- 34 *B. pertussis*

respiratory pathogens in animals and humans

34 non-classical bordetellae:

- 18 *B. holmesii*
- 6 *B. hinzii*
- 1 *B. avium*

respiratory pathogens in animals and in immuno-compromized humans

- 4 *B. trematum*
- 2 *B. ansorpii*

wound and ear infection in humans

- 3 *B. petrii*

environmental / ear infection in humans

questions

- virulence-associated factors determining host specificity?
- virulence-associated factors determining disease outcome?

Approach

- genome-wide SNP-based phylogenetic tree
- genome-wide presence/absence of genes
  - similar evolutionary trends?
- Pairwise genome comparisons (ACT)
  (Artemis Comparison Tool)
- mapping of virulence-associated genes
- Principle Components Analysis (PCA)

ACT: https://www.sanger.ac.uk/science/tools/artemis-comparison-tool-act

Approach

genome-wide SNP-based phylogenetic tree

- align genomes
  - align short reads against reference genome (SSAHA)
  - alignment of multiple genomes
- generate phylogenetic tree

Approach

data format: Sequence alignment in rows
Name SEQUENCE
Name SEQUENCE

```
SAMPLE01C  CGTTGCTGGCCGGATTTGCGCAGCAGGCGCGCGATCTCGTGGTCGTGCGCATTGACGCCCGCCCGCGCATCGACCAGGAACACCAC
SAMPLE02A  CGCTGCTGGCCGGATTTGCGCAGCAGGCGCGCGATCTCGTGGTCGTGCGCATTGACGCCCGCCCGCGCATCGACCAGGAACACCAC
SAMPLE03T  CGCTGCTGGCCGGACTTGCGCAGCAGGCGCGCGATCTCGTGGTCGTGCGCATTGACGCCCGCCCGCGCATCGACCAGGAACACCAC
SAMPLE-04  CGCTGCTGGCCAGATTTACGGAGC----------TTTCGTGGTCGTGCGCGTTGACGCCGGCGCGCGCGTCGACCAGGAACACCAC
SAMPLE05G  CGCTGCTGGCCGGATTTGCGCAGCAGGCGGGCGATTTCGTGGTCGTGCGCGTTGATGCCGGCACGGGCATCGACCAGGAACACGAC
SAMPLE06   CGCTGCTGGCCGGACTTGCGCAGCAGGCGGGCGATCTCGTGGTCATGCGCGTTGATCCCCGCCCGCGCGTCGACCAGGAAGACCAC
SAMPLE-7A  CGCTGCTGACCGGACTTACGCAG------------------------------------------------------------
SAMPLE08B  CGCTGCTGGCCGGACTTGCGCAACAAGCGGGCGAT----------------------CGGCCCGGGCGTCGACCAGGAACACCAC
SAMPLE09   CGCTGCTGCCCGGACTTGCGCAACAGGCGGGCGAT-----------------------------------------ACACCAC
```

Data format: 1 reference genome (5.3 MB), all other genomes aligned against it
Problem: missing data (dashes)
- gene not present
- gene so divergent that the sequence did not align
- multiple copies of a gene

Solution: remove all positions with missing data in any of the genomes

Approach

data format: Sequence alignment in rows

Name SEQUENCE

Name SEQUENCE

$1 $2 $1 = field 1; $2 = field 2

```
SAMPLE01C  CGTTGCTGGCCGGATTTGCGCAGCAGGCGCGCGATCTCGTGGTCGTGCGCATTGACGCCCGCCCGCGCATCGACCAGGAACACCAC
SAMPLE-04  CGCTGCTGGCCAGATTTACGGAGC----------TTTCGTGGTCGTGCGCGTTGACGCCGGCGCGCGCGTCGACCAGGAACACCAC
SAMPLE05G  CGCTGCTGGCCGGATTTGCGCAGCAGGCGGGCGATTTCGTGGTCGTGCGCGTTGATGCCGGCACGGGCATCGACCAGGAACACGAC
SAMPLE-7A  CGCTGCTGACCGGACTTACGCAG-----------------------------------------------------------
```

- **`awk`: change strain names to lower case and replace '-' by '_'**
- **`python`: replace nucleotides by nucleotides plus tab**
- **`awk`: remove extra tab at the end of each line**
- **`python`: transpose rows to columns**
- **`awk`: select only core loci**
- **`grep | wc`: determine the number of loci in the resulting file**
- **`python`: replace nucleotides by numbers**
- **`R:` calculate matrix**
- **`python`: transpose columns to rows**
- **`awk`: add extra tab at the end of each line**
- **`python`: replace nucleotides plus tab by nucleotides**

Approach

data format: Sequence alignment in rows

Name SEQUENCE

Name SEQUENCE

$1 $2 $1 = field 1; $2 = field 2

```
SAMPLE01C  CGTTGCTGGCCGGATTTGCGCAGCAGGCGCGCGATCTCGTGGTCGTGCGCATTGACGCCCGCCCGCGCATCGACCAGGAACACCAC
SAMPLE-04  CGCTGCTGGCCAGATTTACGGAGC----------TTTCGTGGTCGTGCGCGTTGACGCCGGCGCGCGCGTCGACCAGGAACACCAC
SAMPLE05G  CGCTGCTGGCCGGATTTGCGCAGCAGGCGGGCGATTTCGTGGTCGTGCGCGTTGATGCCGGCACGGGCATCGACCAGGAACACGAC
SAMPLE-7A  CGCTGCTGACCGGACTTACGCAG------------------------------------------------------------
```

- **need to manipulate nucleotide sequence in all rows**
- **problem: same letters in sequence names**
- **solution: sequence name lower case, sequence upper case, dashes in names as underline**
- **awk: change strain names to lower case and replace '-' by '_'**

MAKE THE SCRIPT USER FRIENDLY!!!

- write instructions to yourself
- let the computer display what it's currently doing

- awk: change strain names to lower case and replace '-' by '_'

```
#!/bin/bash
# PhyGenome_Align_remove_missing_data.sh
# remove variably present loci, keep only core loci

# enter file names as needed
FILESNP="128genomes.phy"
NAMESNP=${FILESNP%%".phy"}

echo ""
echo "load input file $NAMESNP"
echo ""
echo "awk: change strain names to lower case and '-' to '_'"
echo "-----------------------------------------------"

# make sequence name lower case
cat $FILESNP | awk -v FS="\t" -v OFS="\t" '{$1=tolower($1);
print $0}' > fake
```

write instructions to yourself

you can either define the input file once or enter it again and again throughout the script

echo " " - let the computer display to the user what it is currently doing

Let's go through this command →

- awk: change strain names to lower case and replace '-' by '_'

```
# make sequence name lower case
cat $FILESNP | awk -v FS="\t" -v OFS="\t" '{$1=tolower($1);
print $0}' > fake

# cat - concatenate
# open 1 file, open and combine (=concatenate) several files

# | pipe - string several commands together into a pipeline
#        - input from memory, output into memory

# FS="\t" - Field Separator is tab: $1   $2
# OFS="\t" - Output Field Separator is tab

# '{}' - what to do
# $1=tolower($1) - new field $1 is lower case of current $1
# print $0 - print all fields

# > save as
```

- awk: change strain names to lower case and replace '-' by '_'

```
# make sequence name lower case
cat $FILESNP | awk -v FS="\t" -v OFS="\t" '{$1=tolower($1);
print $0}' > fake


# replace (substitute) "-" to "_" in strain names
cat $FILESNP | awk -v FS="\t" -v OFS="\t" '{gsub(/-/,"_",$1);
print $0}' > fake


# Why "gsub" and not "sub"?  imagine strain name: M1989-03-14

awk '{sub(/-/"_",$1);print $0}'
# replaces only 1st instance: M1989_03-14

awk '{gsub(/-/,"_",$1);print $0}'
# replaces ALL instances in a line: M1989_03_14
```

- awk: change strain names to lower case and replace '-' by '_'

```
# make sequence name lower case
cat $FILESNP | awk -v FS="\t" -v OFS="\t" '{$1=tolower($1);
print $0}' > fake


# replace (substitute) "-" to "_" in strain names
cat $FILESNP | awk -v FS="\t" -v OFS="\t" '{gsub(/-/,"_",$1);
print $0}' > fake


Let's pipe it:
# replace "-" to "_" in strain names and lower case
cat $FILESNP | awk -v FS="\t" -v OFS="\t" '{$1=tolower($1);
print $0}' | awk -v FS="\t" -v OFS="\t" '{gsub(/-/,"_",$1);
print $0}' > fake
```

```
SAMPLE01C  CGTTGCTGGCCGGATTTGCGCAGCAGGCGCGCGATCTCGTGGTCGTGCGCATTGACGCCCGCCCGCGCATCGACCAGGAACACCAC
SAMPLE-04  CGCTGCTGGCCAGATTTACGGAGC----------TTTCGTGGTCGTGCGCGTTGACGCCGGCGCGCGCGTCGACCAGGAACACCAC


sample01c  CGTTGCTGGCCGGATTTGCGCAGCAGGCGCGCGATCTCGTGGTCGTGCGCATTGACGCCCGCCCGCGCATCGACCAGGAACACCAC
sample_04  CGCTGCTGGCCAGATTTACGGAGC----------TTTCGTGGTCGTGCGCGTTGACGCCGGCGCGCGCGTCGACCAGGAACACCAC
```

- **python: transpose rows to columns**

```
# insert tab after each nucleotide to get independent loci,
input_file "fake", output_file "fake2"
echo ""
echo "python: replace nucleotides by numbers plus tab"
echo "-----------------------------------------------"
python2.6 ../../bin/replace_nucs_to_nucsplustab_in_file.py
```

call python v2.6 # where is the script

```
sample01c  CGTTGCTGG...
sample_04  CGCTGCTGG...

sample01c  C         G         T         T         G         C         T         G         G
sample_04  C         G         C         T         G         C         T         G         G
```

Python script: `replace_nucs_to_nucsplustab_in_file.py`

```
#!/usr/bin/env python

input = open('fake', "r")

output = open('fake2', "w")


stext1 = 'A'  rtext1 = 'A\t'

stext2 = 'C'  rtext2 = 'C\t'

stext3 = 'G'  rtext3 = 'G\t'

stext4 = 'T'  rtext4 = 'T\t'

stext5 = '-'  rtext5 = 'Z\t'

stext6 = 'N'  rtext6 = 'Z\t'


output.write(input.read().replace(stext1,
rtext1).replace(stext2, rtext2).replace(stext3,
rtext3).replace(stext4, rtext4).replace(stext5,
rtext5).replace(stext6, rtext6))
```

why Z? Any letter not A C G T or N will do
(or not IUPAC depending on what you wanna do)

- awk: remove extra tab ta the end of the line

```
# remove extra tab at the end of each line
echo ""
echo "awk: remove extra tab at the end of each line"
echo "-----------------------------------------------"
cat fake2 | awk -v FS="\t" -v OFS="\t" '{sub(/[ \t]+$/, "");
print $0}' > fake3
```

- python: transpose rows to columns

```
# transform rows to columns

echo ""

echo "python: transpose rows to columns"

echo "-----------------------------------------------"

cat fake3 | python2.6 ../../bin/rows2columns_transposition.py
> fake4
```

This time we pipe python. Input from memory, output to memory.

Python script: `rows2columns_transposition.py`

```
#!/usr/bin/env python
"""
rows_to_colums_transposition.py
input(sys.stdin) : A file with strains and tab separated
loci in rows
output (sys.stdout): A file with strains and loci in
columns
"""
import sys


for c in zip(*(l.strip().split() for l in
sys.stdin.readlines() if l.strip())):
    print('\t'.join(c))
```

- awk: select core loci (no missing data)

The story so far:

- we renamed $1 to lower case and changed "–" to "_"
- we replaced missing data ("-", "N") with "Z"
- we transposed rows to columns

sample1c	sample_04	sample05g	sample_7a
A	G	A	A
A	G	T	T
A	G	Z	Z
C	C	C	T

```
# select only rows that do not contain "Z" (=core loci only)
echo ""
echo "selecting core loci"
cat fake4 | grep -v "Z"  > fake5

# grep - global regular expression print - ("grab")
# -v --invert-match (select all lines that do not contain Z)
```

- awk: select core loci (no missing data)

The story so far:

- we renamed $1 to lower case and changed “–” to “_”
- we replaced missing data (“-”, “N”) with “Z”
- we transposed rows to columns
- we selected core loci

sample1c	sample_04	sample05g	sample_7a
A	G	A	A
A	G	T	T
C	C	C	T

How many loci did we end up with?

```
# determine the number of loci in the resulting file

cat fake5 | grep -v s | wc -l > fake5a

echo "The dataset from file '$NAMESNP' consists of $(cat
fake5a) core loci. "

# grep -v s – select all lines that do not contain “s”

# wc -l – word count, count the number of lines (-l)

# cat fake5a – open file fake5a, which is just a number
```

- awk: select core loci (no missing data)

The story so far:

- we renamed $1 to lower case and changed “–” to “_”
- we replaced missing data (“-”, “N”) with “Z”
- we transposed rows to columns
- we selected core loci

sample1c	sample_04	sample05g	sample_7a
A	G	A	A
A	G	T	T
C	C	C	T

How many loci did we end up with?

```
# determine the number of loci in the resulting file
# grep -v s - requires a common character (“s”)in all names
# alternatively:
cat fake5 | awk 'NR>1' | wc -l > fake5a

# awk 'NR>1' - select all lines (=rows) after the first
```

- python: transpose rows to columns

```
# transform rows to columns

echo "python: transpose rows to columns"

echo "-----------------------------------------------"

cat fake5 | python2.6 ../../bin/rows2columns_transposition.py >
fake7
```

-awk: add extra tab at the end of each line

```
cat fake7 | awk '{print $0"\t"}' > fake 8
```

python: replace nucleotides plus tab by nucleotides

```
cat fake8 | python2.6
../../bin/replace_nucs_plus_tab_by_nucs.py > fake9
```

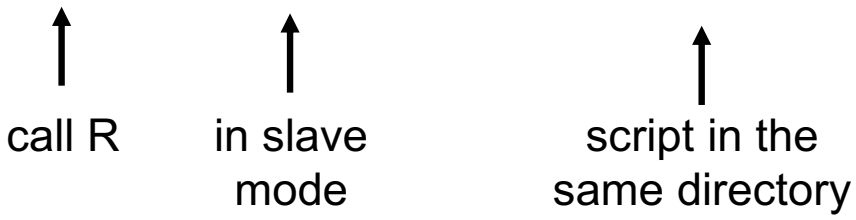
- write final output file

```
echo ""
echo "awk: writing output file"
echo "-----------------------------------------------"
cat fake9 | awk -v FS="\t" -v OFS="\t" '{print $1,$2}' >
$NAMESNP-no-gaps.phy
```

- python: replace nucs by numbers (fake5 > fake6)
  as before (stext and rtext)

- **R: Calculate Distance matrix**

```
echo "R: Calculate Distance matrix."
echo "-----------------------------------------------"
# Run R in '--slave' mode to incorporate in bash script
R --slave -f Dist_mat_Genomes.R
```



R:

- **another scripting language**
- **awesome for calculations**
- **syntax different from bash or python**

Syntax: R vs Python

R: read file

```
a <-read.table(“fake6”, header=TRUE, sep=“\t”)
```

Python: read file

```
input = open('fake6', "r")
```

R: transpose rows to columns

```
y = t(x)
```

Python: transpose rows to columns

```
for c in zip(*(l.strip().split() for l in
sys.stdin.readlines() if l.strip())):
    print('\t'.join(c))
```

R: write file

```
write.table(m5, file = “SEQ1.dist”, sep = “\t”, row.names =
FALSE, column.names = FALSE)
```

Python: write file

```
output = open('fake7', "w")
```

–R: Calculate Distance matrices of SNPs and Genes

```
#!usr/bin/R

#delete all objects

rm(list = ls())

#load packages

library(ade4)

library(MASS)

a <-read.table("fake6", header=TRUE, sep="\t") ## load data

x = t(a) ## transform data to genomes by row and SNPs by col

SEQ1.dist <- as.dist(dist(x, "manhattan")) ## calc matrix

m5 <- as.matrix(SEQ1.dist) ## write as matrix

write.table(m5, file = "SEQ1.dist", sep = "\t", row.names =
FALSE, column.names = FALSE)
```

Distance matrix

0.197									
0.219	0.021								
0.196	0.519	0.558							
0.192	0.513	0.551	0.006						
0.208	0.536	0.575	0.056	0.053					
0.218	0.554	0.594	0.062	0.059	0.036				
0.221	0.558	0.598	0.065	0.060	0.038	0.042			
0.222	0.561	0.601	0.071	0.066	0.044	0.049	0.049		
0.226	0.572	0.613	0.068	0.065	0.037	0.052	0.055	0.061	
0.272	0.642	0.677	0.275	0.271	0.286	0.297	0.298	0.302	0.307

- transfer distance matrix
- change to MEGA format
- MEGA – Molecular Evolutionary Genetics Analysis
- load matrix and display tree

https://www.megasoftware.net/

MEGA format:

#mega

Title distance matrix genome-wide SNPs in 128 Bordetella genomes;

[1] # sample_1a

[2] # sample02

[3] # sample3a

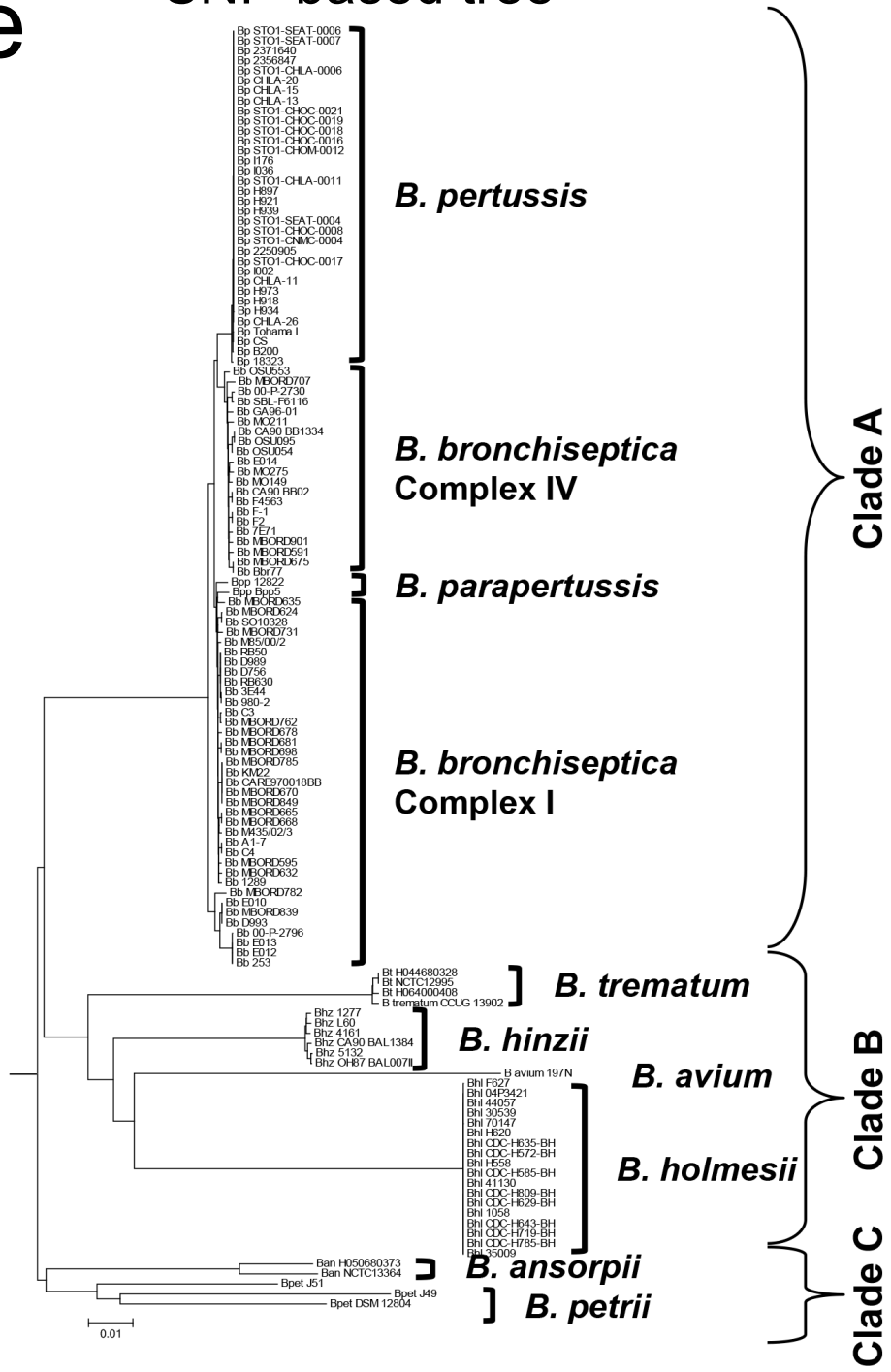
[4] # sample4c

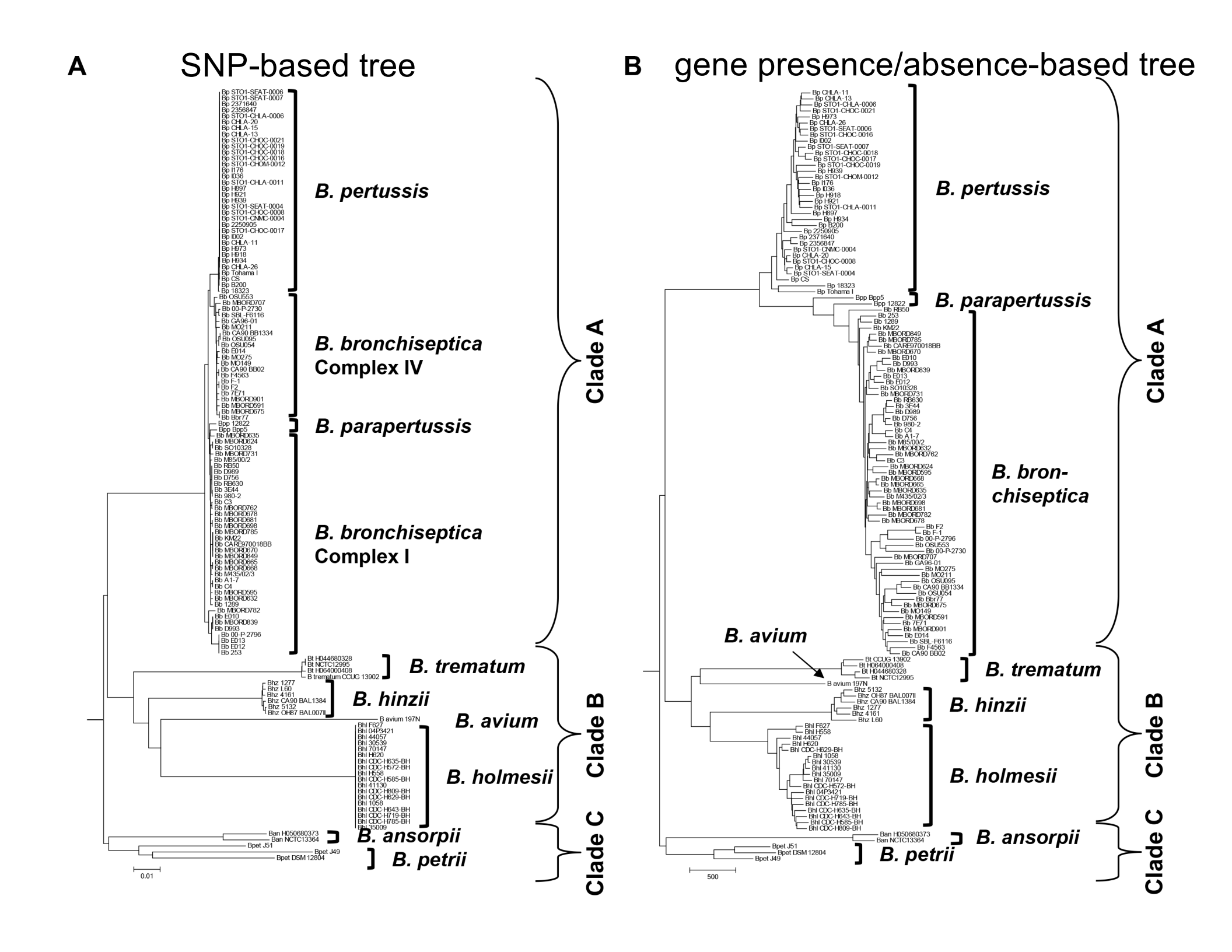
[	1	2	3	4]
[1]				
[2]	0.007695584			
[3]	0.000200096	0.007495488		
[4]	0.00021632	0.007511712	0.000016224	

Change matrix to MEGA format: either by hand in text editor or by scripting

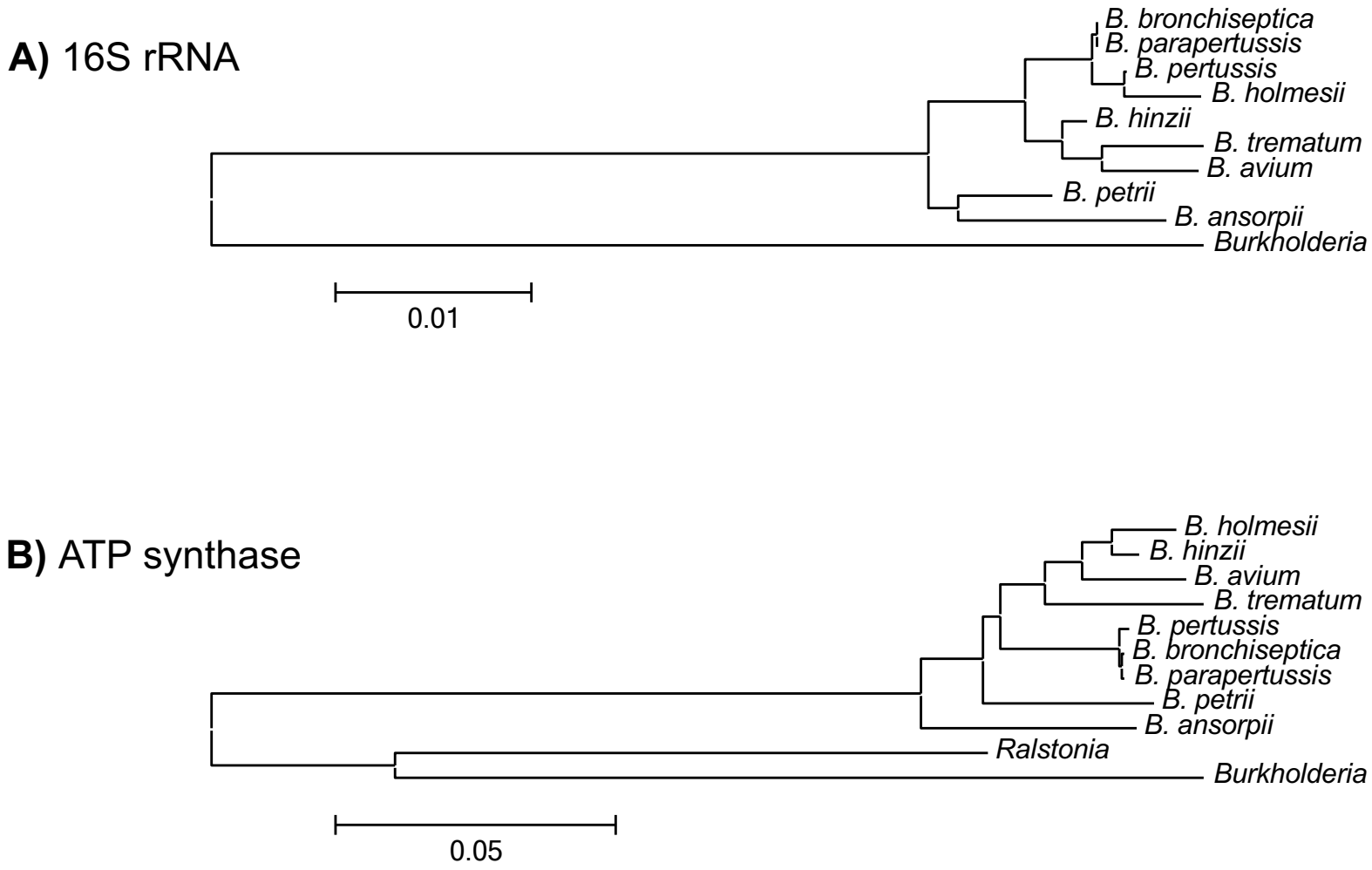
```
echo "Writing output file."
echo ""

printf "#mega\nTitle distance matrix of genome sequences from 10 Bordetella species;\n\n" > 10gen.meg
cat 10gen.phy | awk 'NR==1' | awk -v FS="\t" -v OFS="" '{print "[ 1]  #",$1}' >> 10gen.meg
cat 10gen.phy | awk 'NR==2' | awk -v FS="\t" -v OFS="" '{print "[ 2]  #",$1}' >> 10gen.meg
cat 10gen.phy | awk 'NR==3' | awk -v FS="\t" -v OFS="" '{print "[ 3]  #",$1}' >> 10gen.meg
cat 10gen.phy | awk 'NR==4' | awk -v FS="\t" -v OFS="" '{print "[ 4]  #",$1}' >> 10gen.meg
cat 10gen.phy | awk 'NR==5' | awk -v FS="\t" -v OFS="" '{print "[ 5]  #",$1}' >> 10gen.meg
cat 10gen.phy | awk 'NR==6' | awk -v FS="\t" -v OFS="" '{print "[ 6]  #",$1}' >> 10gen.meg
cat 10gen.phy | awk 'NR==7' | awk -v FS="\t" -v OFS="" '{print "[ 7]  #",$1}' >> 10gen.meg
cat 10gen.phy | awk 'NR==8' | awk -v FS="\t" -v OFS="" '{print "[ 8]  #",$1}' >> 10gen.meg
cat 10gen.phy | awk 'NR==9' | awk -v FS="\t" -v OFS="" '{print "[ 9]  #",$1}' >> 10gen.meg
cat 10gen.phy | awk 'NR==10' | awk -v FS="\t" -v OFS="" '{print "[10]  #",$1,"\n"}' >> 10gen.meg

printf "[\t1\t2\t3\t4\t5\t6\t7\t8\t9\t10  ]\n" >> 10gen.meg

printf "[ 1] \n" >> 10gen.meg
cat 10gens.dist | awk 'NR==2' | awk -v FS="\t" -v OFS="" '{print "[ 2]\t",$1,“ "}' >> 10gen.meg
cat 10gens.dist | awk 'NR==3' | awk -v FS="\t" -v OFS="" '{print "[ 3]\t",$1,"\t",$2,“ "}' >> 10gen.meg
cat 10gens.dist | awk 'NR==4' | awk -v FS="\t" -v OFS="" '{print "[ 4]\t",$1,"\t",$2,"\t",$3,“ "}' >> 10gen.meg
cat 10gens.dist | awk 'NR==5' | awk -v FS="\t" -v OFS="" '{print "[ 5]\t",$1,"\t",$2,"\t",$3,"\t",$4,“ "}' >> 10gen.meg
cat 10gens.dist | awk 'NR==6' | awk -v FS="\t" -v OFS="" '{print "[ 6]\t",$1,"\t",$2,"\t",$3,"\t",$4,"\t",$5,“ "}' >> 10gen.meg
cat 10gens.dist | awk 'NR==7' | awk -v FS="\t" -v OFS="" '{print "[ 7]\t",$1,"\t",$2,"\t",$3,"\t",$4,"\t",$5,"\t",$6,“ "}' >> 10gen.meg
cat 10gens.dist | awk 'NR==8' | awk -v FS="\t" -v OFS="" '{print "[ 8]\t",$1,"\t",$2,"\t",$3,"\t",$4,"\t",$5,"\t",$6,"\t",$7,“ "}' >> 10gen.meg
cat 10gens.dist | awk 'NR==9' | awk -v FS="\t" -v OFS="" '{print "[ 9]\t",$1,"\t",$2,"\t",$3,"\t",$4,"\t",$5,"\t",$6,"\t",$7,"\t",$8,“ "}' >> 10gen.meg
cat 10gens.dist | awk 'NR==10' | awk -v FS="\t" -v OFS="" '{print "[10]\t",$1,"\t",$2,"\t",$3,"\t",$4,"\t",$5,"\t",$6,"\t",$7,"\t",$8,"\t",$9,“\t \n"}' >> 10gen.meg

echo ""
echo "Done."
echo ""
```

Display tree




A
SNP-based tree
B
gene presence/absence-based tree
B. pertussis
B. bronchiseptica Complex IV
B. parapertussis
B. bronchiseptica Complex I
B. trematum
B. hinzii
B. avium
B. holmesii
B. ansorpii
B. petrii
Clade A
Clade B
Clade C
0.01
500

Neighbor-joining trees of 16S rRNA gene sequences and 8 concatenated ATP synthase proteins from *Bordetella*



- **R: Calculate Distance matrices of SNPs and Genes**
- **R: Calculate Mantel correlation between 2 phylogenies**

```
a <-read.table("fake5_gene1", header = TRUE, sep = "\t"
## load data gene 1
x = t(a) ## transform data to genomes by row and SNPs by col
SEQ1.dist <- as.dist(dist(x, "manhattan")) ## calc matrix
m1 <- as.table(SEQ1.dist) ## write as table
##########################################################
z <-read.table("fake5_gene2", header = TRUE, sep = "\t"
## load data gene 2
y = t(z) ## transform data to genomes by row and SNPs by col
SEQ2.dist <- as.dist(dist(y, "manhattan")) ## calc matrix
m2 <- as.table(SEQ2.dist) ## write as table
```

- **R: Calculate Distance matrices of SNPs and Genes**
- **R: Calculate Mantel correlation between 2 phylogenies**

```
m3 <-mantel.rtest(SEQ1.dist, SEQ2.dist, nrepet = 99999)
fileConn <- file("output.txt")
write.lines(paste(m3[2:4], sep = "\t"), fileConn)
close fileConn
```

```
cat output.txt
```

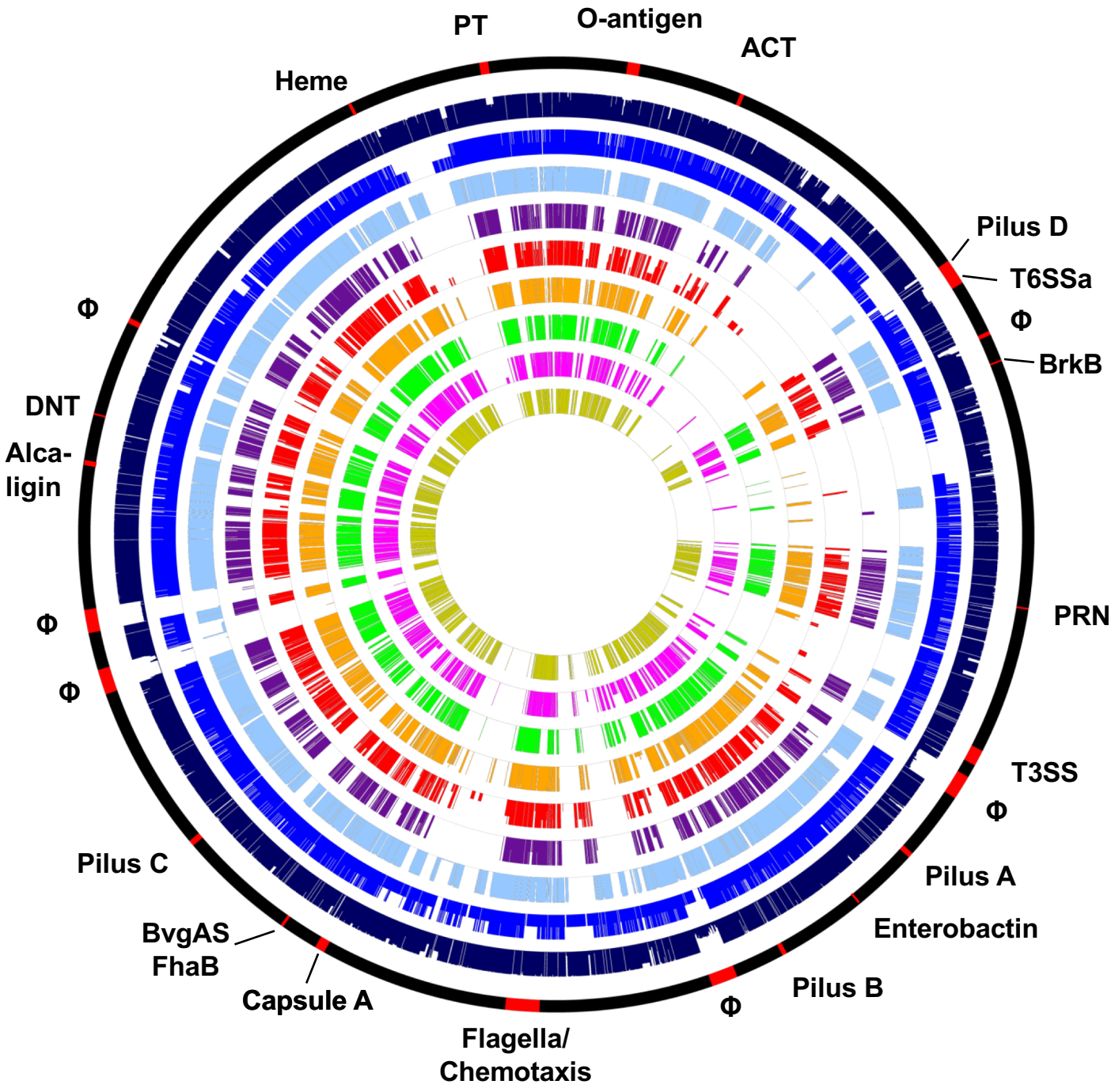
extract values from output.txt

```
cat output.txt | awk 'NR==1' > t1
cat output.txt | awk 'NR==2' > t2
cat output.txt | awk 'NR==3' > t3
printf "r = $(cat t1) \n nrepet = $(cat t2) \n p-value = $(cat t3) \n" >> $NAMEGENE1-$NAMEGENE2.out
```

extract values from output.txt

```
cat output.txt | awk 'NR==1' > t1

cat output.txt | awk 'NR==2' > t2

cat output.txt | awk 'NR==3' > t3

printf "r = $(cat t1) \n nrepet = $(cat t2) \n p-value = $(cat t3) \n" >> $NAMEGENE1-$NAMEGENE2.out


cat $NAMEGENE1-$NAMEGENE2.out

Dataset from file '9BordetellaSNP': 265372 loci.

Dataset from file 'ATPsynthase_AA': 2125 loci.

r = 0.65755                    # R^2 = 0.4324

nrepet = 99999

p-value = 0.00483
```

Presence and absence of genes in 128 genomes from 9 *Bordetella* species



Virtual chromosome of the *B. bronchiseptica* RB50 reference genome with key factor genes or gene clusters in red.

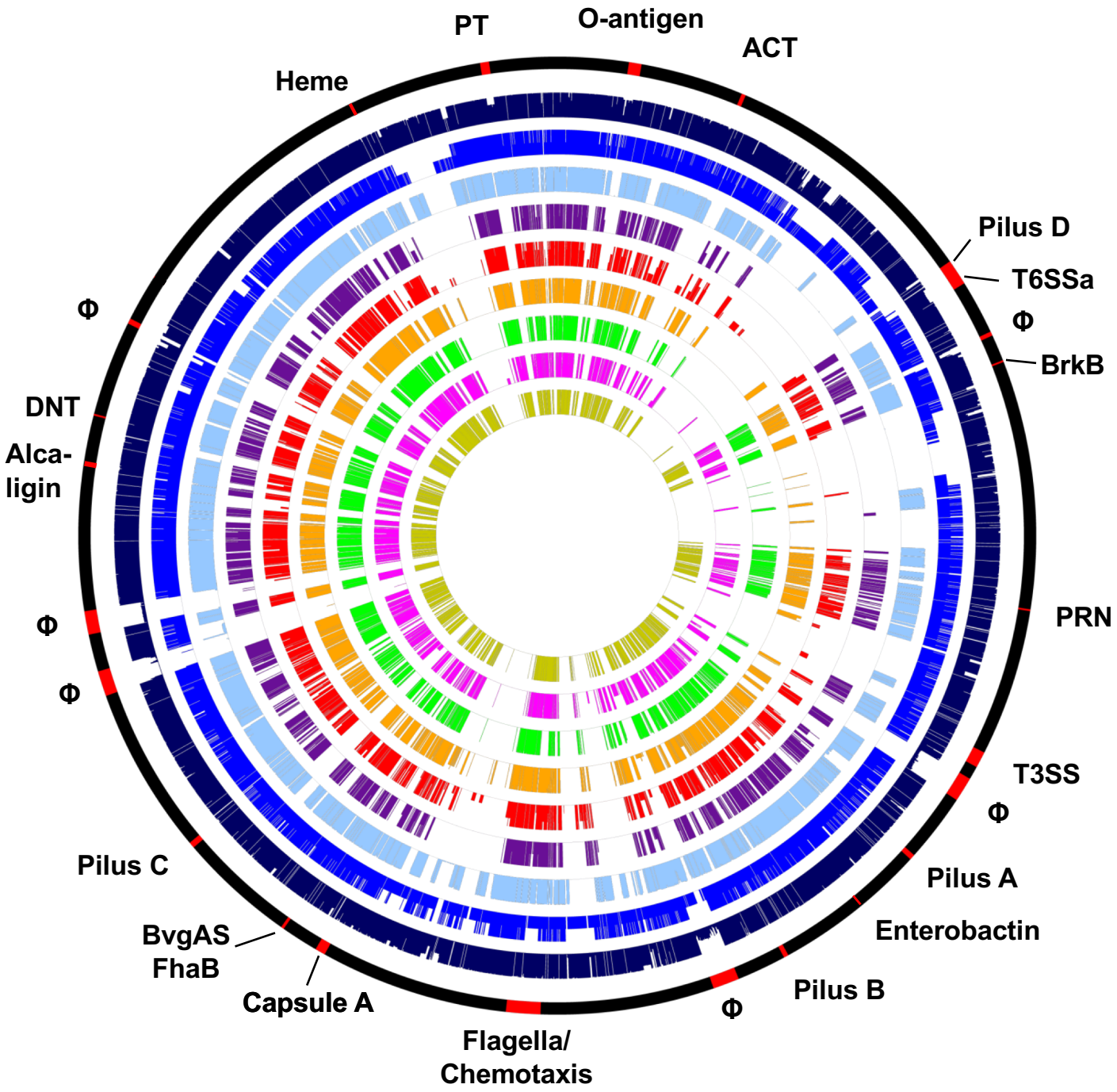
Proportion of genes present in individual genomes per species color-coded by species.

A thin line for each gene indicates the percentage of genomes in each species containing this gene.

colored: gene(s) present
white: gene(s) absent

Φ – prophage

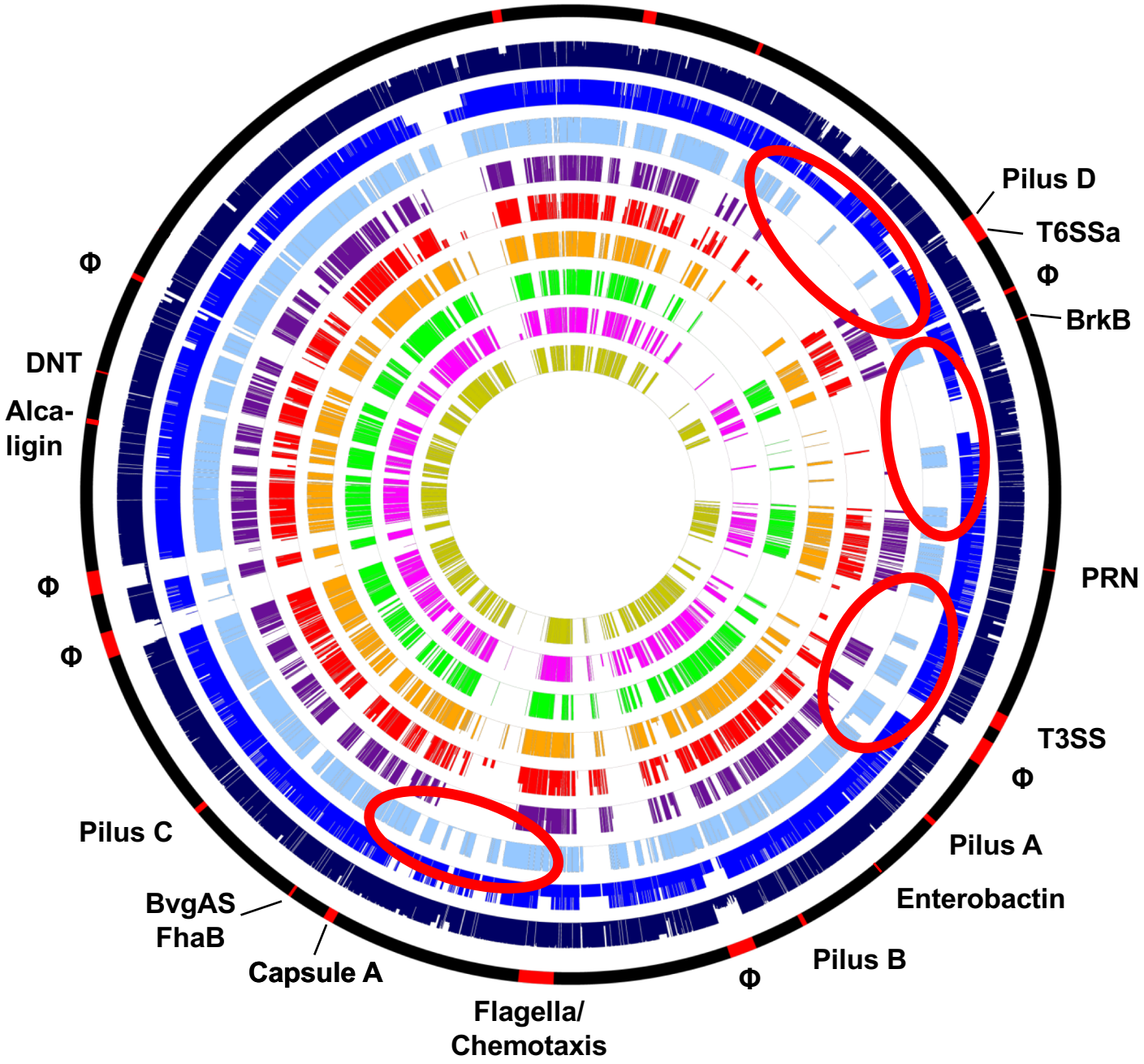
Presence and absence of genes in 128 genomes from 9 *Bordetella* species



Circles

1: Virtual chromosome of *B. bronchiseptica* RB50 with genes of interest;
2: *B. bronchiseptica* (based on 58 genomes);
3: *B. parapertussis* (2);
4: *B. pertussis* (34);
5: *B. ansorpii* (2);
6: *B. petrii* (3);
7: *B. hinzii* (6);
8: *B. holmesii* (18);
9: *B. trematum* (4);
10: *B. avium* (1)

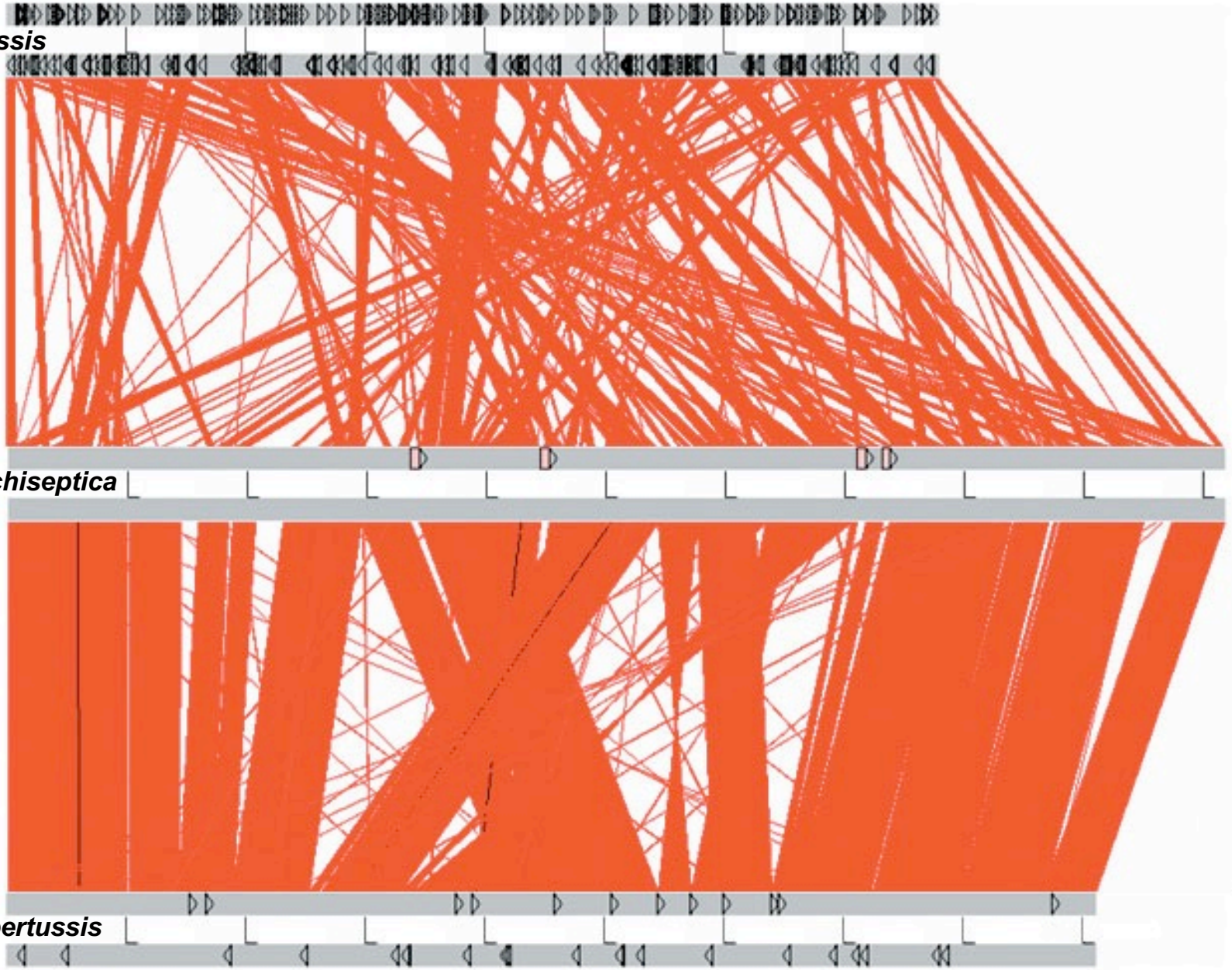
Massive gene loss during the evolution of *B. pertussis* from a *B. bronchiseptica* - like ancestor



Circles

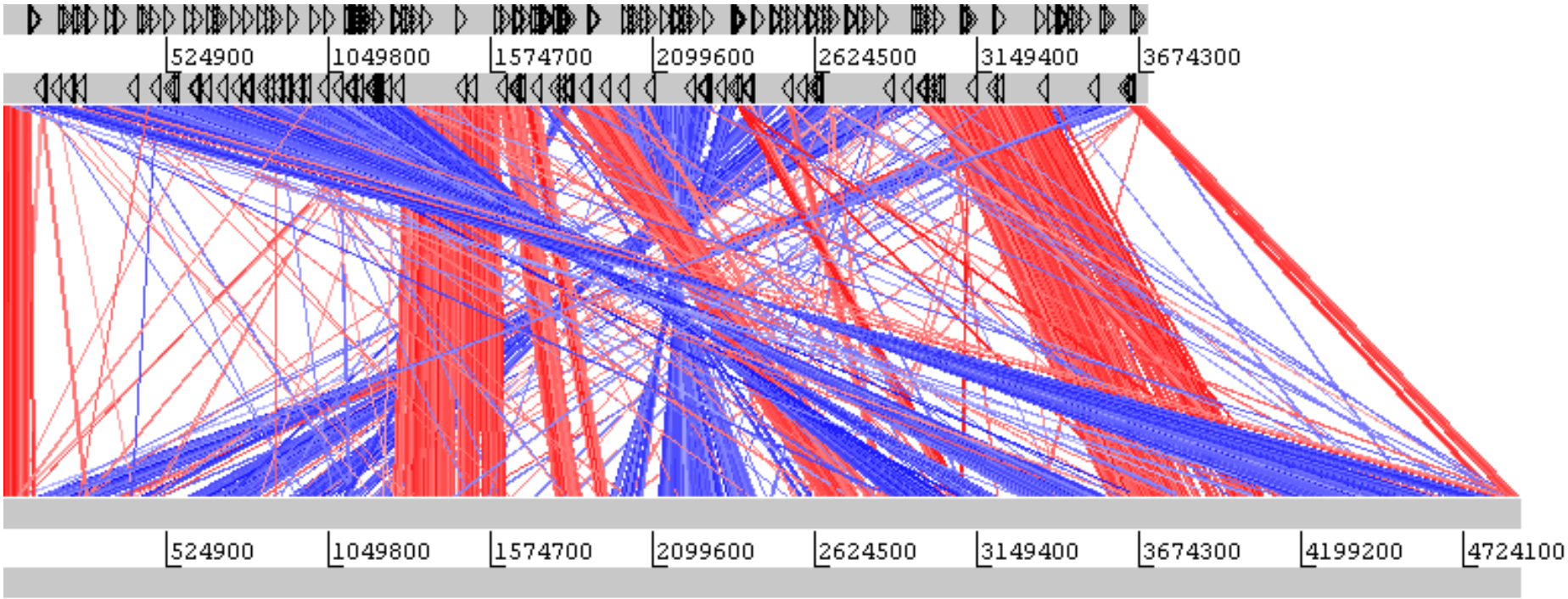
1: Virtual chromosome of *B. bronchiseptica* RB50 with genes of interest;
2: *B. bronchiseptica* (based on 58 genomes);
3: *B. parapertussis* (2);
4: *B. pertussis* (34);
5: *B. ansorpii* (2);
6: *B. petrii* (3);
7: *B. hinzii* (6);
8: *B. holmesii* (18);
9: *B. trematum* (4);
10: *B. avium* (1)

ACT – Artemis Comparison Tool



Parkhill et al 2003 Nat Genet 35: 32-40

B. holmesii



B. hinzii

80% of chromosomal breakpoints are flanked by IS-elements

Linz et al 2016 BMC Genomics 17: 767

How to perform a genome comparison and display in ACT?

1. Whole Genome Blast – genome comparison
2. MSPcrunch – change blast format to Artemis input

```
# need fasta files of both genomes
# generate data base, use "formatbd"
formatdb -i genome1.fasta -p F -o T

# -i: input Fasta file
# -p: T input type protein, F nucleotide sequence
# -o: T output database NCBI styled, F none

# output:
# genome1.nhr
# genome1.nin
# genome1.nsd
# genome1.nsi
# genome1.nsq
```

How to perform a genome comparison and display in ACT?

1. Whole Genome Blast – genome comparison
2. MSPcrunch – change blast format to Artemis input

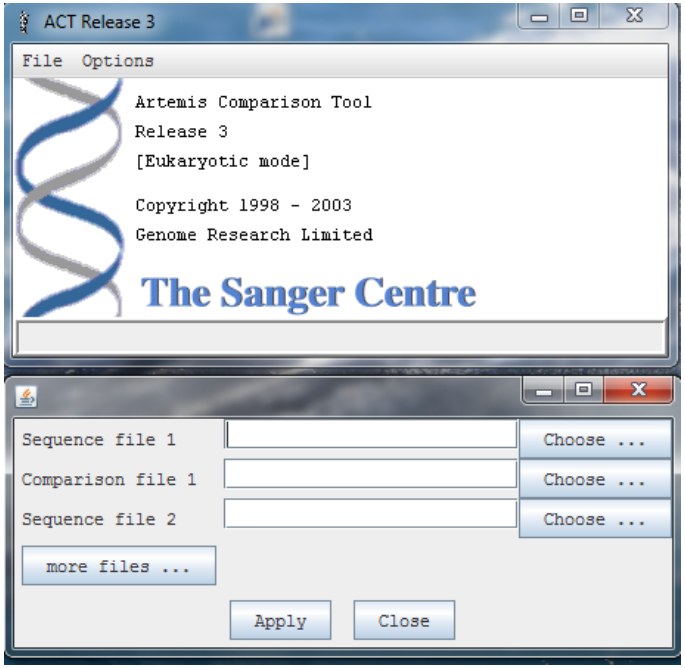
```
# need fasta files of both genomes
# run blastn
# syntax: blastall -p [program] -d [database] -i
[subject genome] -b [max hits] -v [max hits] -o
[output file]

blastall -p blastn -d genome1.fasta -i genome2.fasta
-o genome1-genome2.out -v 1000000 -b 1000000

Query: 4599606 tggtgaggtcgggcgaatcgtcca
               |||||||||||| ||||||||||
Sbjct: 4074107 tggtgaggtcggacgaatcgtcca

Query: 4599666 caggagcttgttgcattgcgatgc
               ||||  ||||||||||||||||||
Sbjct: 4074047 cagggacttgttgcattgcgatgc
```

How to perform a genome comparison and display in ACT?

1. Whole Genome Blast – genome comparison
2. MSPcrunch – change blast format to Artemis input

```
# take blast output and change format to table
MSPcrunch -d genome1-genome1.out > genome1-
genome2.cmp

what you get:
score % sim from   to     gen1  from  to   genome2
10689 99.58 181497 183650 AXSJ     1  2154 Bb_RB50
 8233 99.82 183699 185350 AXSJ  2143  3794 Bb_RB50


# so, we got:
#     genome1.fasta (or genome1.gbk)
#     genome1-genome2.cmp
#     genome2.fasta (or genome2.gbk)
```

Load your files in ACT



genome1.fasta (or genome1.gbk)
genome1-genome2.cmp
genome2.fasta (or genome2.gbk

https://www.sanger.ac.uk/science/tools/artemis-comparison-tool-act

Blastall and MSPcrunch - Download, install and run

`Blastall`

go to: `ftp://ftp.ncbi.nlm.nih.gov/toolbox/ncbi_tools/old`

select toolbox folder, e.g. `20120620`

click on `ncbi.tar.gz` to download

go to "Downloads" on your computer

to unpack type: `tar -xvzf ncbi.tar.gz`

to `make` type: `./ncbi/make/makedis.csh`

change directory: `cd ncbi/bin`

copy everything to: `/home/[user]/bin` (change to your bin directory)

`MSPcrunch`

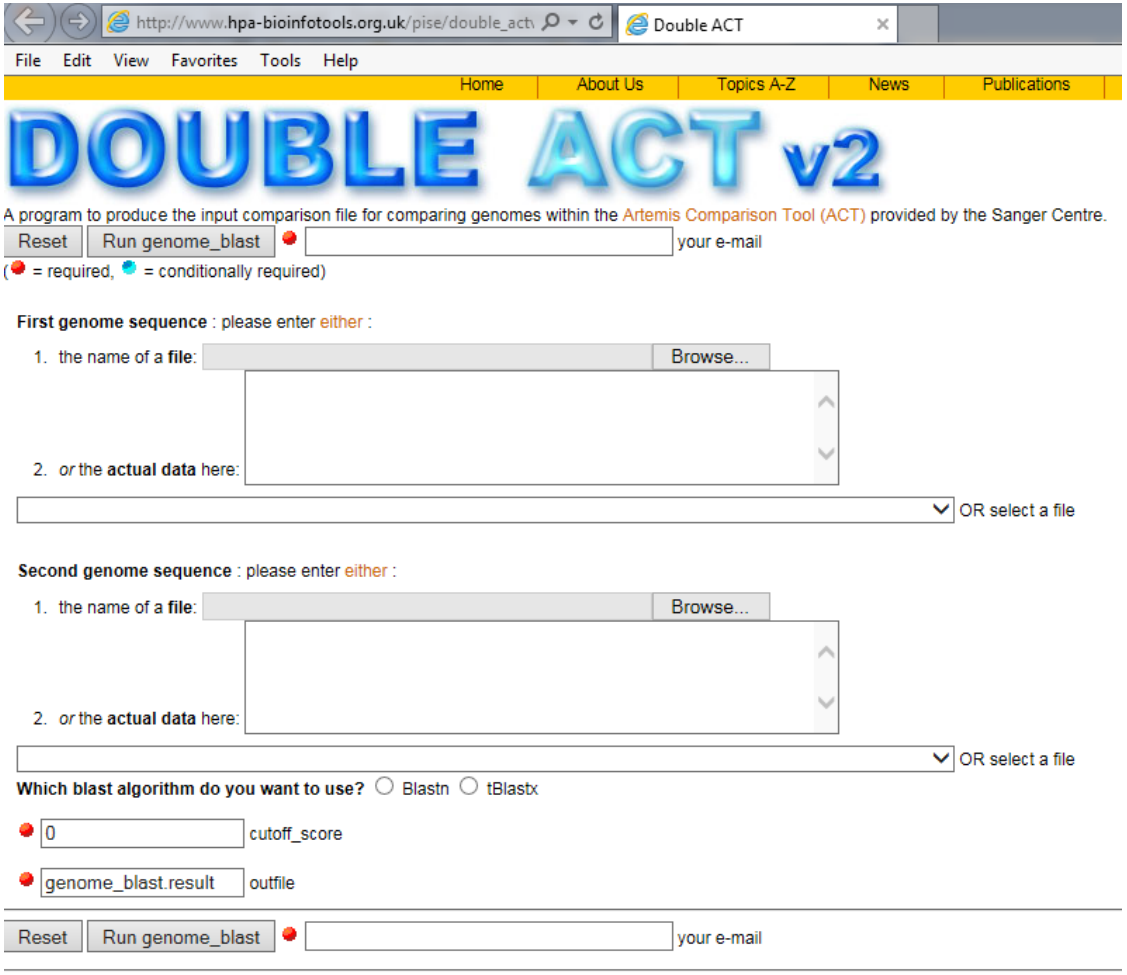
Get MSPcrunch from:

`http://sonnhammer.sbc.su.se/download/software/MSPcrunch+Blixem/`

install (or get the compiled program from me)

Alternatively: **internet double_ACT**

www.hpa-bioinfotools.org.uk/pise/double_actv2.html



Pro: choice between blastn
= sequence vs sequence

or tblastx
= translated seq
vs. translated seq

Con: - slower

- get email with a link
- open the output file
- select all, copy
- paste in text editor
- save as comparison file

Let's shift gears:

run genome comparison against multiple genomes in a loop

genome1: BhinziiL60.fasta vs genome2: BhinziiF582.fa
BhinziiH568.fa
BhinziiNCTC.fa
Bhinzii5132.fa
Bhinzii1277.fa
BhinziiCA90.fa

```
#!/bin/bash
# multiple_genomes_to_ACT.sh
# Author Bodo Linz
# run BLASTn and MSPcrunch for several genomes

DATABASE=BhinziiL60.fasta
BLASTALL=~/bin/blastall          # define location of program blastall
MSPCRUNCH=~/bin/MSPcrunch        # define location of program MSPcrunch
GENOME1=${DATABASE%%".fasta"} # database name without ".fasta"

# has the database already been formatted?

if [ -f ${DATABASE}.nhr -a ${DATABASE}.nin -a ${DATABASE}.nsd -a
${DATABASE}.nsi -a ${DATABASE}.nsq ]; then \
        echo "The database is already formatted"
else
        formatdb -i ${DATABASE} -p F -o T
        echo "Done formatting the database $GENOME1.fasta"
fi
```

Let's shift gears:

run genome comparison against multiple genomes in a loop

genome1: BhinziiL60.fasta vs genome2:

BhinziiF582.fa
BhinziiH568.fa
BhinziiNCTC.fa
Bhinzii5132.fa
Bhinzii1277.fa
BhinziiCA90.fa

```
# list the genomes to compare
files=$(ls Bhinzii*.fa)# generate list of files

# BLAST the target sequence against the reference genome
echo "Running blastn of $GENOME1 against
$files"
echo "----------------------------------------"
echo ""

for file in $files; do GENOME2=${file%%".fa"}; $BLASTALL -p
blastn -d $DATABASE -i $GENOME2.fa -o $GENOME1-$GENOME2.out;
done

# loop: for every file in list $files; do something; done


echo "Done with BLAST of $GENOME1 against
$files"
echo "----------------------------------------"
```

Let's shift gears:

run genome comparison against multiple genomes in a loop

genome1: BhinziiL60.fasta
genome2: BhinziiF582.fa, BhinziiH568.fa, BhinziiNCTC.fa, Bhinzii5132.fa, Bhinzii1277.fa, BhinziiCA90.fa

```
# Now: do the same for MSPcrunch
# list the BLAST output files
files=$(ls Bhinzii*.out)      # BhinziiL60-BhinziiF582.out etc.

# transform the blast output to ACT *.cmp table in MSPcrunch
echo "Running MSPcrunch of files
$files"
echo ""
echo "----------------------------------------"
echo ""
for file in $files; do name=${file%%".out"}; $MSPCRUNCH -d
$name.out > $name.cmp; done

echo "Done with MSPcrunch."
echo "----------------------------------------"
echo ""
echo "Done. Run ACT to visualize the genome comparison."
echo ""
```

To be continued on Sept. 25th

Thank you.