Multi-Gene Phylogenetic Inference and Comparative Tree Analysis in Rose Species

Samantha A. Taylor

15 November 2021

About the Data Set:

For this assignment, you will be using multi-gene data to perform a partitioned run of a concatenated alignment in MrBayes, and a coalescent analysis of independent gene trees in Astral. The data set you will be using comes from 11 species of wild and cultivated roses, plus one wild strawberry species to use as an outgroup.

You will have 15 gene sequences for each species, and your goal is to use all of the sequences to estimate the phylogenetic relationships among the different roses. You can download a zipped file of all of the gene sequences here.

Part One: Alignment & Concatenation in R

Create a concatenated alignment file in nexus format (to run in MrBayes), along with a partition information file in the right format to be used by MrBayes. You should refer to the Practice Exercise on Concatenated and Partitioned Runs to help you do this. Use Clustal-Omega for your alignment method. Once you have created your files, upload them to the cluster to run in MrBayes.

Creating Concatenated Alignments

```
my.files = list.files(path="./Roses", pattern="*", full.names=TRUE)

seqs = readDNAStringSet(my.files[1])
seqs = seqs[order(names(seqs))]

x = msa(seqs, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
my.lengths = rep(0, length(my.files))
my.lengths[1] = ncol(x)

for (i in 2:length(my.files)) {
    new = readDNAStringSet(my.files[i])
    new = new[order(names(new))]
    y = msa(new, method="ClustalOmega", order="input")
    my.lengths[i] = ncol(y)
    temp = paste0(x,y)
    names(temp) = names(new)
    x = DNAStringSet(temp)
}
```

```
## using Gonnet
## using Gonnet
## using Gonnet
## using Gonnet
## using Gonnet
## using Gonnet
## using Gonnet
## using Gonnet
## using Gonnet
## using Gonnet
## using Gonnet
## using Gonnet
## using Gonnet
## using Gonnet
```

```
concat.aln = DNAMultipleAlignment(x)

write.phylip(concat.aln, "RosesConcat.phy")
write.nexus.data(as.DNAbin(concat.aln), file="RosesConcat.nex")
```

Creating Partition Files

```
my.starts = c(1)
for (i in 1:(length(my.lengths)-1)) {
     my.starts[i+1] = my.lengths[i] + my.starts[i]
}

my.ends = my.starts + (my.lengths-1)

my.loci = gsub("\\.fasta", "", basename(file.path("./Roses/", my.files)))

info1 = paste(my.starts, my.ends, sep="-")
info2 = paste(my.loci, info1, sep="=")
info3 = paste("DNA,", info2, sep=" ")

write.table(info3, file="Roses-Raxml-part.txt", quote=FALSE, row.names=FALSE, col.names=FALSE)

info3 = gsub("DNA,", "charset", info3)
```

```
info3 = paste(info3, ";", sep="")
loc.list = paste(my.loci, collapse=",")
list.w.length = paste(c(length(my.loci), loc.list), collapse=":")
info4 = paste(c("partition", "cpGenes", "=", list.w.length), collapse=" ")
info4 = paste(info4, ";", sep="")
write.table(c(info3,info4), file="Roses-MrB-part.txt", quote=FALSE, row.names=FALSE, col.names=FALSE)
```

Part Two: Partitioned Analysis in MrBayes

Using your concatenated alignment file and your partition information from Part One, perform a partitioned analysis in MrBayes. Again, you will want to refer to the practice exercise to help you do this. Please adjust your parameter settings such that:

1. You are using the GTR +G +I substitution model for ALL of your loci.
2. You unlink the parameter estimation for all loci for the mutation rates, base frequencies, gamma shape parameters, and proportion of invariant sites.
3. You allow the rate prior to be variable for all loci (prset applyto=(all) ratepr=variable)
4. You define the wild strawberry (fragaria) is defined as the outgroup
5. You perform 1,000,000 iterations of the MCMC
6. You save your parameter estimates every 500 iterations

Question 1:

Once you have set up your model, please paste in the final block of code/settings from your mb_input file. (5 pts)

```
touch mb_input
nano mb_input

sed -i '/execute .*/ r Roses-MrB-part.txt' mb_input

begin mrbayes;
        set autoclose=yes nowarn=yes;
        execute RosesConcat.nex;
        charset loc0114=1-314;
        charset loc0180=315-663;
        charset loc0271=664-1135;
        charset loc0273=1136-1517;
        charset loc0535=1518-1817;
        charset loc0692=1818-2122;
        charset loc0716=2123-2526;
        charset loc0751=2527-2858;
        charset loc0778=2859-3236;
        charset loc0895=3237-3590;
        charset loc1018=3591-3906;
        charset loc1571=3907-4316;
        charset loc1723=4317-4657;
        charset loc1762=4658-5107;
```

```
        charset loc1851=5108-5501;
        partition cpGenes = 15:loc0114,loc0180,loc0271,loc0273,loc0535,loc0692,loc0716,loc0751,loc0778,l
        set partition=cpGenes;
        lset nst=6 rates=invgamma;
        lset app=(1,2,3,4,5) nst=1 rates=equal;
        unlink revmat=(all) pinvar=(all) statefreq=(all) shape=(all);
        prset applyto=(all) ratepr=variable;
        Outgroup fragaria;
        mcmc ngen=1000000 samplefreq=500;
        sump relburnin=yes burninfrac=0.25;
    sumt relburnin=yes burninfrac=0.25 conformat=simple;
end;
```

```
touch run_mb.slurm
nano run_mb.slurm

#!/bin/bash
#SBATCH --time=0:30:00
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=4
#SBATCH --mem=32gb
#SBATCH --partition=Centaurus

module load mrbayes

mpirun -np 4 mb mb_input >mb.log
```

After you have completed your run of MrBayes, download the .pstat file and the .con.tre file to examine them in R.

```
bayesTree = read.nexus("RosesConcat.nex.con.tre")
bayesTree1 = bayesTree[[1]]

pstat = read.table("RosesConcat.nex.pstat", header=TRUE, skip=1)
```

Question 2:

Do the values in your .pstat file suggest that the MCMC achieved convergence? Why or why not? (2 pts)

```
min(pstat$avgESS)
```

```
## [1] 157.2374
```

```
max(pstat$PSRF)
```

```
## [1] 1.014039
```

```
mcmc = read.table("RosesConcat.nex.mcmc", comment.char="[", header=TRUE, stringsAsFactors=FALSE)
mcmc$AvgStdDev.s.[nrow(mcmc)]
```

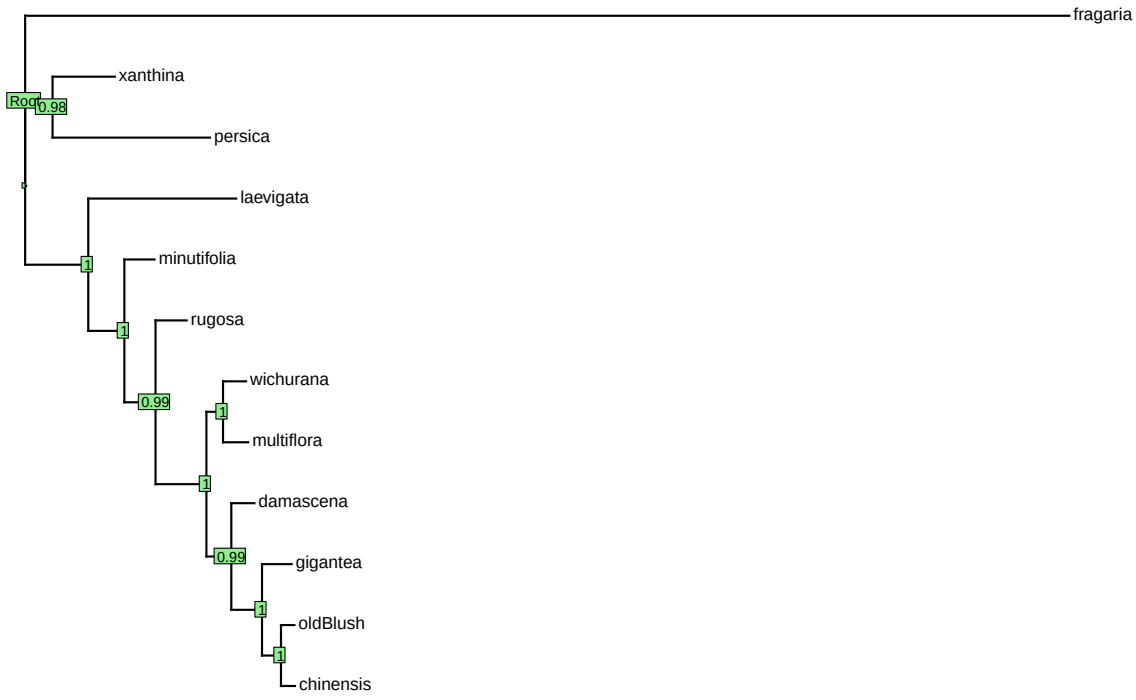
```
## [1] 0.003612
```

Answer: There are multiple indicators from my run which indicate that the MCMC converged and showed good mixing behavior. The first things I can check are the ESS and PSRF values in the .pstat table, with my lowest ESS value being 157.2374 and my maximum PSRF value being 1.014039. Because lowest ESS value is greater than 100, I have a sufficient number of uncorrelated parameter values in my chain. Because my maximum PSRF value is less than 1.1, my between-chain variance is nearly equal to the within-chain variance, so it is unlikely I could reduce the variance by much more if I ran the chain for longer. The standard deviation between the 2 runs (at the time I stopped the program AKA the last value in my mcmc file), is 0.003612. Because it is less than 0.01, my two independent runs appear to have converged on the same parameter space.

Question 3:

Provide a plot of your consensus tree from MrBayes with posterior probability values at each node. (3 pts)

```
bayesTree1$node.label = round(((as.numeric(bayesTree1$node.label))), digits=2)
bayesTree2 = root(bayesTree1, "fragaria", resolve.root=T)
plotTree(bayesTree2, edge.width=2, font=1)
nodelabels(bayesTree2$node.label, cex=0.8, bg = "lightgreen")
```



Part Three: Single Gene Alignments & Trees

In order to perform a coalescent analysis in ASTRAL, we need to create an individual gene tree for each of our 15 genes. ASTRAL will then search for the species tree which optimizes the majority of the gene trees.

To create the individual gene trees, you can go back to using RAxML just like you've been doing on past assignments, and run it individually on each of our 15 loci. To get the data ready for RAxML, you need to create a separate alignment file in phylip format for each of the 15 locus files. Again, you should use Clustal-Omega as your alignment method. Then, upload your phylip files to the cluster and run RAxML on each one. For each gene, use 1,000 bootstrap iterations.

When you do these runs, make sure you are updating the output name each time so that you do not overwrite any of your files - you want to save both the best tree and the bootstrap trees for every locus!

```
my.files = list.files(path="./Roses", pattern="*", full.names=TRUE)

seq_1 = readDNAStringSet(my.files[1])
seq_2 = readDNAStringSet(my.files[2])
seq_3 = readDNAStringSet(my.files[3])
seq_4 = readDNAStringSet(my.files[4])
seq_5 = readDNAStringSet(my.files[5])
seq_6 = readDNAStringSet(my.files[6])
seq_7 = readDNAStringSet(my.files[7])
seq_8 = readDNAStringSet(my.files[8])
seq_9 = readDNAStringSet(my.files[9])
seq_10 = readDNAStringSet(my.files[10])
seq_11 = readDNAStringSet(my.files[11])
seq_12 = readDNAStringSet(my.files[12])
seq_13 = readDNAStringSet(my.files[13])
seq_14 = readDNAStringSet(my.files[14])
seq_15 = readDNAStringSet(my.files[15])

msa_seq_1 = msa(seq_1, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_2 = msa(seq_2, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_3 = msa(seq_3, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_4 = msa(seq_4, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_5 = msa(seq_5, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_6 = msa(seq_6, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_7 = msa(seq_7, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_8 = msa(seq_8, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_9 = msa(seq_9, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_10 = msa(seq_10, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_11 = msa(seq_11, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_12 = msa(seq_12, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_13 = msa(seq_13, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_14 = msa(seq_14, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
msa_seq_15 = msa(seq_15, method="ClustalOmega", order="input")
```

```
## using Gonnet
```

```
write.phylip(msa_seq_1, "seq_1_RosesConcat_mas.phy")
write.phylip(msa_seq_2, "seq_2_RosesConcat_mas.phy")
write.phylip(msa_seq_3, "seq_3_RosesConcat_mas.phy")
write.phylip(msa_seq_4, "seq_4_RosesConcat_mas.phy")
write.phylip(msa_seq_5, "seq_5_RosesConcat_mas.phy")
write.phylip(msa_seq_6, "seq_6_RosesConcat_mas.phy")
write.phylip(msa_seq_7, "seq_7_RosesConcat_mas.phy")
write.phylip(msa_seq_8, "seq_8_RosesConcat_mas.phy")
write.phylip(msa_seq_9, "seq_9_RosesConcat_mas.phy")
write.phylip(msa_seq_10, "seq_10_RosesConcat_mas.phy")
write.phylip(msa_seq_11, "seq_11_RosesConcat_mas.phy")
write.phylip(msa_seq_12, "seq_12_RosesConcat_mas.phy")
write.phylip(msa_seq_13, "seq_13_RosesConcat_mas.phy")
write.phylip(msa_seq_14, "seq_14_RosesConcat_mas.phy")
write.phylip(msa_seq_15, "seq_15_RosesConcat_mas.phy")
```

```
#!/bin/bash
#SBATCH --time=0:30:00
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --mem=8GB
#SBATCH --partition=Centaurus

module load raxml

for infile in ./*_RosesConcat_mas.phy
do
        base=$(basename ${infile} _RosesConcat_mas.phy)
        raxmlHPC -s ${infile} \
                -n ${base}Boot \
                -f a \
                -m GTRGAMMAI \
                -N 1000 \
                -o fragaria \
                -p 1234 -x 1234 \
                >${base}_raxml_log.txt
done
```

Part Four: Run ASTRAL on the RAxML results

Once RAxML has finished running on all of the files, you need to concatenate the resulting bestTree files for each gene into one long Newick format file.

Next, you need to create a text file that lists the /filepath/ to each bootstrap file (the path can be absolute, or relative to the directory you plan to run ASTRAL from). Use a simple bash loop that I used to write the relative paths of each file to a new text file called "RAxML_Roses_BSfiles".

Once you've done this, running ASTRAL is relatively quick and easy, as I have already installed it in the class folders on the cluster.

```
cat RAxML_bestTree* >Roses_allLoci_bestTree.tre

for file in ./RAxML_bootstrap*; do echo $file >>"RAxML_Roses_BSfiles"; done

java -jar /projects/class/binf6205_001/ASTRAL/astral.5.7.8.jar -i Roses_allLoci_bestTree.tre  -b RAxML_
```

The -i flag specifies my input concatenated bestTree file, the -b flag specifies my file that lists the location of my bootstrap files, the -r flag specifies how many replicates to run (in this case just 1), and the -o flag gives the name of the output tree file to create. Note that by default Astral prints a lot of messages to the screen while it runs, so I'm redirecting those to a log file.

If you look at the Astral log, you may notice it throws a bunch of warnings about the small number of loci; the program was really designed to use thousands (or tens of thousands!) of individual gene trees, so our example with only 15 is not a very realistic usage case for it.

Part Five: Download and Plot the ASTRAL Results

Once ASTRAL has finished running, download the resulting tree file (this will be whatever filename you specified after the -o flag in the Astral command line). This output file is in Newick format, so you can read it into R using the read.tree() function.

Once you have read the file into R, you can mostly treat this the same way as you would the output from MrBayes. So, you need to take just the first tree from the file (because there will be more than one), and you need to adjust your node labels (i.e., your posterior probabilities) to look like percents. There is also one extra step you need to take, because ASTRAL does not report branch lengths for terminal branches, and this causes R a lot of problems. To take care of this, we will manually assign these branches a length of "1":

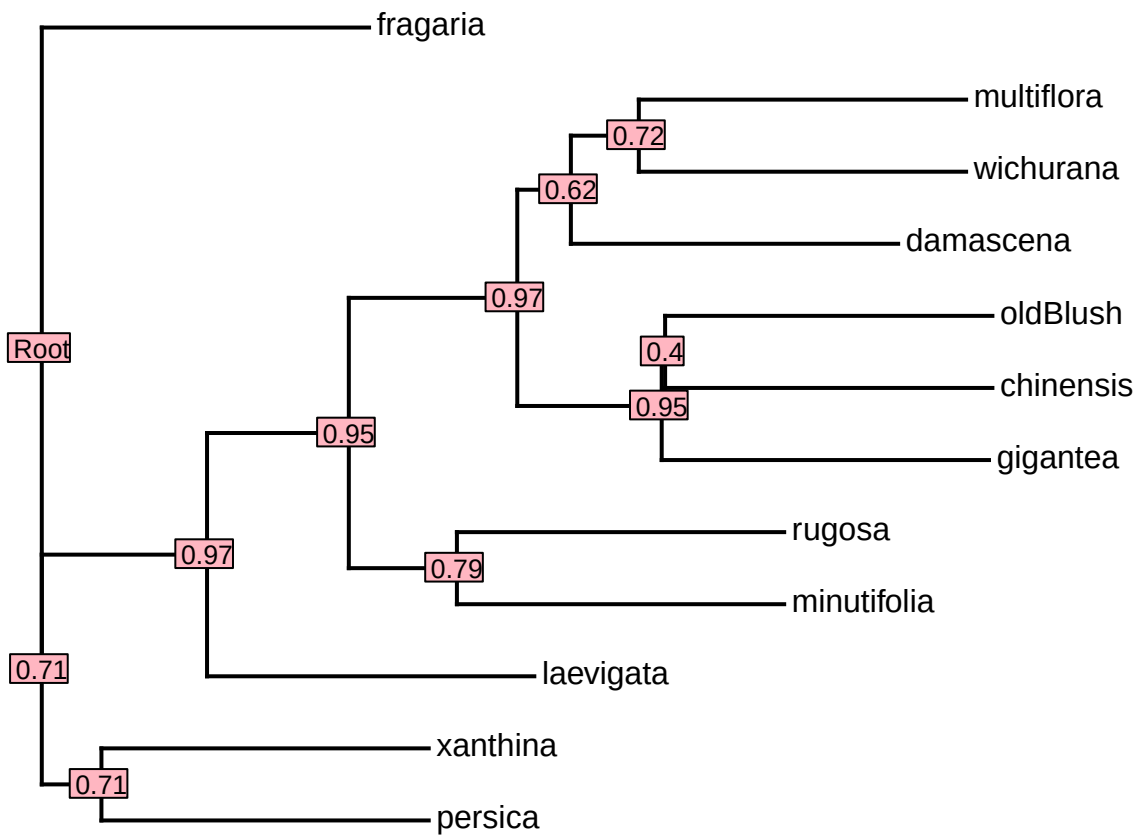
```
astralTree = read.tree("roses_astral.tre")
astralTree1 = astralTree[[1]]
astralTree1$edge.length[which(is.na(astralTree1$edge.length))] = 1
```

Now, make sure to root (or re-root) the tree, and then plot it with posterior probabilities just like you would a tree from MrBayes.

Question 4:

Provide a plot of your ASTRAL tree with posterior probability values at each node. (3 pts)

```
astralTree1$node.label = round(((as.numeric(astralTree1$node.label))), digits=2)
astralTree2 = root(astralTree1, "fragaria", resolve.root=T)
plotTree(astralTree2, edge.width=2, font=1)
nodelabels(astralTree2$node.label, cex=0.8, bg = "lightpink")
```



Answer:

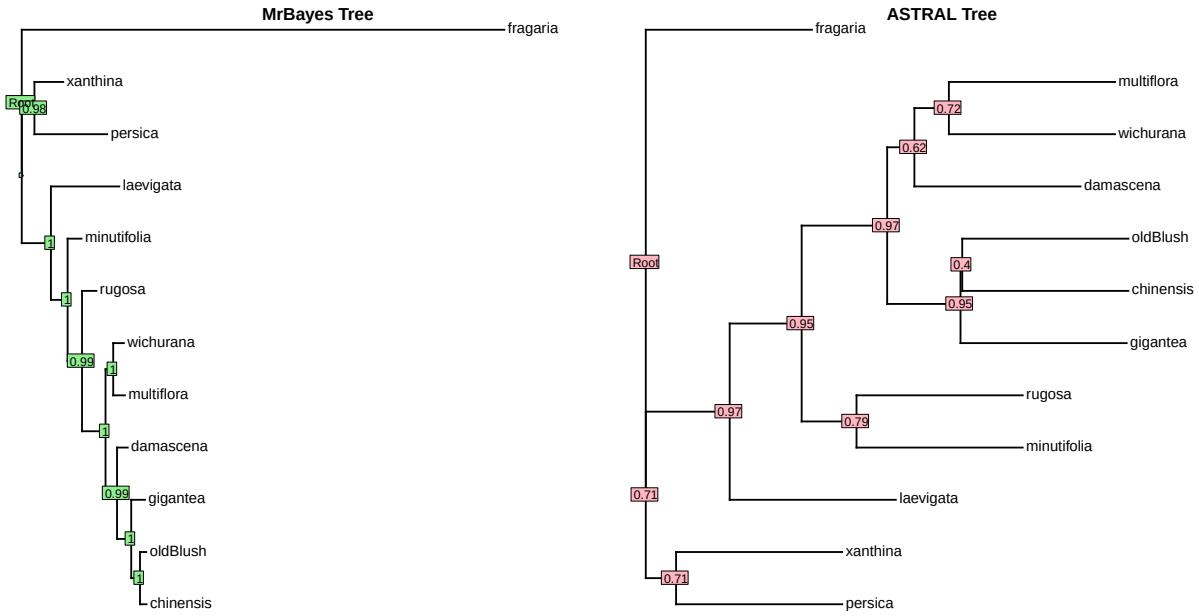
Question 5:

Perform a comparison of your ASTRAL tree and your MrBayes tree. Are they identical, or are there some differences in terms of the tree topologies? (1 pt)

```
par(mfrow=c(1,2))

bayesTree1$node.label = round(((as.numeric(bayesTree1$node.label))), digits=2)
bayesTree2 = root(bayesTree1, "fragaria", resolve.root=T)
plotTree(bayesTree2, edge.width=2, font=1)
nodelabels(bayesTree2$node.label, cex=0.8, bg = "lightgreen")
par(mar=c(1,1,1,0))
title("MrBayes Tree")

astralTree1$node.label = round(((as.numeric(astralTree1$node.label))), digits=2)
astralTree2 = root(astralTree1, "fragaria", resolve.root=T)
plotTree(astralTree2, edge.width=2, font=1)
nodelabels(astralTree2$node.label, cex=0.8, bg = "lightpink")
par(mar=c(1,1,1,0))
title("ASTRAL Tree")
```



Answer: Comparing the ASTRAL and MrBayes trees, the tree topologies have a few differences: 1) In the MrBayes tree, minutifolia is not in the same direct clade as rugosa, whereas in the ASTRAL tree the two species have their own clade. 2) In the ASTRAL tree, damascena in closest in relation to the ancestor of multiflora/wichurana, whereas in the MrBayes tree, damascena shares a closer relation with the ancestor of gigantea and oldBlush/chinensis than it does with the ancestor of multiflora/wichurana.

Question 6:

The Old Blush rose was one of the earliest varieties of cultivated rose (it has been around for roughly 1,000 years!). It is also probably the first cultivated rose ever introduced to Europe. Based on your phylogeny, where does this rose come from? (Hint: what is its closest relative?). (1 pt)

Answer: The Old Blush's closest relative is the chinensis, the China Rose, suggesting that the Old Blush comes from southwest China.
