Bayesian R for Project

Ray Stokes

2025-09-23

Equations

As a reminder, the main Bayesian equation for this project WAS:

$$P(S|ID, AQ, RD, AN) = \frac{P(AQ|S)P(AN|S)P(ID|S)P(RD|S)P(S)}{P(AQ)P(AN)P(ID)P(RD)}$$

Where the variables are:
S is the species
ID is the percentage ID
RD is the read count
AN is the assay name
AQ is the aquamaps probability (probability at that location, doesn't have to be aquamaps. We might be able to rename variable to L if it is clearer)

I have altered this slightly by adding some new terms and removing the denominator, so we now have:

$$f(S|ID, RD, AN, GP, DR, AQ, NT, DI) \propto f(ID|DR, S) \times f(RD|S) \times f(AN|S) \times f(GP|S) \times f(AQ|S)...$$

$$... \times f(NT|S) \times f(DI|S) \times f(S)$$

Where new term GP is related to the percentage of genus with eDNA in the database, DR is the normalised diversity rate for that species, NT is the number of species found in the target zone (in reference databases) and DI is the closest distance of same species in database to the observation.

Meaning of the terms:

- $f(ID|DR, S)$ is the ID match given the diversity of the species. If a species is more diverse, lower percentage matches are OK. Hence, this alters the match to: $p_{new} = p^{1-d}$ for diversity rate d. E.g for a match of 98% but high diversity rate, we can treat this as a 100% match: $0.98^{1-1} = 1$

- $f(RD|S)$ probability of match given the number of read counts. Say we have a probability of 0.3 that we pick up a correct DNA match. If we have 10 DNA matches, then the probability that at least 1 is correct is $1 - 0.7^{10} = 0.97$ probability that the species is actually present in the area, so the more counts, the more likely. If we assume there could be some bias (that each count is not independent), then we can be harsher by reducing the count by some factor, meaning $p_{new} = 1 - p^{b \times counts}$, where p is the prior p, distributed uniformly on [0,1], and so is b.
- $f(AN|S)$ is the value we assign for the assay name, closer to 1 if more trustworthy
- $f(GP|S)$ is the percentage of species in the genus whose DNA is in the database

- $f(AQ|S)$ is the aqua maps probability of finding the species at that location

- $f(NT|S)$ is the number of species found in the database, using $p_{new} = 1 - p^{counts}$ for uniform prior p, similar to treatment of reads

- $f(DI|S)$ is the probability of species in area, having the closest observation of species in database to the location. This probability will be higher for smaller distances r, and is given by: $P = e^{-kr}$. We can choose k so to determine how close you need to be for a good observation. For example, we want to find the distance where the probability is 0.5:
$\frac{1}{2} = e^{-kr} \implies \log(\frac{1}{2}) = -kr \implies \log(2) = kr \implies k = \frac{\log(2)}{r}$. For example, if you want a distance of 100km away to be a 0.5 probability, choose $k = \frac{\log(2)}{100} = 0.006931472$. Don't worry, this will be incorporated into the program so you only need to specify the distance for 0.5 probability.

For most of the proportions, we will use a Beta distribution as the model.
The expected value for a beta distribution with parameters α and β is $\frac{\alpha}{\alpha+\beta}$. If we want our beta distribution to have a shape so that the expected value is close to our proportion p, then we have to set $\frac{\alpha}{\alpha+\beta} = p$. This means $\beta = \frac{1-p}{p}\alpha$. The higher the value of α, the more informative.

The posterior can also be broken into match based and location based probabilities, which makes it easier to determine how location affects the probability:

$$f(S|ID, RD, AN, GP, DR, AQ, NT, DI) \propto [f(ID|DR, S) \times f(RD|S) \times f(AN|S) \times f(GP|S) \times f(S)]...$$

$$... \times [f(AQ|S) \times f(NT|S) \times f(DI|S)]$$

This is the same as:

$$f(S|M, L) \propto f(M|S) \times f(L|S)$$

For match M and location L.

Main function for Open BUGS:

```
eDNA_OpenBUGS <- function(percent_ID, read_counts, p_assay, diverse_score, percent_genus_db, p_species,
                          confidence_int=0.95, num_iterations=11000, num_burnin = 1000, show_trace=F, sh
{

  # Given one DNA sample and the relevant data from the same row, including related data from external
  #  make the distributions in OpenBUGS

  # ARGS:
    # percent_ID - the percentage match BLAST hit for species, as a decimal (range 0 to 1)
    # read_counts - the number of counts for the
    # p_assay - probability score assigned to assay type
    # diverse_score - normalized diversification score between 0 and 1. 0 means species is not diverse,
    # percent_genus_db - percentage of species in same genus that have eDNA recorded in databases (as a
    # p_species - proportion for the species (incorporates how rare or common it is)
    # p_aquamaps - aquamaps probability for species at that location
    # num_in_area - number of species in the general area
    # nearest_distance - closest distance between observation and another observation of the species in
    # distance_for_50_percent - distance where we assume there is a 0.5 probability of the species bein
    # confidence_int - decimal for desired confidence (credible) interval. Default is 0.95 for a 95% co
    # num_iterations - number of iterations to use for OpenBUGS simulation. The default is 11000
    # num_burnin - number of iterations to discard at the start, to let simulation 'warm up'
```

```
  # show_trace - whether you want the trace plots to be shown (TRUE/FALSE)
  # show_all_densities - whether you want the density plots for each variable to be shown (TRUE/FALSE
  # show_posterior_density -  show posterior density plots for location, match and combination of both
  # show_stats - shows the Bayesian estimates if TRUE
  # debug - TRUE/FALSE, debug in OpenBUGS if wanted
# RETURNS:
  # A data frame with expected values and confidence intervals for match, location and combined poste


# load libraries. Might be better to load once, from another function instead of loading each time fu
library(R2OpenBUGS)
library(coda)

# Write the OpenBUGS code:
writeLines("
  model{

    # percentage identity and species diversity
    ID ~ dbeta(a_id, b_id)
    p_id_dr <- pow( p_i, e)
    e <- 1 - dr
    a_id <- 2
    b_id <- abs(1 - p_id_dr) / p_id_dr * a_id

    # read counts
    RD <- 1 - pow( (1 - p_rd), (a_rd * counts) )
    p_rd ~ dunif(0, 1)
    a_rd ~ dunif(0, 1)

    # assay prob
    AN ~ dbeta(a_an, b_an)
    a_an <- 2
    b_an <- abs(1 - p_an) / p_an * a_an

    # Probabity using percentage of recorded genus
    GP ~ dbeta(a_gp, b_gp)
    a_gp <- 2
    b_gp <- abs(1 - p_gp) / p_gp * a_gp

    # Probability using proportion for species
    S ~ dbeta(a_s, b_s)
    a_s <- 2
    b_s <- abs(1 - p_s) / p_s * a_s

    # Probability from aquamaps
    AQ ~ dbeta(a_aq, b_aq)
    a_aq <- 2
    b_aq <- abs(1 - p_aq) / p_aq * a_aq

    # Number of species found in target area
    NT <- 1 - pow( (1 - p_ta), num_ta)
    p_ta ~ dunif(0, 1)
```

```
    # Distance to closest observation of species in database
    DI ~ dbeta(a_di, b_di)
    a_di <- 2
    b_di <- abs(1 - p_di) / p_di * a_di


  }", con="OpenBUGS_eDNA.txt")

# Need to pass all the variables so BUGS can use them.
# Also need to convert distance into a probability so it is compatible


prob_dist = exp(-nearest_distance*log(2) / distance_for_50_percent)

# we cannot have any probability 0 or 1, so adjust probabilities slightly
Auto_Correct_Probabilities <- function(p)
{
  if (p > 0.9999)
  {
    p = 0.9999
  }else if (p < 0.0001)
  {
    p = 0.0001
  }
  return(p)
}

percent_ID = Auto_Correct_Probabilities(percent_ID)
diverse_score = Auto_Correct_Probabilities(diverse_score)
p_assay = Auto_Correct_Probabilities(p_assay)
percent_genus_db = Auto_Correct_Probabilities(percent_genus_db)
p_species = Auto_Correct_Probabilities(p_species)
p_aquamaps = Auto_Correct_Probabilities(p_aquamaps)
prob_dist = Auto_Correct_Probabilities(prob_dist)


res <- bugs(data=list(p_i = percent_ID, dr = diverse_score, counts =read_counts,
                      p_an = p_assay, p_gp = percent_genus_db, p_s=p_species,
                      p_aq = p_aquamaps, num_ta = num_in_area, p_di = prob_dist),
            inits = NULL,
            n.chains = 4, n.iter = num_iterations, n.burnin = num_burnin,
            parameters.to.save = c("ID", "RD","AN","GP","S","AQ","NT", "DI"),
            model.file = "OpenBUGS_eDNA.txt",
            DIC = FALSE, codaPkg = TRUE,debug=debug)

# extract the data
codaobj <- read.bugs(res, quiet=TRUE)
all_distributions <- as.data.frame(as.matrix(codaobj))

# Convert each distribution into a density
ID = density(all_distributions$ID)
RD = density(all_distributions$RD)
AN = density(all_distributions$AN)
```

```r
GP = density(all_distributions$GP)
S = density(all_distributions$S)
AQ = density(all_distributions$AQ)
NT = density(all_distributions$NT)
DI = density(all_distributions$DI)

# Make all plots have the same x-axis range
min_x = max(min(ID$x), min(RD$x), min(AN$x), min(GP$x), min(S$x), min(AQ$x), min(NT$x), min(DI$x))
max_x = min(max(ID$x), max(RD$x), max(AN$x), max(GP$x), max(S$x), max(AQ$x), max(NT$x), max(DI$x))
dx = 0.001
x = seq(min_x, max_x, dx)

# Override previous plots with plots in new range
ID = approx(ID$x, ID$y, xout=x)
RD = approx(RD$x, RD$y, xout=x)
AN = approx(AN$x, AN$y, xout=x)
GP = approx(GP$x, GP$y, xout=x)
S = approx(S$x, S$y, xout=x)
AQ = approx(AQ$x, AQ$y, xout=x)
NT = approx(NT$x, NT$y, xout=x)
DI = approx(DI$x, DI$y, xout=x)

Normalise_density <- function(y, dx)
{ # make the area under the curve 1
  area = sum(y)*dx
  return(y / area)
}

# percentage ID * number of reads * assay type * genus percentage in database * proportion of species
posterior_match = Normalise_density(ID$y, dx) * Normalise_density(RD$y, dx) * Normalise_density(AN$y,
posterior_match = Normalise_density(posterior_match, dx)

# aquamaps probability * number of species in target area * closest distance
posterior_location = Normalise_density(AQ$y, dx) * Normalise_density(NT$y, dx) * Normalise_density(DI
posterior_location = Normalise_density(posterior_location, dx)

combined_posterior = posterior_match * posterior_location
combined_posterior = Normalise_density(combined_posterior, dx)


# Now that we have the plots for the 3 posterior densities, we need to calculate the expected values
Get_Expected_value <- function(x,y,dx)
{
  # Approximate numerical approximation of expected value for distribution
  # Other numerical methods may be better, e.g Simpsons rule, etc
  expected_value = sum(x*y)*dx
  return(expected_value)
}

expected_c = Get_Expected_value(x, combined_posterior, dx)
expected_m = Get_Expected_value(x, posterior_match, dx)
expected_l = Get_Expected_value(x, posterior_location, dx)
```

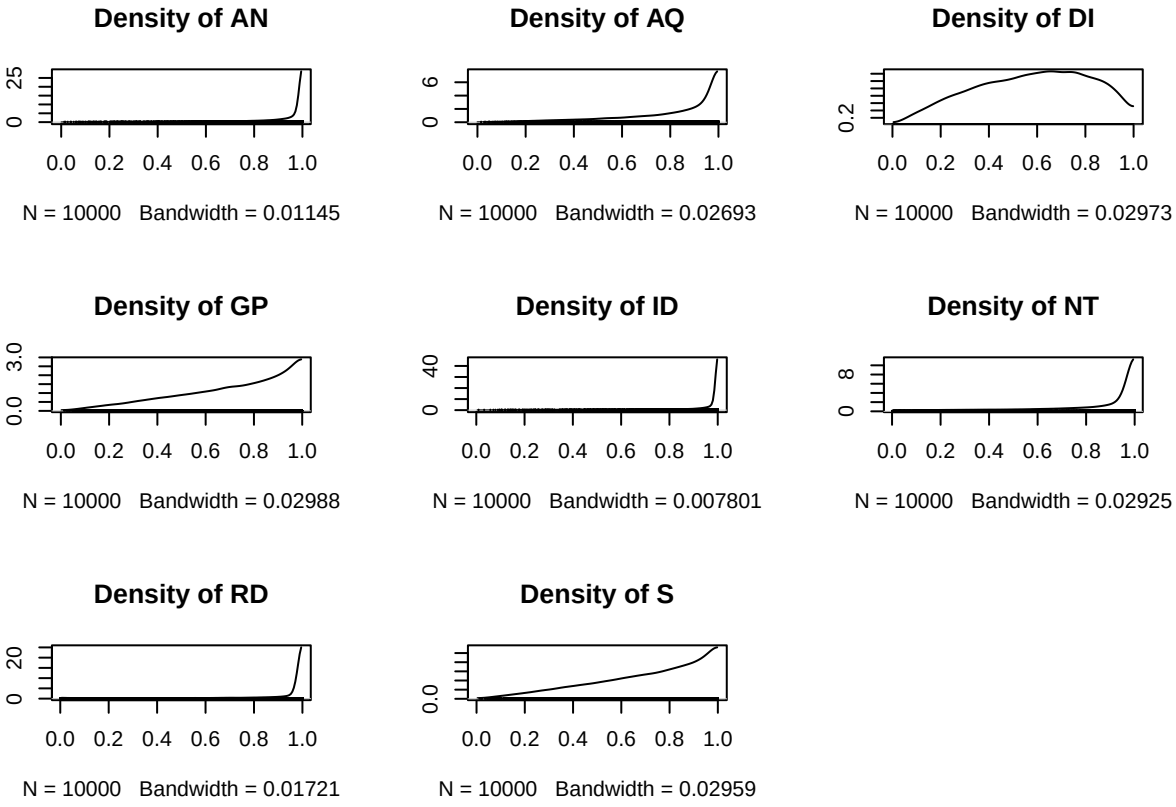
```
# Find the cumulative density function for credible intervals
cdf_c = cumsum(combined_posterior*dx)
cdf_m = cumsum(posterior_match*dx)
cdf_l = cumsum(posterior_location*dx)

# Normalise cumulative sum
cdf_c = cdf_c/max(cdf_c)
cdf_m = cdf_m/max(cdf_m)
cdf_l = cdf_l/max(cdf_l)

# Find the index of the credible intervals
conf_index_c = c( max(which(cdf_c < (1-confidence_int)/2)),   min(which(cdf_c > 0.5 + confidence_int/1
conf_index_m = c( max(which(cdf_m < (1-confidence_int)/2)),   min(which(cdf_m > 0.5 + confidence_int/1
conf_index_l = c( max(which(cdf_l < (1-confidence_int)/2)),   min(which(cdf_l > 0.5 + confidence_int/1

conf_c = c(x[conf_index_c[1]], x[conf_index_c[2]])
conf_m = c(x[conf_index_m[1]], x[conf_index_m[2]])
conf_l = c(x[conf_index_l[1]], x[conf_index_l[2]])

# intervals cannot be less than 0 and more than 1
conf_c[1] = max(conf_c[1],0)
conf_m[1] = max(conf_m[1],0)
conf_l[1] = max(conf_l[1],0)
conf_c[2] = min(conf_c[2],1)
conf_m[2] = min(conf_m[2],1)
conf_l[2] = min(conf_l[2],1)


# Save this information into a data frame
df <- data.frame(c("Combined Posterior Probability", "Match Probability", "Location Probability"),c(ex
names(df) <- c("Probability Type","Expected Value", "Lower Confidence Interval","Upper Confidence Inte

if (show_stats)
{
  print(summary(codaobj))
}

if (show_all_densities)
{
  plot(codaobj, trace=FALSE)
}

if (show_posterior_density)
{
  par(mfrow = c(2,2))

  # Total Posterior Density Plot
  plot(x,combined_posterior,main="Total Posterior with Credible Interval",xlab="Probability",ylab="Den
  lines(c(conf_c[1],conf_c[1]), c(0,combined_posterior[conf_index_c[1]]),lty = 2)
  lines(c(conf_c[2],conf_c[2]), c(0,combined_posterior[conf_index_c[2]]),lty = 2)

  # Match Posterior Density Plot
  plot(x,posterior_match,main="DNA Match Posterior with Credible Interval",xlab="Probability",ylab="De
```

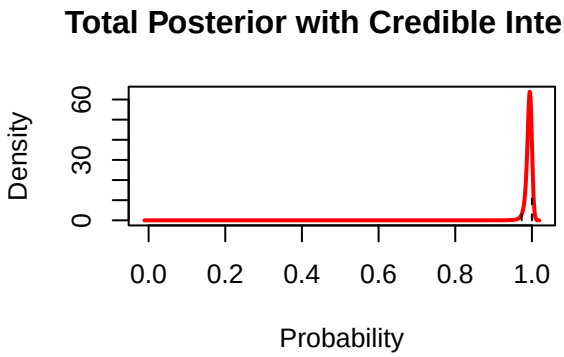
```
    lines(c(conf_m[1],conf_m[1]), c(0,posterior_match[conf_index_m[1]]),lty = 2)
    lines(c(conf_m[2],conf_m[2]), c(0,posterior_match[conf_index_m[2]]),lty = 2)

    # Location Posterior Density Plot
    plot(x,posterior_location,main="Location Posterior with Credible Interval",xlab="Probability",ylab=
    lines(c(conf_l[1],conf_l[1]), c(0,posterior_location[conf_index_l[1]]),lty = 2)
    lines(c(conf_l[2],conf_l[2]), c(0,posterior_location[conf_index_l[2]]),lty = 2)

    # Show all posterior densities on  the same plot
    plot(x,combined_posterior,main="All 3 Posteriors",xlab="Probability",ylab="Density",type='l',lwd=1,
    lines(x,posterior_match,type='l',lwd=1,col='blue',lty=2)
    lines(x,posterior_location,type='l',lwd=1,col='green',lty=3)
    legend("top",legend = c("Total", "Match", "Location"),col=c('red','blue','green'),lty=c(1,2,3))
  }

  if (show_trace)
  {
    plot(codaobj, density=FALSE)

  }

  return(df)
}
```

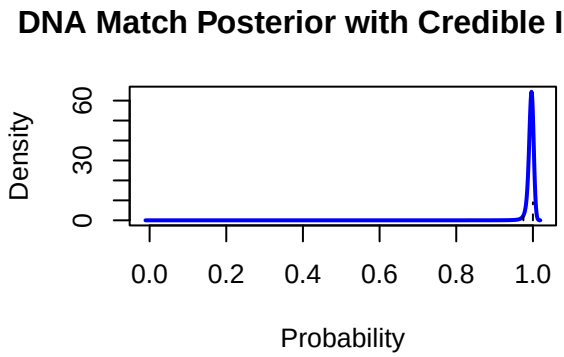
```
# Test everything:

percent_ID = 0.9
read_counts = 20
p_assay = 0.9
diverse_score = 0.2
percent_genus_db = 0.7
p_species = 0.7
p_aquamaps = 0.8
num_in_area = 4
nearest_distance = 40
distance_for_50_percent = 50
num_iterations = 11000
num_burnin = 1000
show_trace=T
show_all_densities=T
show_posterior_density=T
show_stats=T
debug=F

test_probs = eDNA_OpenBUGS(percent_ID, read_counts, p_assay, diverse_score, percent_genus_db, p_species=
                           distance_for_50_percent =   distance_for_50_percent,num_iterations=num_itera
                           show_all_densities=show_all_densities,show_posterior_density=show_posterior_
```

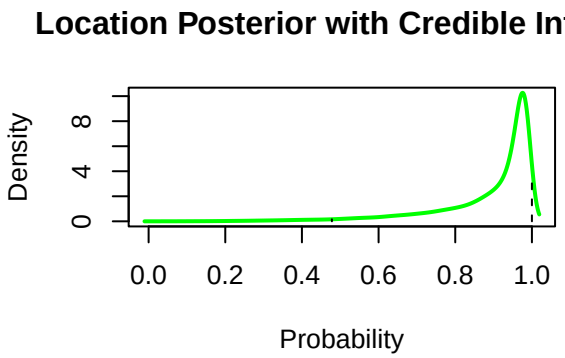
```
##
## Iterations = 1001:11000
## Thinning interval = 1
## Number of chains = 4
## Sample size per chain = 10000
##
```

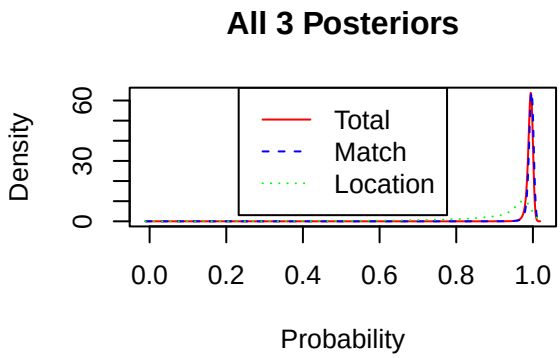
```
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##      Mean     SD  Naive SE Time-series SE
## AN 0.9008 0.1664 0.0008322      0.0008245
## AQ 0.7996 0.2146 0.0010728      0.0010769
## DI 0.5746 0.2335 0.0011674      0.0011616
## GP 0.6984 0.2347 0.0011735      0.0011668
## ID 0.9202 0.1519 0.0007593      0.0007493
## NT 0.8008 0.2663 0.0013316      0.0013316
## RD 0.8472 0.2646 0.0013231      0.0013042
## S  0.6991 0.2324 0.0011621      0.0011664
##
## 2. Quantiles for each variable:
##
##       2.5%    25%    50%    75%  97.5%
## AN 0.38410 0.8787 0.9820 0.9992 1.0000
## AQ 0.24530 0.6882 0.8794 0.9716 0.9997
## DI 0.11980 0.3956 0.5925 0.7639 0.9532
## GP 0.17380 0.5372 0.7474 0.8994 0.9935
## ID 0.43169 0.9178 0.9924 0.9999 1.0000
## NT 0.09568 0.6881 0.9380 0.9960 1.0000
## RD 0.08056 0.8189 0.9950 1.0000 1.0000
## S  0.17890 0.5404 0.7483 0.8964 0.9934
```

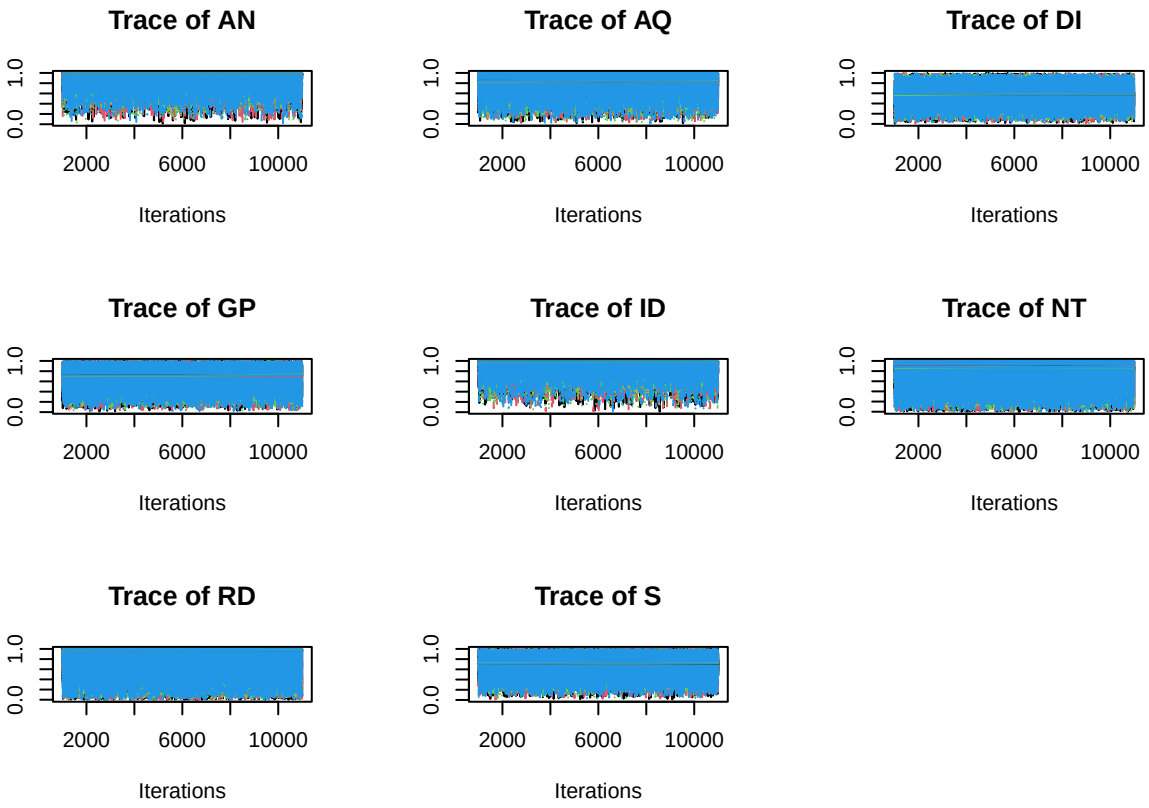



Total Posterior with Credible Interval
Density
0
30
60
0.0
0.2
0.4
0.6
0.8
1.0
Probability


DNA Match Posterior with Credible Inter
Density
0
30
60
0.0
0.2
0.4
0.6
0.8
1.0
Probability


Location Posterior with Credible Interv
Density
0
4
8
0.0
0.2
0.4
0.6
0.8
1.0
Probability


All 3 Posteriors
Density
0
30
60
Total
Match
Location
0.0
0.2
0.4
0.6
0.8
1.0
Probability



Takes roughly 5 seconds, 5s times 1,067,313 rows is 5336565s, or 62 days, way too long, needs to be about 1 day

```
test_probs
```

```
##                 Probability Type Expected Value Lower Confidence Interval
## 1 Combined Posterior Probability      0.9924091                 0.9731944
## 2              Match Probability      0.9944791                 0.9751944
## 3           Location Probability      0.8869968                 0.4781944
##   Upper Confidence Interval                          Info
## 1                         1 Using  95 % credible interval
## 2                         1 Using  95 % credible interval
## 3                         1 Using  95 % credible interval
```