

# Delaware River PIT tag data analysis

Ben Letcher

2022-07-24

*Note:* the results shown here are preliminary and have not been officially reviewed by USGS, NYDEC or PA Fish and Boat

This notebook uses targets to manage running code and updating R objects. Targets sets up dependencies among specified objects and only re-runs code as necessary (when an upstream component gets updated). This can save run times for projects with models that take a while to run, like capture-mark-recapture models.

Data preparation and model running happens using targets and exploration of the data and model runs is below in this Markdown document.

'knit' the document to update all targets and the markdown exploration below.

'tar\_make()' runs all the R scripts and functions specified in '\_targets.R'. Only updated code or sections that are downstream from updated data are re-run.

'tar\_read()' reads 'target' data into the global environment.

To set up a targets project, use use\_targets()

This section (tar\_make()) reruns the model

```
# tar_watch(seconds = 10, outdated = FALSE, targets_only = TRUE)

# comment this out when knitting - get Latex error that it can't find the check mark the tar_make() uses
#tar_make()

# tar_prune() # cleans unused data files
#tar_invalidate(everything())
#tar_invalidate(ends_with("ttt"))

#str(d)
```

Load data for analysis

Without a postscript, the data are for all fish (main and trib)

```
dRaw0 <- tar_read(dRaw0) #all data - including untagged
dRaw <- tar_read(dRaw) #all data for CMR models
d <- tar_read(target_d)
eh <- tar_read(target_eh)
```

Visualize the network - does not work with pdf output

```
#tar_visnetwork()
```

Which rivers (Water) riverN corresponds to

```
table(d$Water, d$riverN)
#>
#>           1    2    3    4    5    6
#>  Balls Creek  41    0    0    0    0    0
#> Cold Spring Creek  0  95    0    0    0    0
#>  Roods Creek    0    0  159    0    0    0
#>  Sands Creek    0    0    0  139    0    0
#> Shehawken Creek  0    0    0    0  91    0
#> West Br Delaware River  0    0    0    0    0 5634
```

Raw data summary tables

```
kable(data.frame(ftable(d$date)))
```

Var1	Freq
2018-05-07	110
2018-05-08	34
2018-05-09	48
2018-06-11	129
2018-06-12	99
2018-06-13	88
2018-07-16	212
2018-07-17	176
2018-07-18	142
2018-08-21	11
2018-09-17	21
2018-09-20	89
2018-10-22	85
2018-10-23	131
2018-10-24	129
2019-04-08	240
2019-04-10	129
2019-05-06	170
2019-05-07	91
2019-06-10	169
2019-06-11	129
2019-07-15	212
2019-07-16	312
2019-07-17	25
2019-08-12	131
2019-08-13	139
2019-08-14	186
2019-08-15	49
2019-09-16	108
2019-09-17	55

Var1	Freq
2019-09-18	293
2019-10-21	262
2019-10-22	31
2019-10-23	74
2020-07-16	146
2020-07-20	249
2020-07-21	29
2020-08-10	89
2020-08-11	41
2020-08-17	145
2020-08-20	110
2020-09-10	187
2020-09-14	252
2020-09-15	47
2020-10-13	368
2020-10-14	55
2020-10-15	132

```
#kable(data.frame(ftable(d$Water, d$riverN)))

#kable(data.frame(ftable(d$Water, d$riverN, d$date)))
kable(data.frame(ftable(d$species)))
```

Var1	Freq
brook trout	13
brown trout	5534
rainbow trout	611

```
### Number of unique tags
length(unique(d$tag))
#> [1] 4610
```

Group observations by month.

Luckily, sampling periods do not span months, so we can use month as a grouping variable for sampling occasion

```
kable(data.frame(ftable(d$dateYM)))
```

Var1	Freq
2018-05	192
2018-06	316
2018-07	530
2018-08	11
2018-09	110
2018-10	345
2019-04	369
2019-05	261

Var1	Freq
2019-06	298
2019-07	549
2019-08	505
2019-09	456
2019-10	367
2020-07	424
2020-08	385
2020-09	486
2020-10	555

## Tag information

Grouped by Water (sampling area)

```
tagN <- d %>%
  group_by(tag, Water) %>%
  summarize(n = n()) %>%
  filter(tag != "") %>%
  arrange(desc(n))
#> `summarise()` has grouped output by 'tag'. You can override using the `.groups`
#> argument.

### Number of times individual fish were observed
table(tagN$n)
#>
#>      1      2      3      4      5      6      7      8      9
#> 3641  611  223   90   30   12    4    1    1

### Number of times individual fish were observed by river
(table(tagN$Water, tagN$n))
#>
#>
#>           1      2      3      4      5      6      7      8      9
#> Balls Creek      30    4    1    0    0    0    0    0    0
#> Cold Spring Creek 39    9    4    2    1    1    1    0    0
#> Roods Creek      78   21    6    4    1    0    0    0    0
#> Sands Creek      97   16    2    1    0    0    0    0    0
#> Shehawken Creek   45   10    6    2    0    0    0    0    0
#> West Br Delaware River 3352 551 204  81  28  11   3   1   1
```

Grouped by main/trib

```
tagN_mt <- d %>%
  group_by(tag, mainTrib) %>%
  summarize(n = n()) %>%
  filter(tag != "") %>%
  arrange(desc(n))
#> `summarise()` has grouped output by 'tag'. You can override using the `.groups`
#> argument.

### Number of times individual fish were observed
```

```
table(tagN_mt$n)
#>
#>   1    2    3    4    5    6    7    8    9
#> 3641 611 223  90  30  12   4   1   1

### Number of times individual fish were observed by river
(table(tagN_mt$mainTrib, tagN_mt$n))
#>
#>      1    2    3    4    5    6    7    8    9
#> main 3352 551 204  81  28  11   3   1   1
#> trib  289  60  19   9   2   1   1   0   0
```

Grouped by state

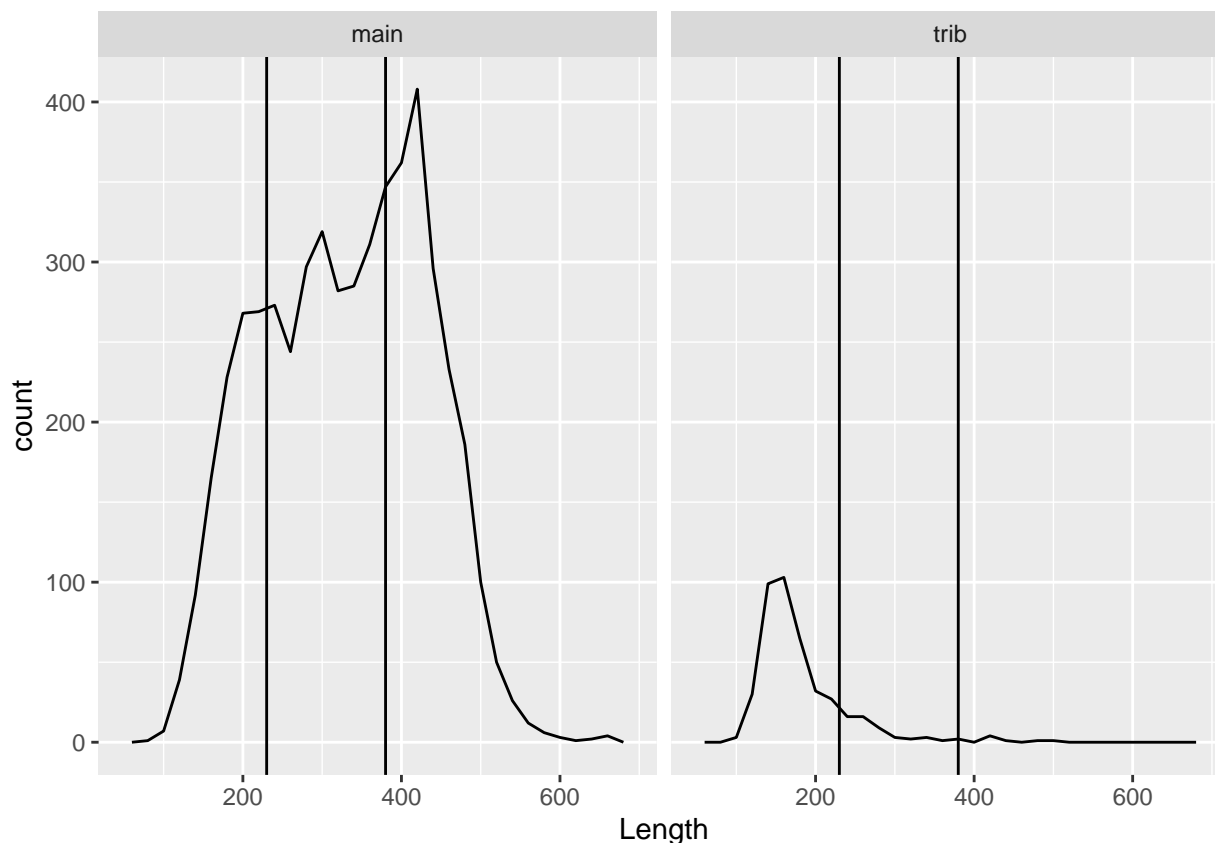
```
tagN_s <- d %>%
  group_by(tag, state) %>%
  summarize(n = n()) %>%
  filter(tag != "") %>%
  arrange(desc(n))
#> `summarise()` has grouped output by 'tag'. You can override using the `.groups`
#> argument.

### Number of times individual fish were observed
table(tagN_s$n)
#>
#>   1    2    3    4    5    6    7    9
#> 3949 640 170  57  23   9   2   1

### Number of times individual fish were observed by river
table(tagN_s$state, tagN_s$n)
#>
#>      1    2    3    4    5    6    7    9
#> 1  963   69   4    0    0    0    0    0
#> 2 1660  274   58  17   7    0    0    0
#> 3 1025  232   87  35  15   8    2    1
#> 4  264   59  15   5   1   1    0    0
#> 5   29    6   6   0   0   0    0    0
#> 6    8    0   0   0   0   0    0    0
```

Basic summary plots of raw tagging data

```
ggplot(d %>% filter(species == "brown trout"), aes(Length)) +
  geom_freqpoly() +
  geom_vline(xintercept = c(tar_read(target_sizeCutoff1), tar_read(target_sizeCutoff2))) +
  facet_grid(~mainTrib)
#> `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
# dTame <- d %>%
#       select(Latitude, Longitude, tag, dateTime, species, Length, Weight) %>%
#       filter(tag != "", tag != "ad")
#
# write.csv(dTame, './dataOut/dTame.csv', row.names = FALSE)
```

## Encounter histories

This is the data structure for the capture-recapture models. Each column is a sampling ‘occasion’ (here = month) and each row is an individual, where a ‘1’ indicates capture and a ‘0’ indicates not captured.

```
str(eh$eh)
#>  num [1:3673, 1:17] 1 1 1 1 1 1 1 1 1 1 ...
#>  - attr(*, "dimnames")=List of 2
#>    ..$ : NULL
#>    ..$ : chr [1:17] "date_2018-05" "date_2018-06" "date_2018-07" "date_2018-08" ...
kable(head(eh$eh,8))
```

date	2018-05	2018-06	2018-07	2018-08	2018-09	2018-10	2018-04	2018-05	2018-06	2018-07	2018-08	2018-09	2018-10	2018-07	2018-08	2018-09	2018-10
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

date	2018-05	2018-06	2018-07	2018-08	2018-09	2018-10	2018-04	2019-05	2019-06	2019-07	2019-08	2019-09	2019-10	2019-07	2020-08	2020-09	2020-10
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

```
table(paste(eh$first, eh$last, sep="_"))
#>
#> 1_17 10_17 11_17 12_17 13_17 14_17 15_17 16_17 2_17 3_17 4_17 5_17 6_17
#> 173 360 295 264 257 297 246 291 264 373 8 62 231
#> 7_17 8_17 9_17
#> 233 130 189
```

Transitions  
*River size1 size2 size3*  
Main 1 2 3  
Trib 4 5 6

There are very few transitions between main and trib and we can characterize size transitions better with a growth model - run phiT\_pT models separately for main and trib

```
s <- eh$stateMatrix
s[s == 0] <- NA

s2 <- data.frame(s) %>%
  unite("all", sep = "", na.rm = TRUE, remove = TRUE) %>%
  #distinct() %>%
  arrange()

table(s2)
#> all
#>      1      11      111      1112      112      1122      12      122
#>    774     58       3        1        6        3       34       14
#>   1222    132       2       22       222     2222    22222    2223
#>      5       1    1097     185      46      17       7        4
#>   22233    223    2233    22333    223333    2233333    22333333    23
#>      2      25      10        6        2        1        1       53
#>     233    2333    23333    233333    253        3       32       33
#>      33      8       1        2        1      687        2      172
#>     333    3333    33333    333333    3333333    33333333    4       42
#>      71     32     12       7        2        1      182        1
#>      44     444    4444    44444    444444    44445    4444555    4445
#>      47     12      3        1        1        1        1        1
#>     4455     45     455     4555     5      5455     55      555
#>       2       2       2        2      17        1        1        2
#>     556      6      63
#>       1       6       1
```

Summary info for years and occasions

```
years <- colnames(eh$eh) %>%
  substr(6,9) %>%
  as.numeric()

occs <- colnames(eh$eh)
```

## Models

‘phi’ = apparent survival (probability of staying in the area = p(survival) + p(not moving out of area)).  
‘p’ = probability of capture given that the fish is alive.

## phiT\_pT\_main

Load data ‘main’ for analysis

```
#d_tt <- tar_read(target_d_trib)
eh_main <- tar_read(target_eh_main)

str(eh_main$eh)
#>  num [1:3388, 1:16] 1 1 1 1 1 1 1 1 1 1 ...
#>  - attr(*, "dimnames")=List of 2
#>    ..$ : NULL
#>    ..$ : chr [1:16] "date_2018-05" "date_2018-06" "date_2018-07" "date_2018-09" ...
kable(head(eh_main$eh,8))
```

date	2018-05	2018-06	2018-07	2018-09	2018-10	2018-04	2018-05	2018-06	2018-07	2018-08	2018-09	2018-10	2018-07	2018-08	2018-09	2018-10
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

```
table(paste(eh_main$first, eh_main$last, sep="_"))
#>
#>  1_16 10_16 11_16 12_16 13_16 14_16 15_16 2_16 3_16 4_16 5_16 6_16 7_16
#>  173  242  244  238  297  246  291  264  313  51  216  233  130
#>  8_16 9_16
#>  189  261
```

```
### Read the model run into global memory
if (tar_exist_objects(c("tt_modelOut_main"))){
  mod_tt_main <- tar_read(tt_modelOut_main)

  #MCMCplot(object = mod_tt_main$mcmc, params = "betaPhiOut")
  #MCMCplot(object = mod_tt_main$mcmc, params = "betaPOut")
}
```

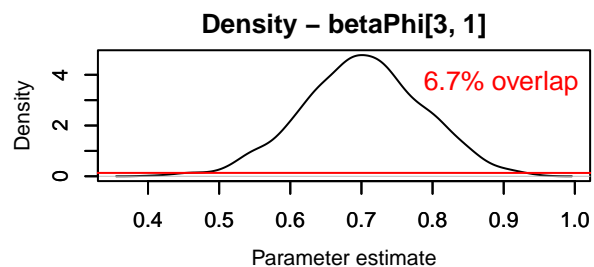
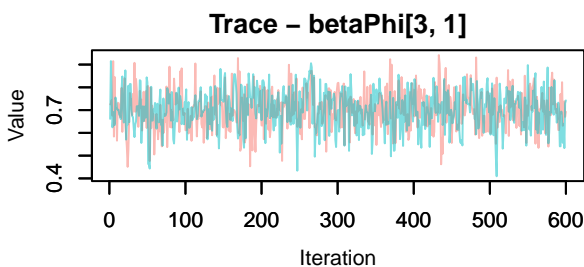
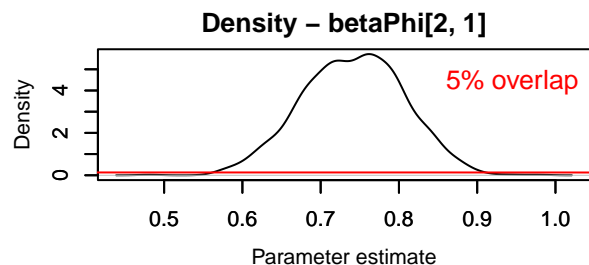
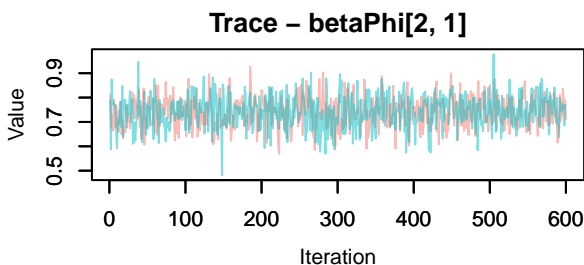
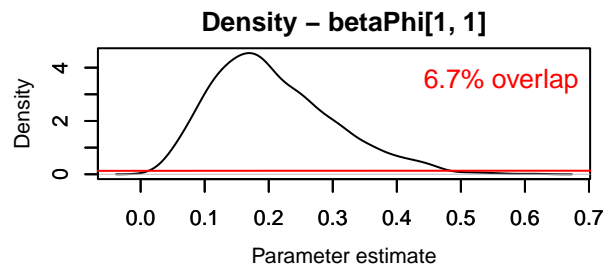
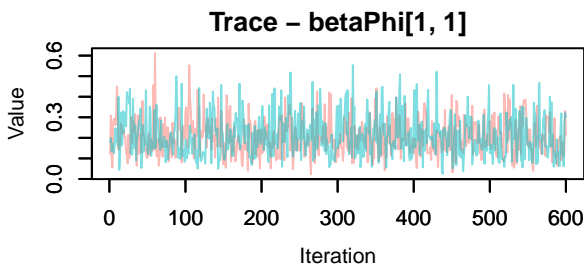


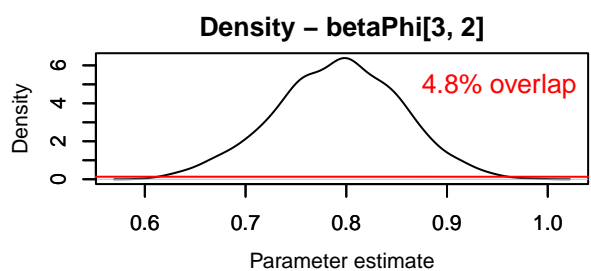
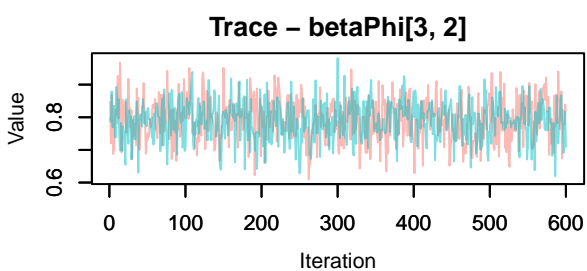
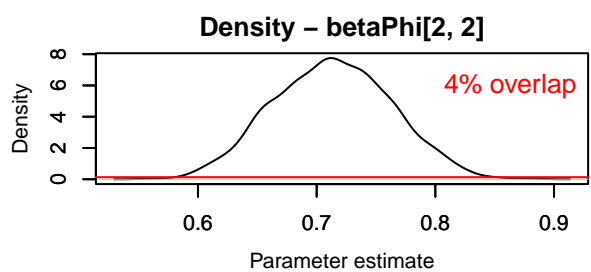
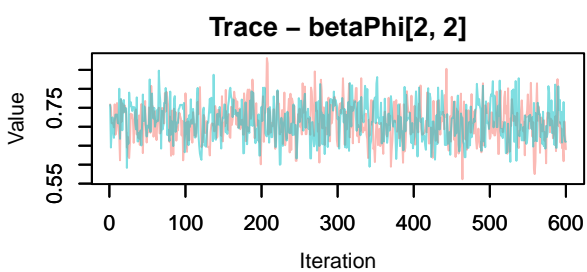
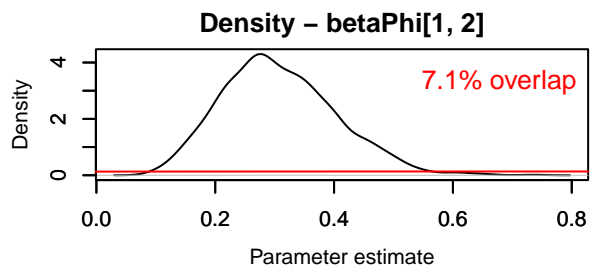
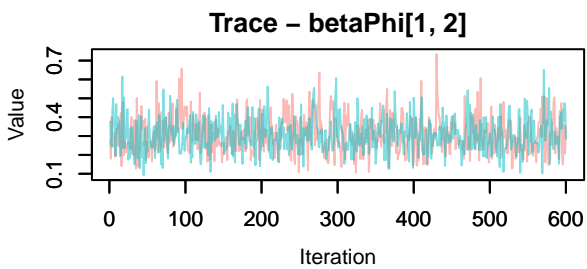
```
priors <- rnorm(tar_read(tt_runData_main)$nIter * tar_read(tt_runData_main)$nChains, 0, 1/sqrt(.1))
MCMCtrace(object = mod_tt_main$mcmc,
           #ISB = FALSE,
           #exact = TRUE,
           params = c("betaPhi"),
           pdf = FALSE,
           priors = priors
           )

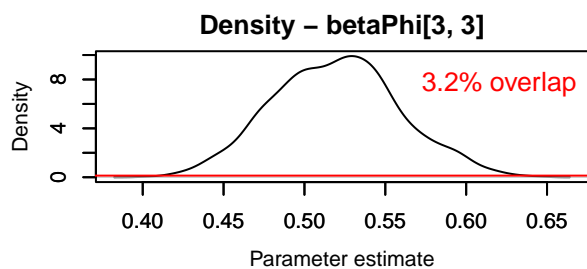
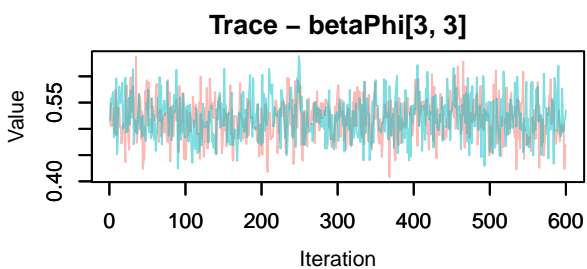
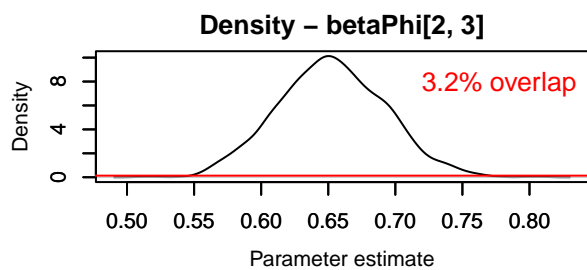
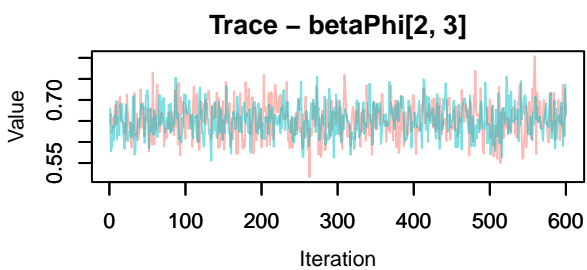
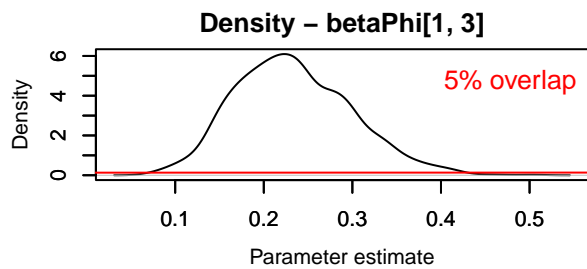
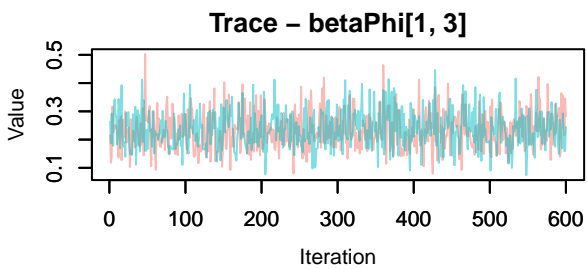
(modSummary_tt_main <- MCMCsummary(object = mod_tt_main$mcmc, round = 3))
}
```

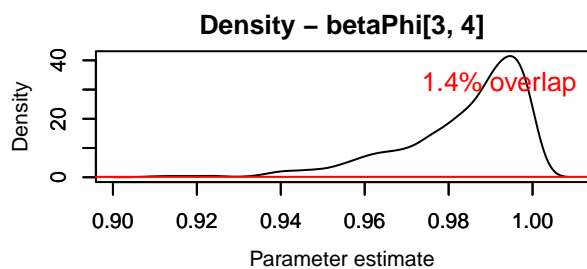
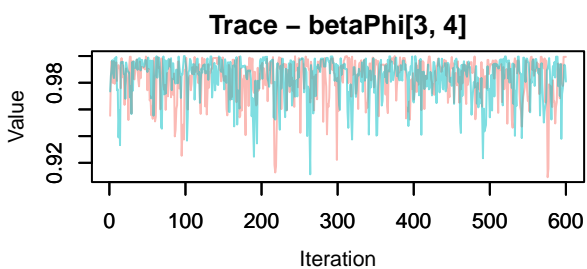
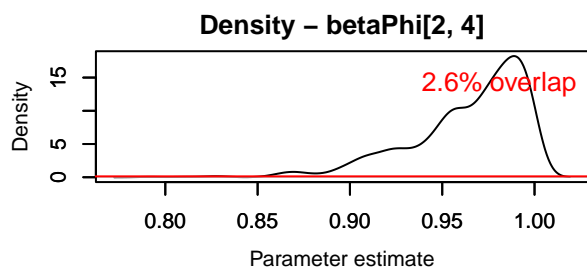
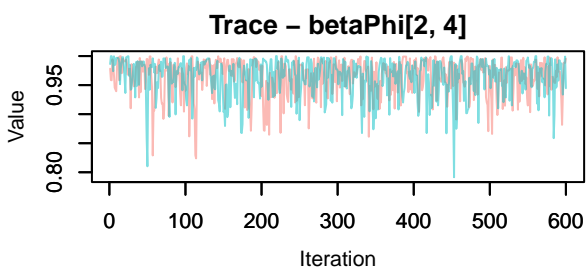
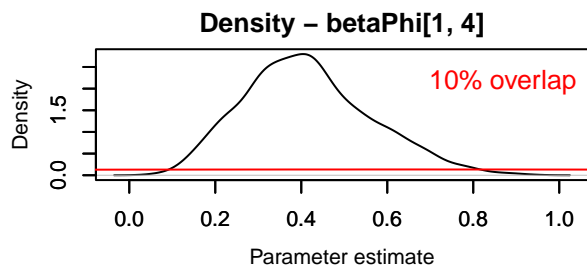
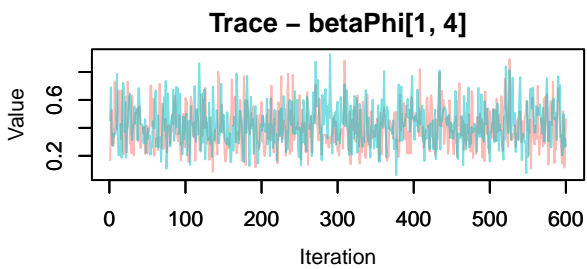
*#> Warning in MCMCtrace(object = mod\_tt\_main\$mcmc, params = c("betaPhi"), pdf = FALSE, : Only one prior specified for > 1 parameter. Using a single prior for all parameters.*

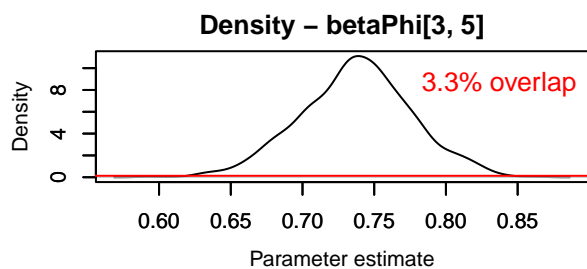
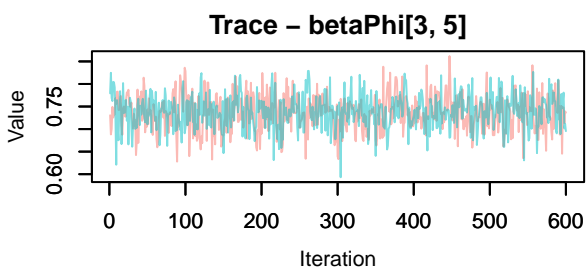
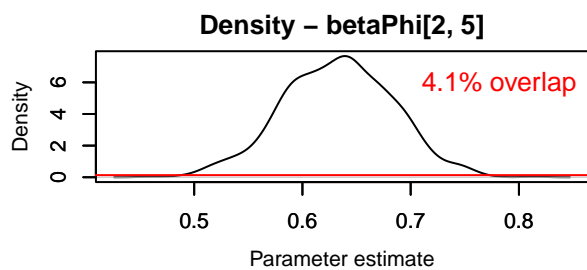
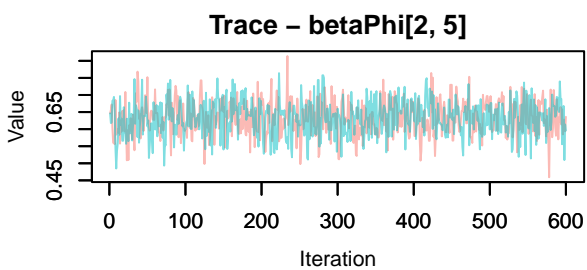
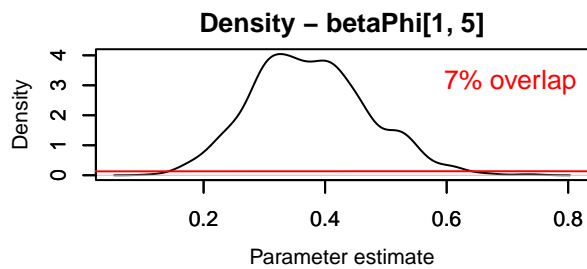
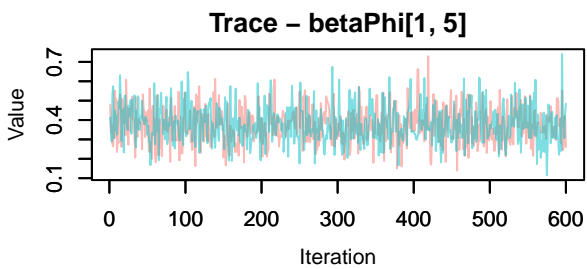
*#> Warning in MCMCtrace(object = mod\_tt\_main\$mcmc, params = c("betaPhi"), pdf = FALSE, : Number of samples in prior is greater than number of total or specified iterations (for all chains) for specified parameter. Only last 1200 iterations will be used.*

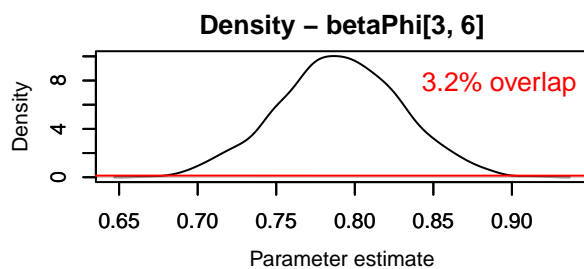
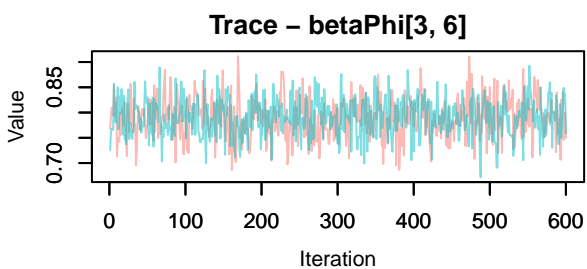
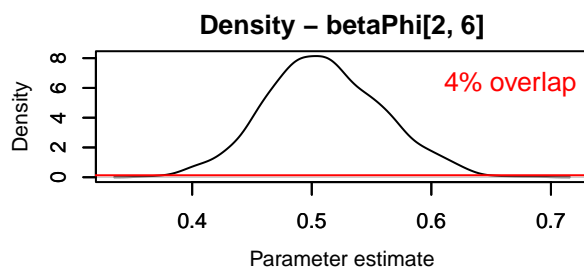
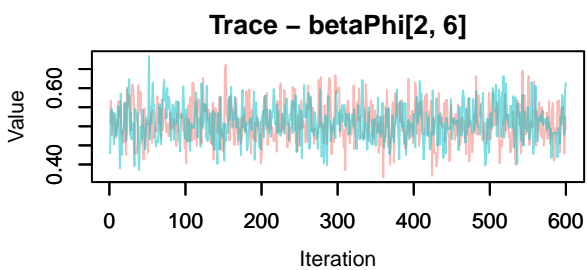
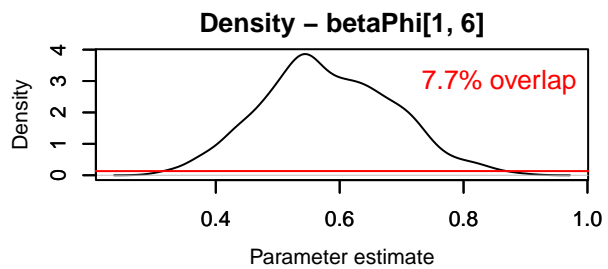
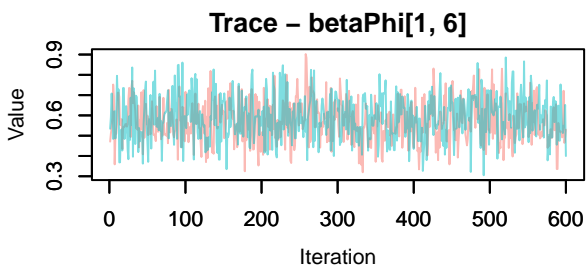


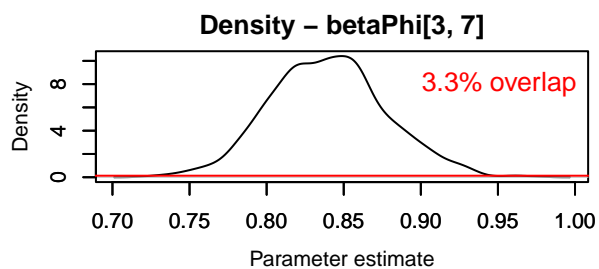
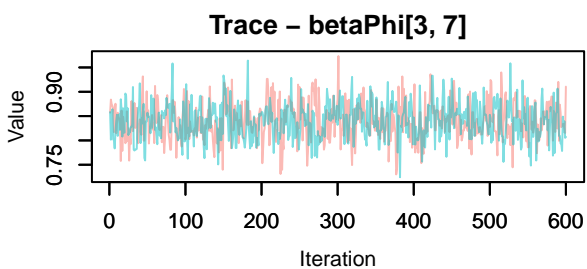
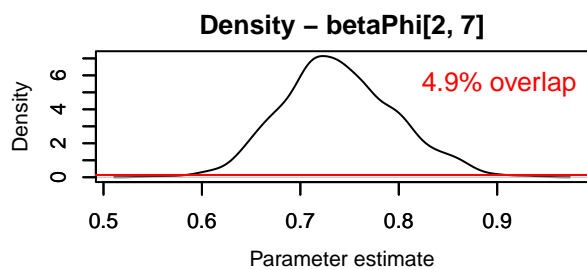
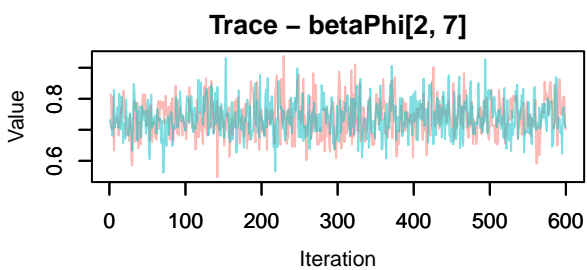
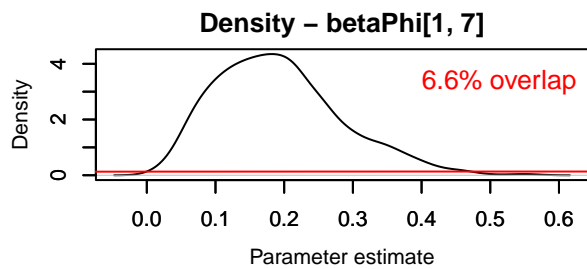
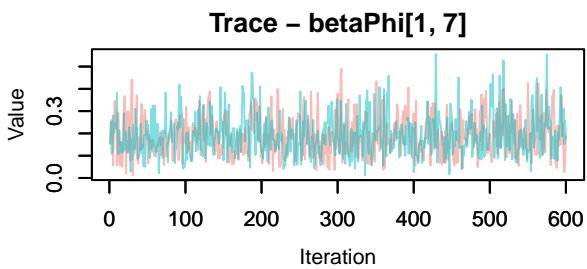


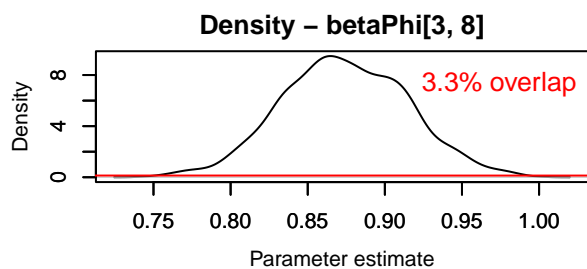
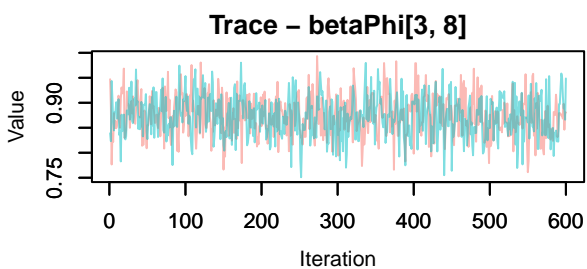
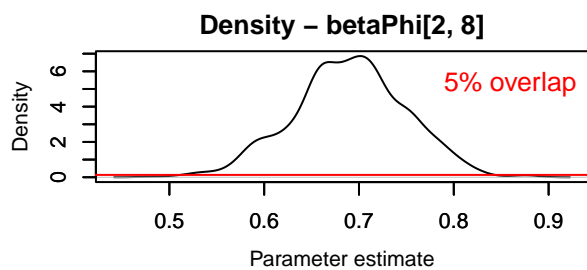
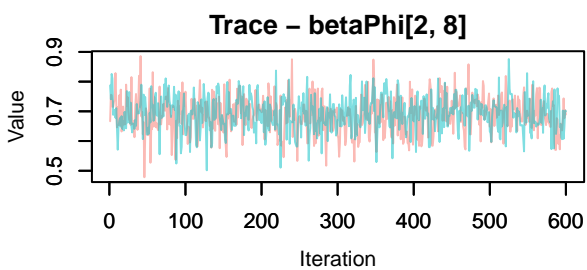
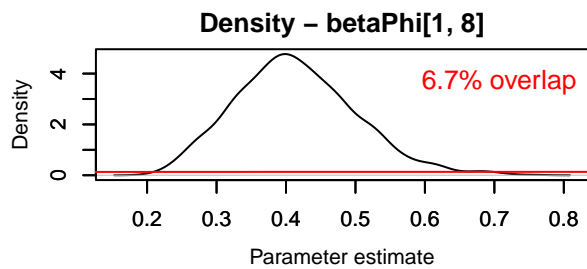
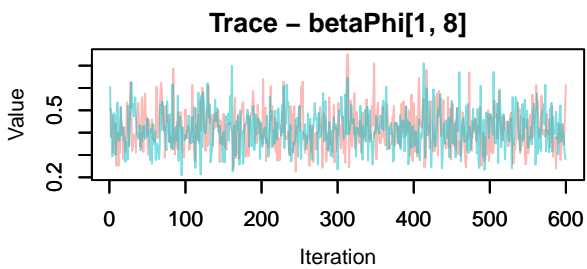




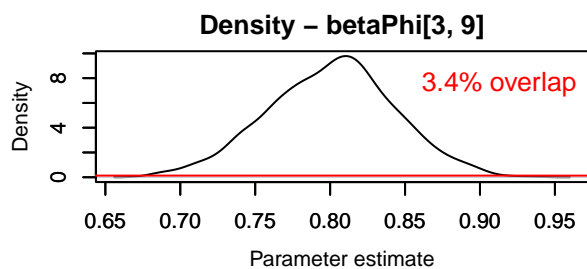
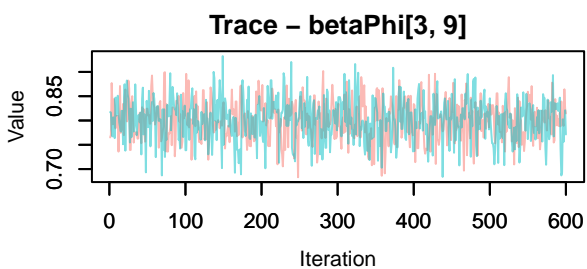
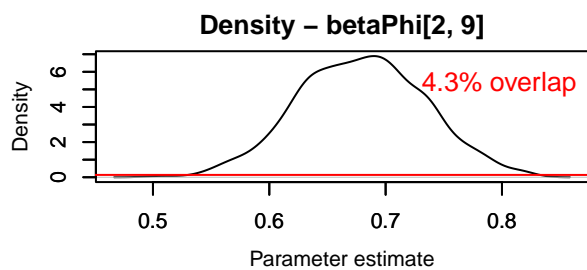
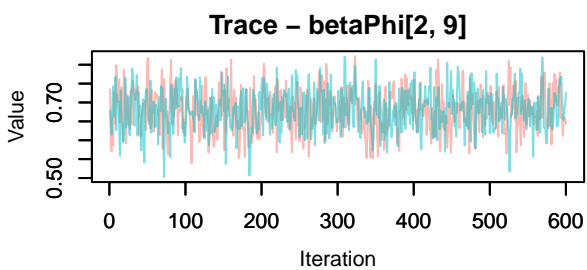
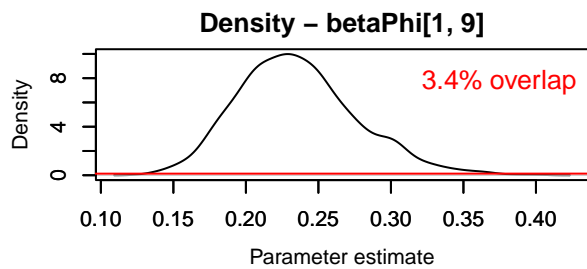
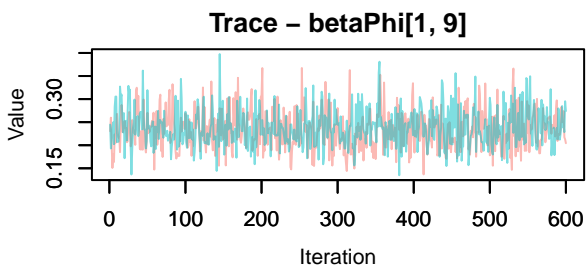


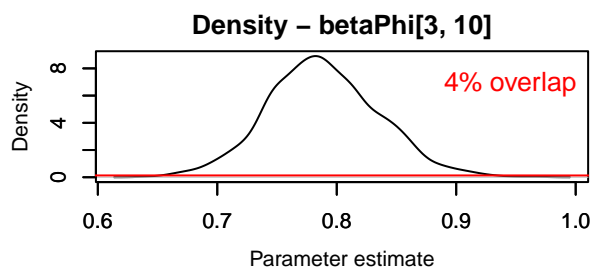
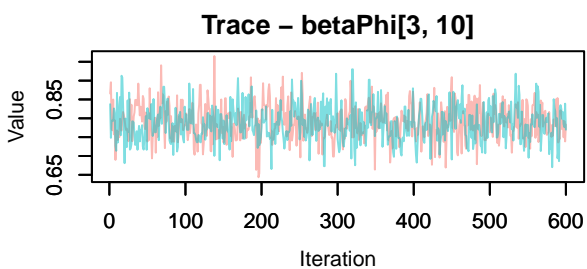
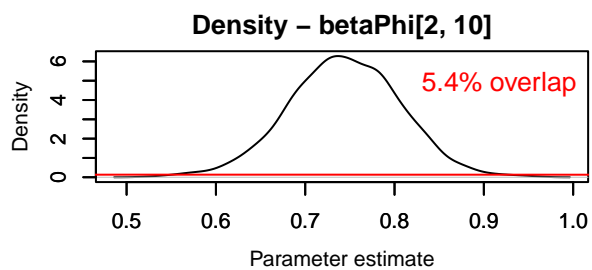
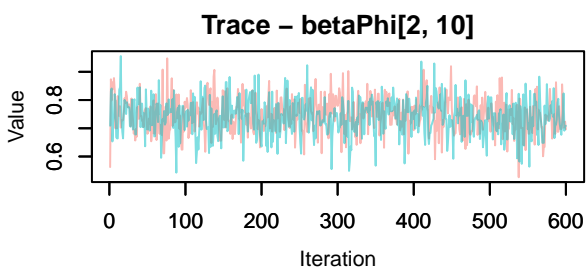
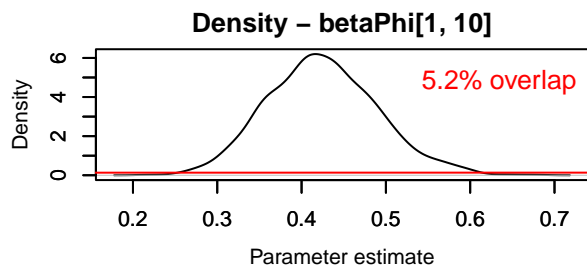
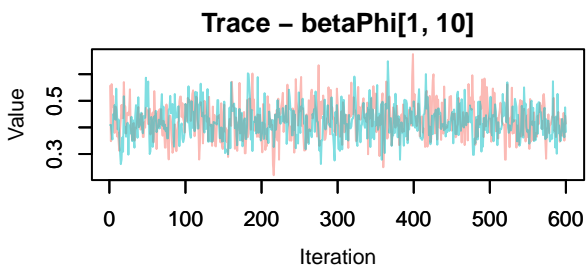


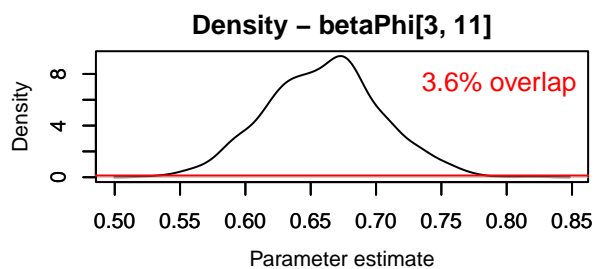
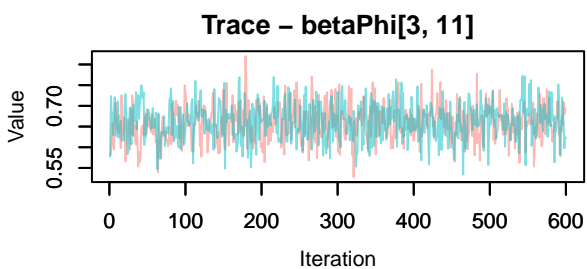
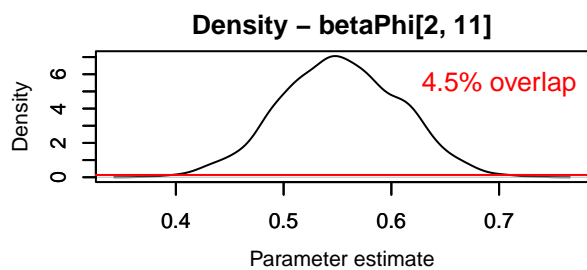
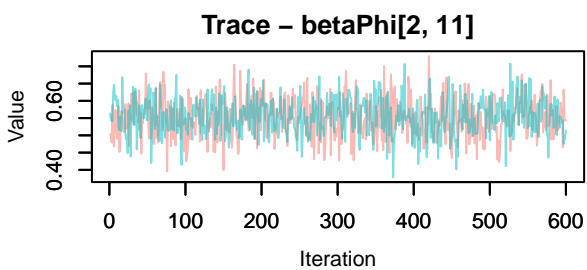
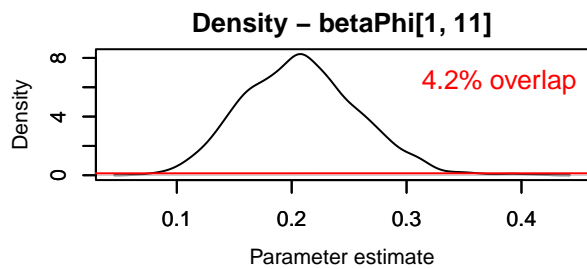
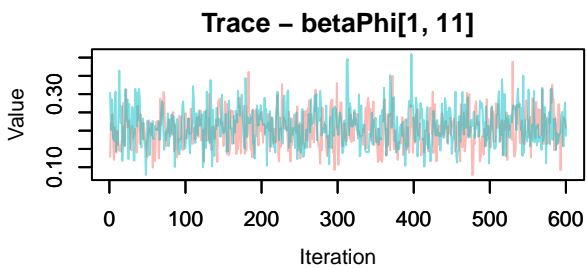


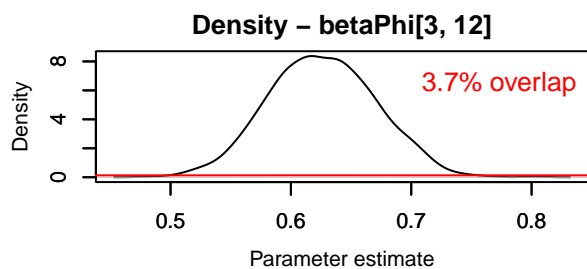
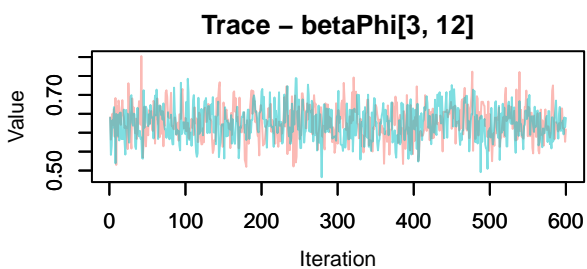
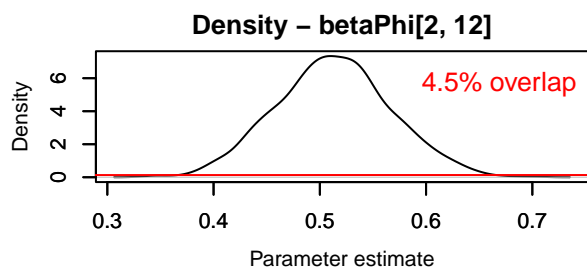
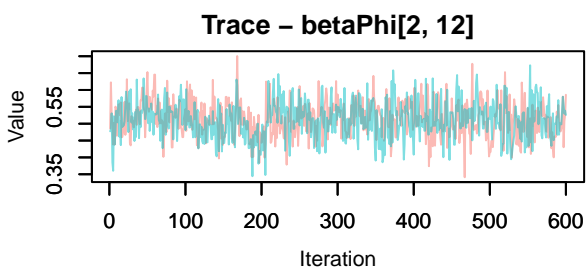
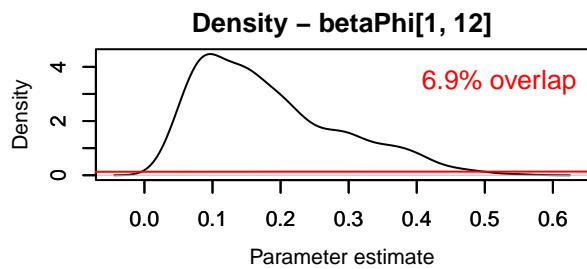
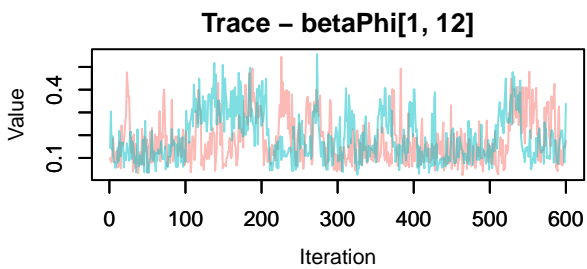


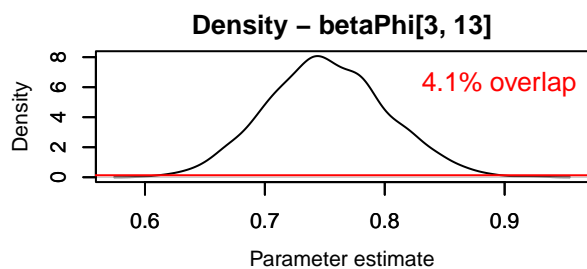
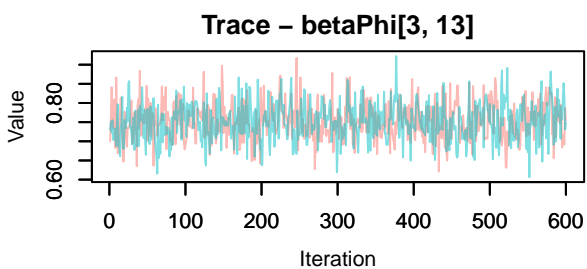
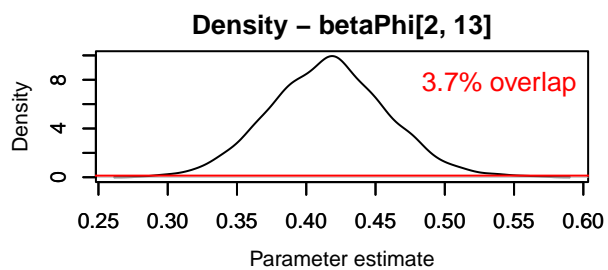
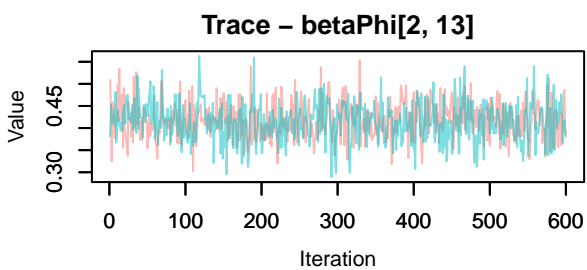
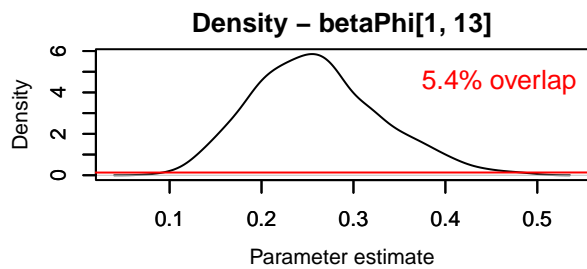
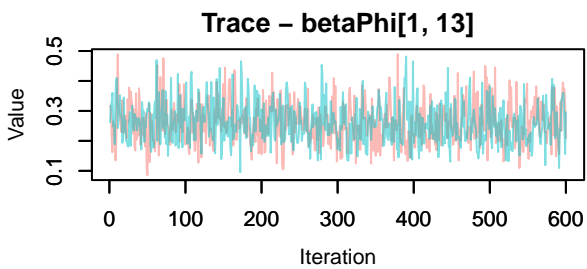


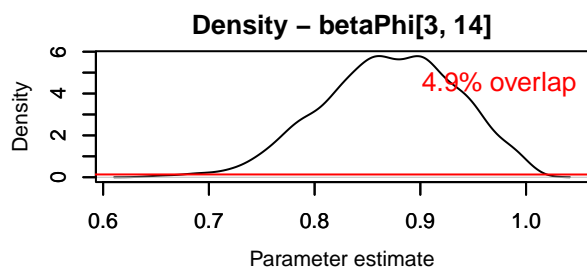
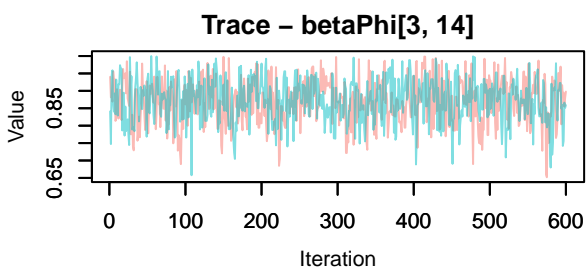
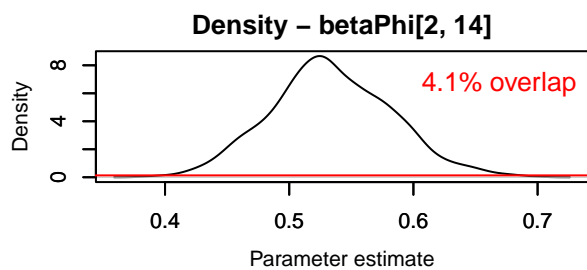
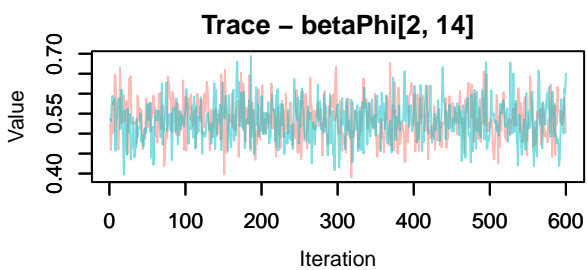
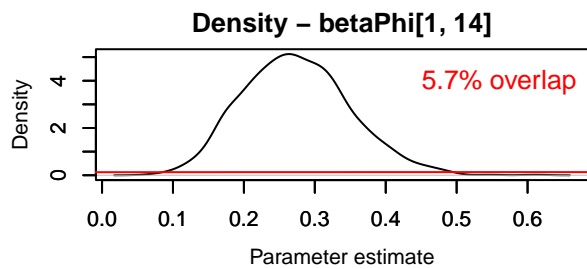
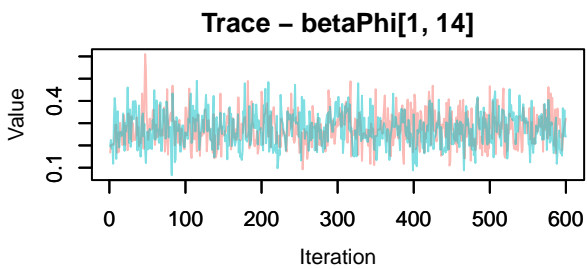


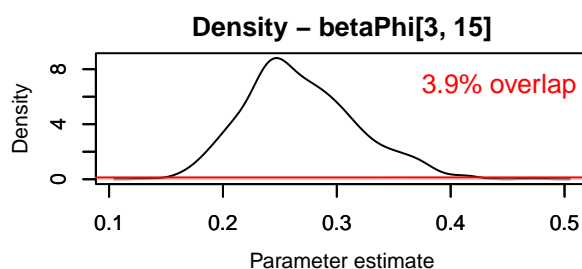
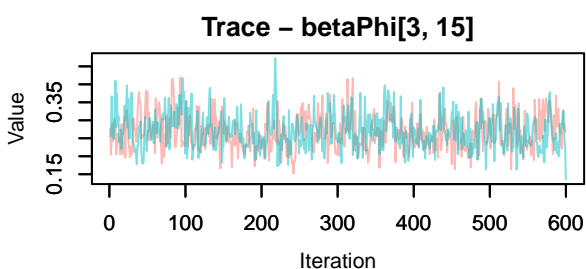
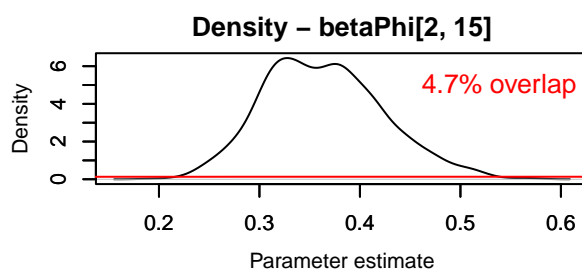
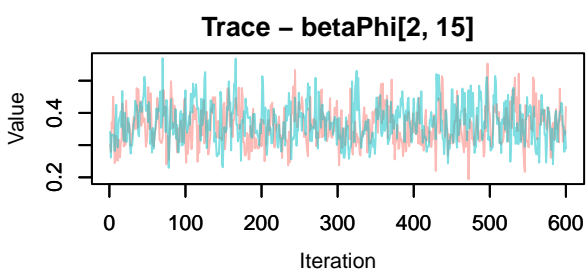
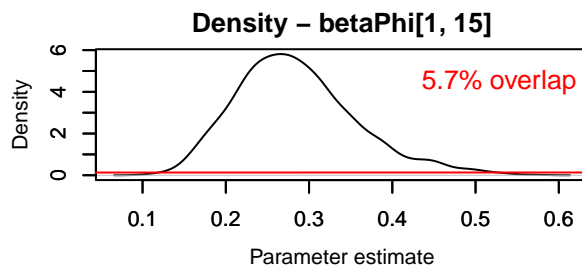
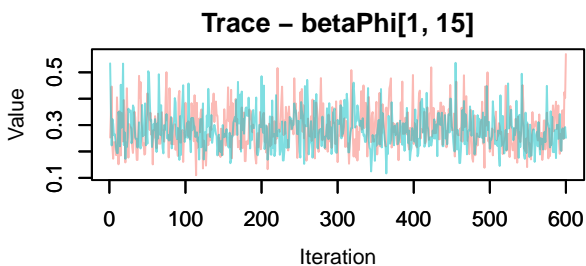












#>		mean	sd	2.5%	50%	97.5%	Rhat	n.eff
#>	betaP[1, 1]	0.439	0.292	0.012	0.414	0.967	1.00	847
#>	betaP[2, 1]	0.064	0.061	0.002	0.046	0.232	1.00	661
#>	betaP[3, 1]	0.061	0.061	0.002	0.042	0.212	1.02	591
#>	betaP[1, 2]	0.294	0.247	0.011	0.220	0.883	1.00	826
#>	betaP[2, 2]	0.028	0.027	0.001	0.019	0.100	1.01	654
#>	betaP[3, 2]	0.048	0.043	0.001	0.036	0.165	1.00	670
#>	betaP[1, 3]	0.280	0.245	0.007	0.208	0.901	1.00	803
#>	betaP[2, 3]	0.018	0.017	0.000	0.013	0.064	1.00	498
#>	betaP[3, 3]	0.012	0.012	0.000	0.009	0.044	1.00	585
#>	betaP[1, 4]	0.294	0.239	0.010	0.225	0.870	1.00	1044
#>	betaP[2, 4]	0.027	0.027	0.001	0.018	0.096	1.01	467
#>	betaP[3, 4]	0.014	0.014	0.000	0.009	0.053	1.06	473
#>	betaP[1, 5]	0.191	0.177	0.004	0.139	0.690	1.02	598
#>	betaP[2, 5]	0.044	0.044	0.001	0.030	0.160	1.01	478
#>	betaP[3, 5]	0.012	0.012	0.000	0.008	0.044	1.01	508
#>	betaP[1, 6]	0.387	0.278	0.013	0.344	0.940	1.00	830
#>	betaP[2, 6]	0.032	0.030	0.001	0.022	0.113	1.02	499
#>	betaP[3, 6]	0.012	0.012	0.000	0.009	0.042	1.03	417
#>	betaP[1, 7]	0.403	0.274	0.018	0.348	0.925	1.01	805
#>	betaP[2, 7]	0.028	0.027	0.001	0.020	0.100	1.04	587
#>	betaP[3, 7]	0.011	0.010	0.000	0.008	0.039	1.03	511
#>	betaP[1, 8]	0.141	0.135	0.003	0.098	0.502	1.00	519
#>	betaP[2, 8]	0.034	0.033	0.001	0.024	0.113	1.02	527
#>	betaP[3, 8]	0.015	0.015	0.000	0.010	0.054	1.01	532
#>	betaP[1, 9]	0.102	0.097	0.004	0.075	0.365	1.01	430

```

#> betaP[2, 9]      0.032 0.031 0.001 0.022 0.108 1.00    514
#> betaP[3, 9]      0.014 0.013 0.001 0.010 0.047 1.04    575
#> betaP[1, 10]     0.149 0.155 0.002 0.098 0.598 1.01    504
#> betaP[2, 10]     0.024 0.023 0.001 0.017 0.087 1.02    660
#> betaP[3, 10]     0.013 0.013 0.000 0.010 0.047 1.02    489
#> betaP[1, 11]     0.267 0.264 0.006 0.163 0.911 1.00    256
#> betaP[2, 11]     0.021 0.021 0.001 0.014 0.077 1.04    611
#> betaP[3, 11]     0.012 0.011 0.000 0.008 0.043 1.02    534
#> betaP[1, 12]     0.493 0.293 0.027 0.482 0.975 1.00    692
#> betaP[2, 12]     0.036 0.036 0.001 0.025 0.130 1.06    410
#> betaP[3, 12]     0.020 0.019 0.000 0.014 0.072 1.00    503
#> betaP[1, 13]     0.171 0.166 0.006 0.121 0.634 1.00    677
#> betaP[2, 13]     0.034 0.034 0.000 0.023 0.120 1.00    371
#> betaP[3, 13]     0.020 0.019 0.001 0.015 0.072 1.05    489
#> betaP[1, 14]     0.234 0.208 0.005 0.178 0.788 1.02    506
#> betaP[2, 14]     0.051 0.051 0.001 0.035 0.184 1.00    431
#> betaP[3, 14]     0.065 0.062 0.001 0.046 0.225 1.00    617
#> betaP[1, 15]     0.473 0.296 0.019 0.471 0.978 1.01    791
#> betaP[2, 15]     0.424 0.286 0.017 0.381 0.961 1.01    366
#> betaP[3, 15]     0.484 0.296 0.011 0.493 0.975 1.00    522
#> betaPhi[1, 1]     0.207 0.095 0.059 0.191 0.426 1.02    788
#> betaPhi[2, 1]     0.741 0.065 0.614 0.744 0.865 1.00   1022
#> betaPhi[3, 1]     0.702 0.085 0.534 0.702 0.862 1.00    805
#> betaPhi[1, 2]     0.308 0.095 0.147 0.297 0.505 1.00    857
#> betaPhi[2, 2]     0.713 0.049 0.619 0.713 0.808 1.00    773
#> betaPhi[3, 2]     0.792 0.062 0.666 0.794 0.911 1.01    757
#> betaPhi[1, 3]     0.233 0.065 0.121 0.228 0.373 1.00    983
#> betaPhi[2, 3]     0.652 0.039 0.574 0.651 0.733 1.00    787
#> betaPhi[3, 3]     0.520 0.039 0.446 0.520 0.597 1.01    722
#> betaPhi[1, 4]     0.414 0.148 0.162 0.404 0.729 1.01    963
#> betaPhi[2, 4]     0.964 0.031 0.891 0.973 0.999 1.01    688
#> betaPhi[3, 4]     0.983 0.015 0.944 0.988 1.000 1.01    509
#> betaPhi[1, 5]     0.376 0.095 0.206 0.371 0.570 1.00   1121
#> betaPhi[2, 5]     0.633 0.051 0.530 0.634 0.730 1.00    799
#> betaPhi[3, 5]     0.737 0.039 0.661 0.738 0.814 1.00    801
#> betaPhi[1, 6]     0.581 0.106 0.380 0.573 0.797 1.04    865
#> betaPhi[2, 6]     0.510 0.049 0.417 0.508 0.609 1.00   1140
#> betaPhi[3, 6]     0.790 0.039 0.713 0.791 0.867 1.00    728
#> betaPhi[1, 7]     0.192 0.091 0.051 0.183 0.397 1.00    845
#> betaPhi[2, 7]     0.739 0.058 0.636 0.737 0.859 1.00    778
#> betaPhi[3, 7]     0.840 0.038 0.769 0.839 0.918 1.00    645
#> betaPhi[1, 8]     0.414 0.087 0.261 0.407 0.608 1.00    907
#> betaPhi[2, 8]     0.690 0.060 0.573 0.691 0.802 1.01    796
#> betaPhi[3, 8]     0.874 0.041 0.797 0.873 0.953 1.00    691
#> betaPhi[1, 9]     0.236 0.041 0.166 0.231 0.325 1.02    891
#> betaPhi[2, 9]     0.679 0.054 0.571 0.679 0.786 1.00    871
#> betaPhi[3, 9]     0.801 0.042 0.713 0.803 0.883 1.00    775
#> betaPhi[1, 10]    0.425 0.065 0.305 0.422 0.563 1.01    973
#> betaPhi[2, 10]    0.744 0.063 0.616 0.744 0.867 1.00    938
#> betaPhi[3, 10]    0.788 0.047 0.699 0.786 0.885 1.01    724
#> betaPhi[1, 11]    0.208 0.050 0.118 0.206 0.309 1.02    976
#> betaPhi[2, 11]    0.553 0.055 0.446 0.552 0.660 1.00    807
#> betaPhi[3, 11]    0.660 0.044 0.577 0.662 0.748 1.00    853
#> betaPhi[1, 12]    0.183 0.105 0.045 0.158 0.420 1.01    118

```



```
#> betaPhi[2, 12] 0.513 0.054 0.408 0.512 0.621 1.00 573
#> betaPhi[3, 12] 0.626 0.045 0.542 0.625 0.712 1.00 751
#> betaPhi[1, 13] 0.260 0.070 0.138 0.255 0.410 1.00 857
#> betaPhi[2, 13] 0.416 0.042 0.336 0.416 0.502 1.02 814
#> betaPhi[3, 13] 0.752 0.050 0.658 0.750 0.852 1.00 684
#> betaPhi[1, 14] 0.274 0.075 0.141 0.271 0.433 1.01 834
#> betaPhi[2, 14] 0.534 0.049 0.442 0.531 0.637 1.00 844
#> betaPhi[3, 14] 0.871 0.063 0.744 0.873 0.985 1.00 687
#> betaPhi[1, 15] 0.285 0.073 0.167 0.277 0.452 1.02 817
#> betaPhi[2, 15] 0.364 0.059 0.260 0.361 0.491 1.01 443
#> betaPhi[3, 15] 0.268 0.049 0.183 0.262 0.375 1.00 489
#kable(modSummary %>%
#      add_column(data.frame(year = rep(years[1:15], 2), dateYM = rep(occs[1:15], 2)) )

# d %>%
# summarize(unique(data.frame(dateYM, occ)))
```

## phiT\_pT\_trib

Load data 'trib' for analysis

```
#d_tt <- tar_read(target_d_trib)
eh_trib <- tar_read(target_eh_trib)

str(eh_trib$eh)
#> num [1:288, 1:12] 1 1 1 1 1 1 1 1 1 1 1 ...
#> - attr(*, "dimnames")=List of 2
#> ..$ : NULL
#> ..$ : chr [1:12] "date_2018-07" "date_2018-08" "date_2018-09" "date_2018-10" ...
kable(head(eh_trib$eh,8))
```

date_2018-07	date_2018-08	date_2018-09	date_2018-10	date_2019-07	date_2019-08	date_2019-09	date_2019-10	date_2020-07	date_2020-08	date_2020-09	date_2020-10
1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	1	1	0	0	0	0	0
1	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0

```
table(paste(eh_trib$first, eh_trib$last, sep="_"))
#>
#> 1_12 2_12 3_12 4_12 5_12 6_12 7_12 8_12
#> 60 8 12 15 100 53 20 20
```

```
### Read the model run into global memory
if (tar_exist_objects(c("tt_modelOut_trib"))) {
```

```

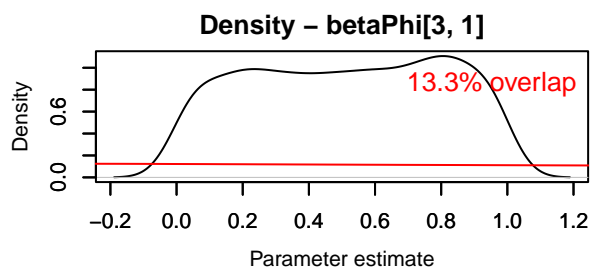
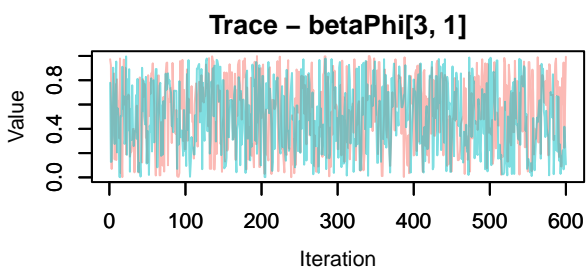
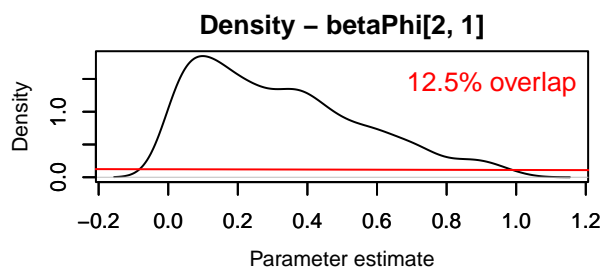
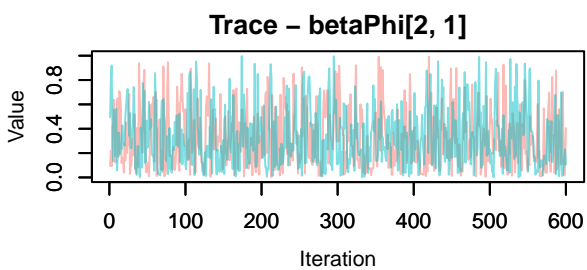
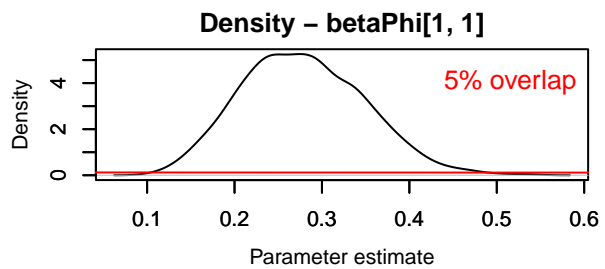
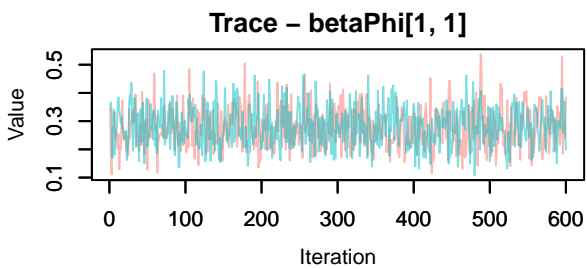
mod_tt_trib <- tar_read(tt_modelOut_trib)

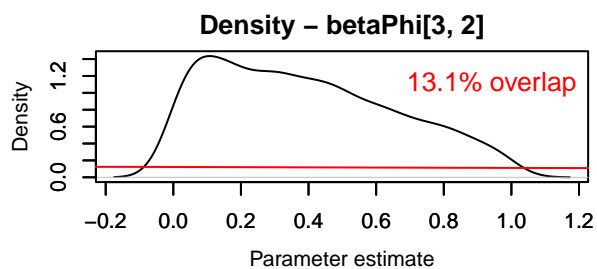
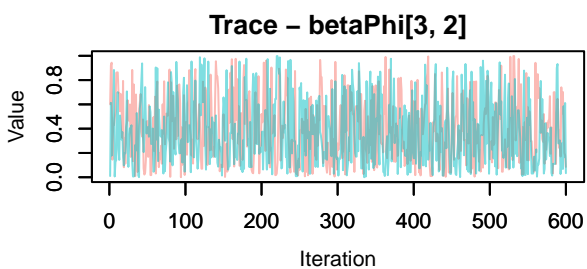
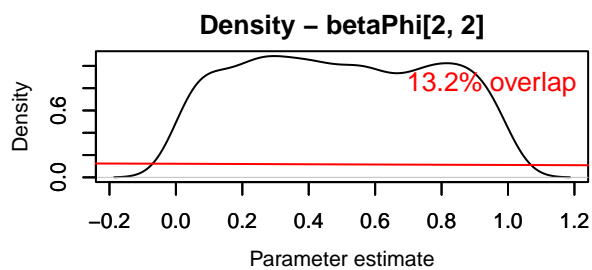
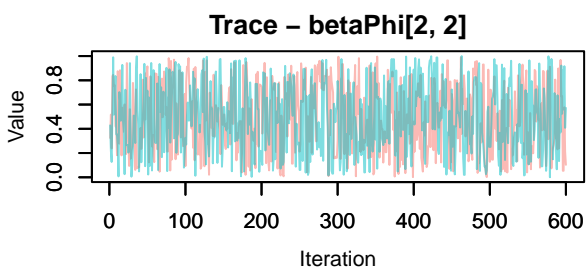
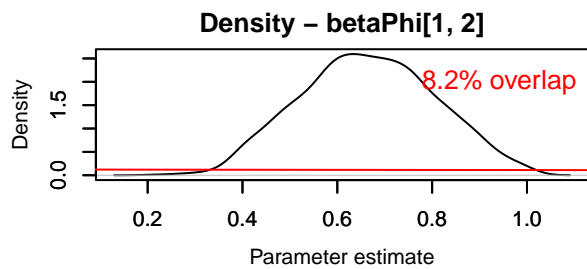
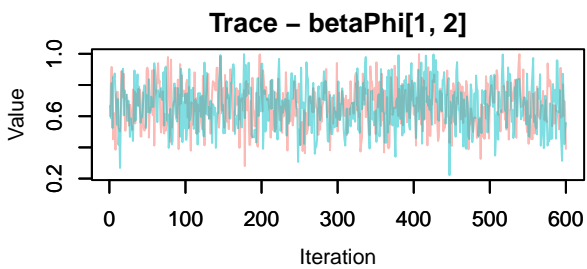
#MCMCplot(object = mod_tt_trib$mcmc, params = "betaPhi")
priors <- rnorm(tar_read(tt_runData_trib)$nIter * tar_read(tt_runData_trib)$nChains, 0, 1/sqrt(.1))
MCMCtrace(object = mod_tt_trib$mcmc,
          #ISB = FALSE,
          #exact = TRUE,
          params = c("betaPhi"),
          pdf = FALSE,
          priors = priors
          )

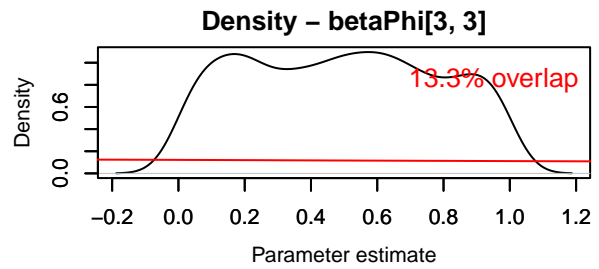
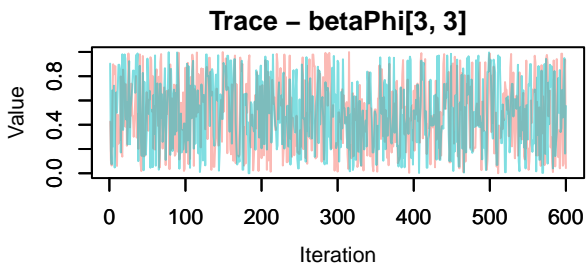
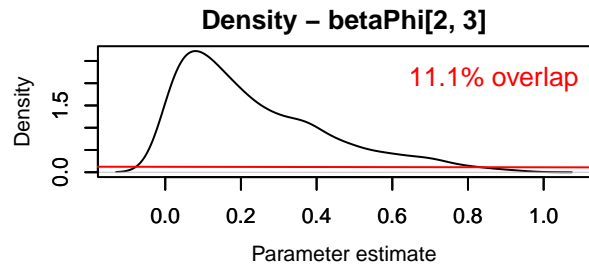
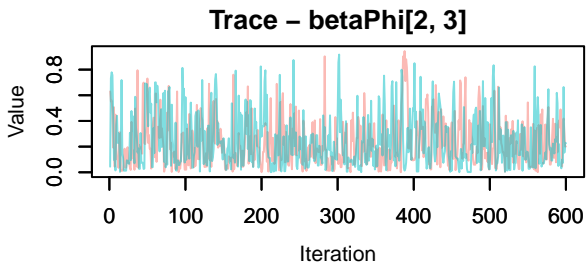
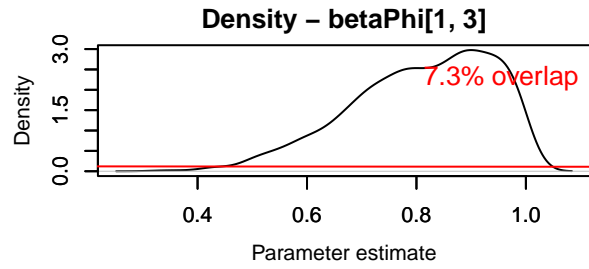
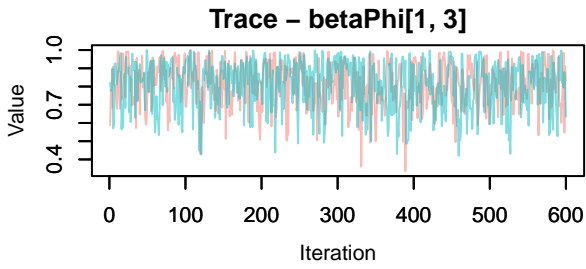
#
(modSummary_tt_trib <- MCMCsummary(object = mod_tt_trib$mcmc, round = 3))

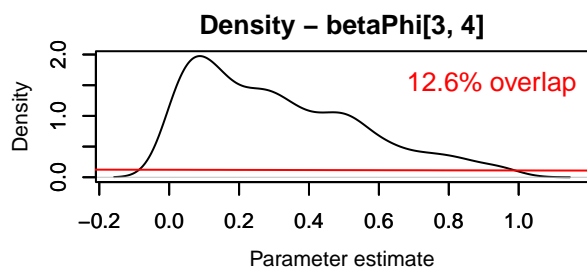
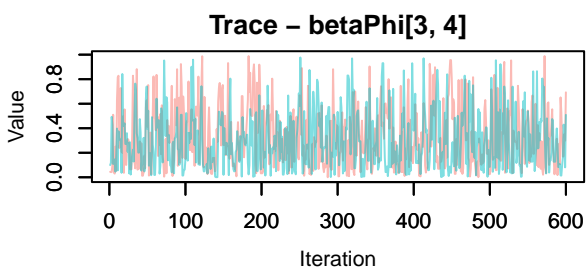
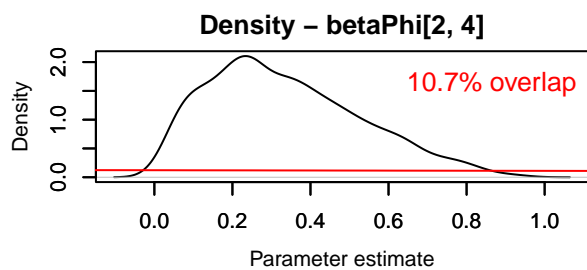
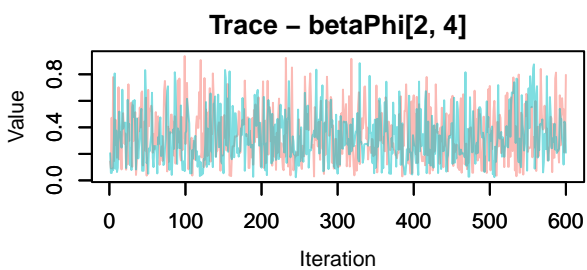
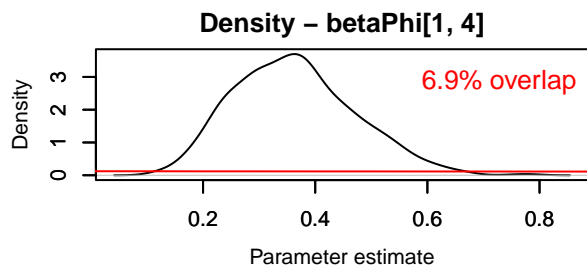
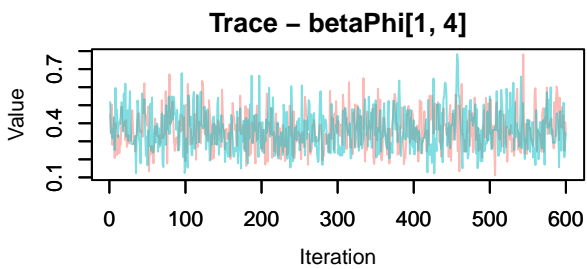
# MCMCplot(object = mod$mcmc, params = "betaPhiRiverOut")
#
# priors <- rnorm(tar_read(ttt_runData)$nIter * tar_read(ttt_runData)$nChains, 0, 1/sqrt(.1))
# MCMCtrace(object = mod$mcmc,
#           #ISB = FALSE,
#           #exact = TRUE,
#           params = c("betaPhiRiverOut"),
#           pdf = FALSE,
#           priors = priors
#           )
#
# }
#> Warning in MCMCtrace(object = mod_tt_trib$mcmc, params = c("betaPhi"), pdf =
#> FALSE, : Only one prior specified for > 1 parameter. Using a single prior for
#> all parameters.
#> Warning in MCMCtrace(object = mod_tt_trib$mcmc, params = c("betaPhi"), pdf =
#> FALSE, : Number of samples in prior is greater than number of total or specified
#> iterations (for all chains) for specified parameter. Only last 1200 iterations
#> will be used.

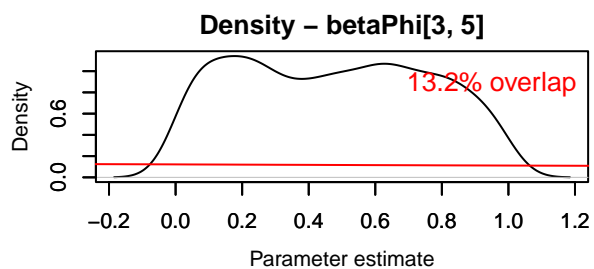
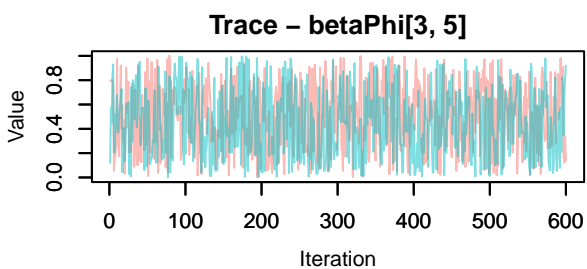
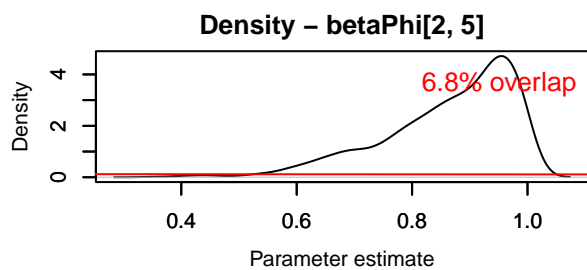
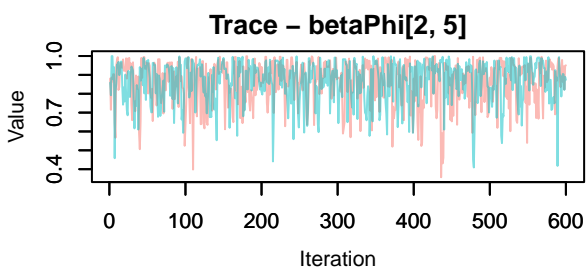
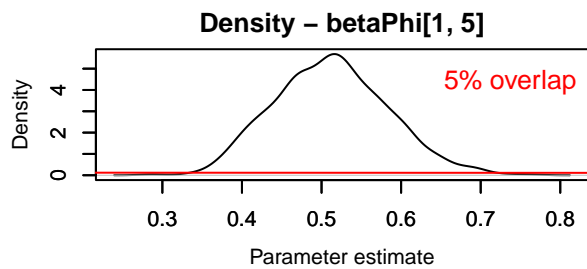
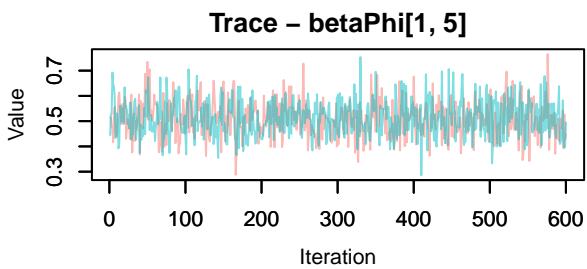
```

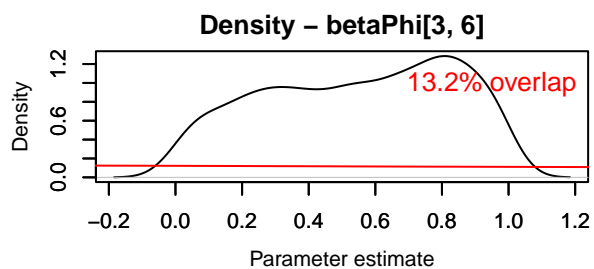
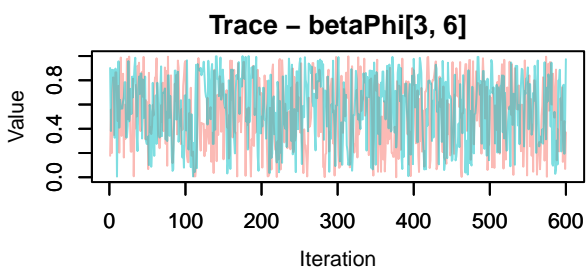
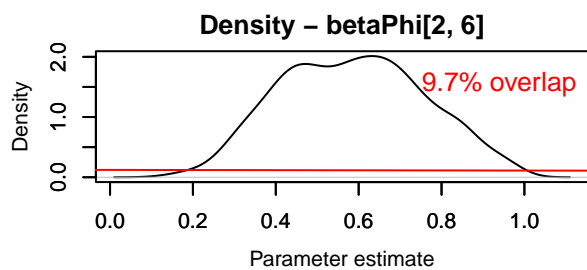
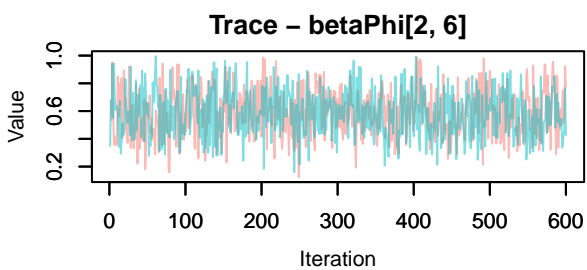
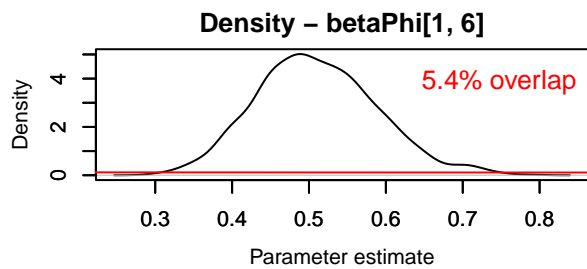
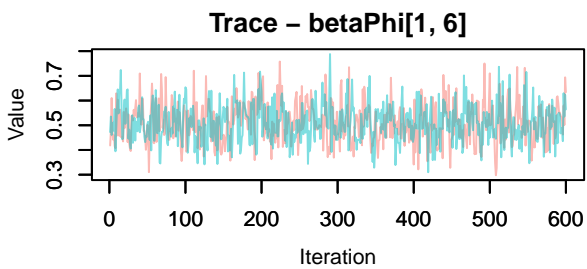




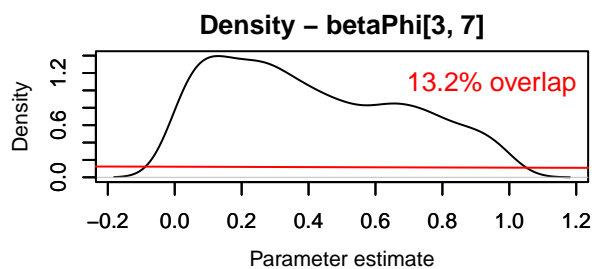
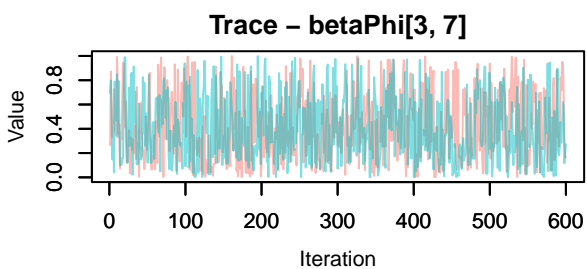
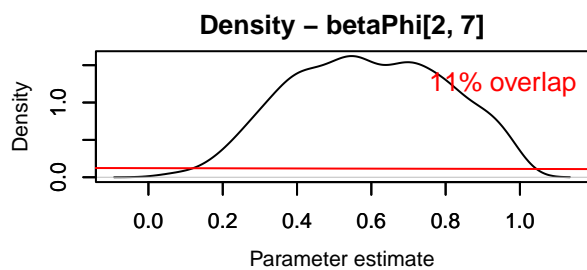
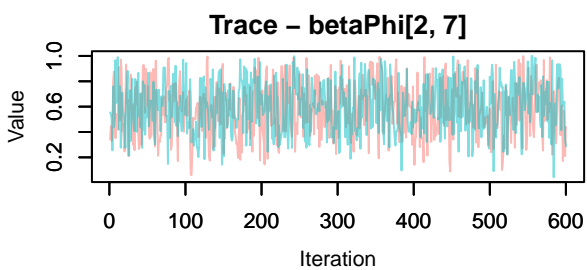
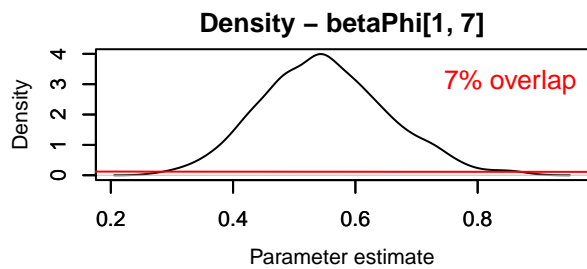
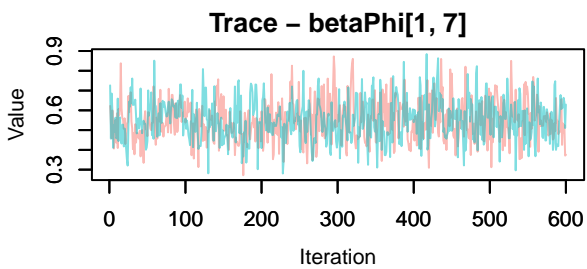


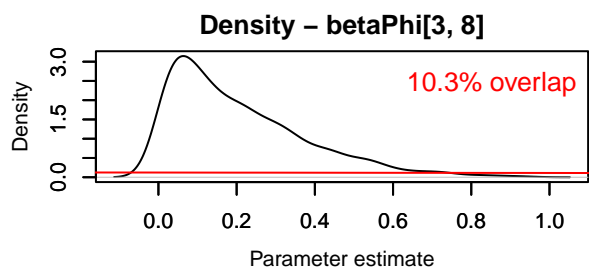
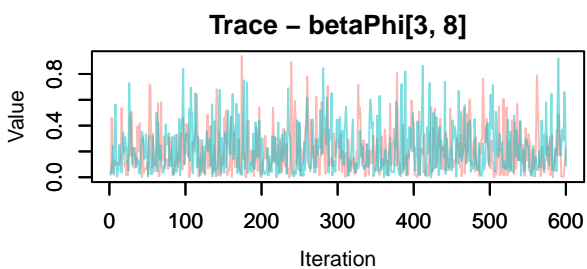
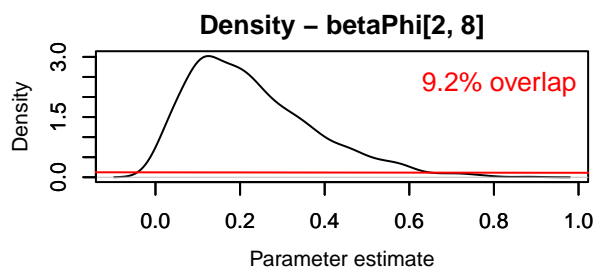
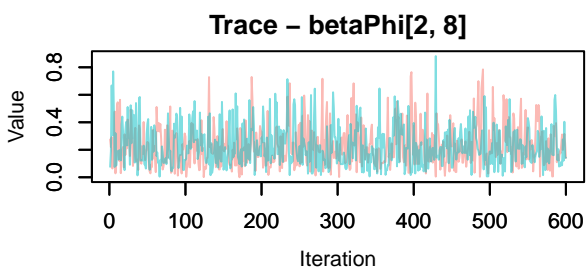
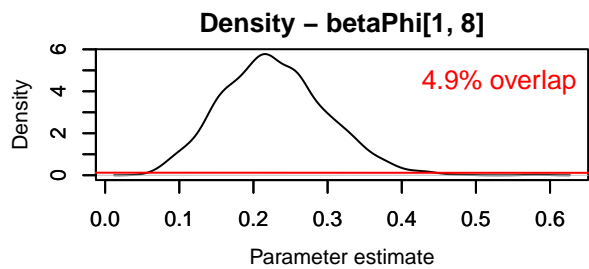
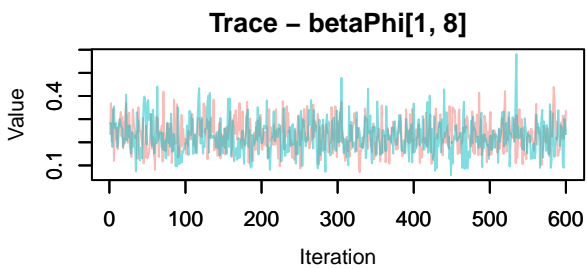


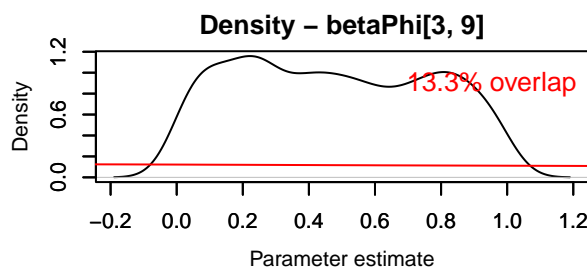
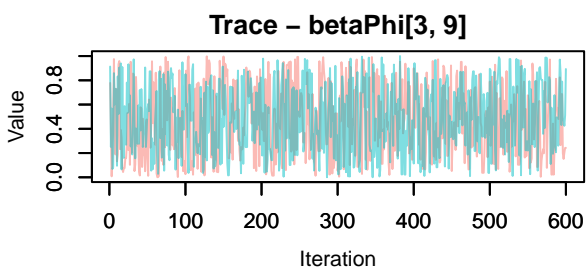
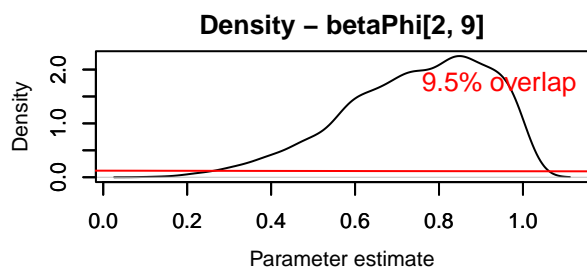
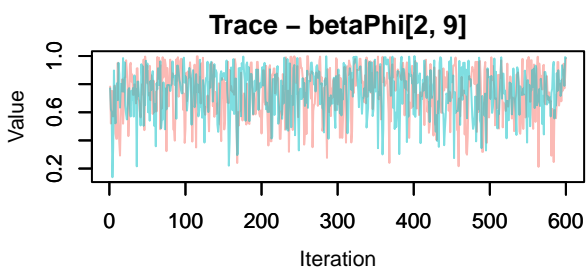
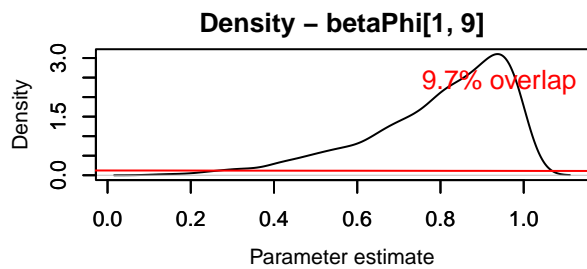
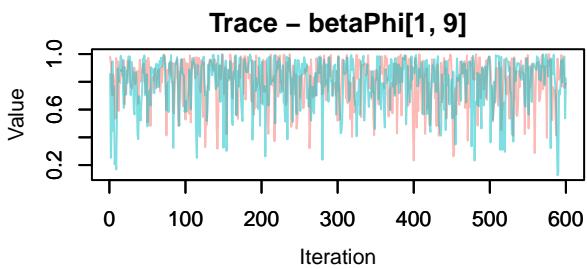


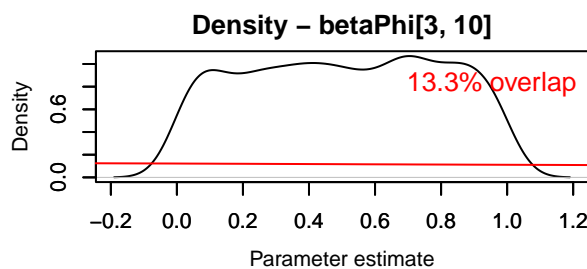
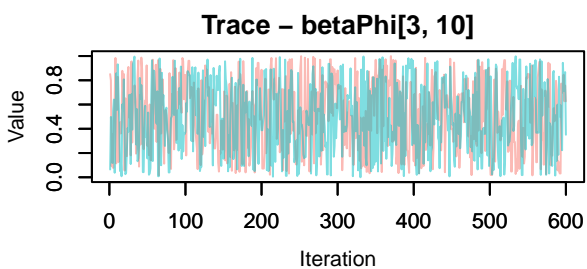
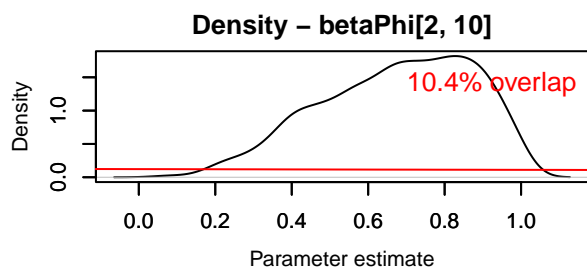
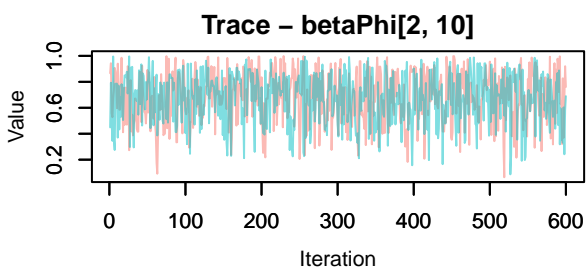
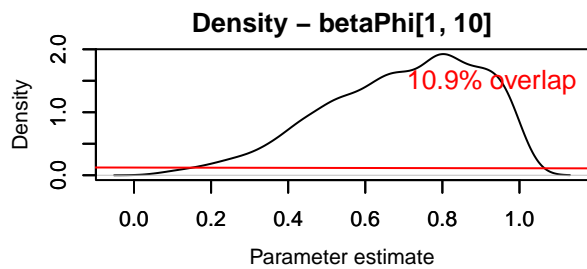
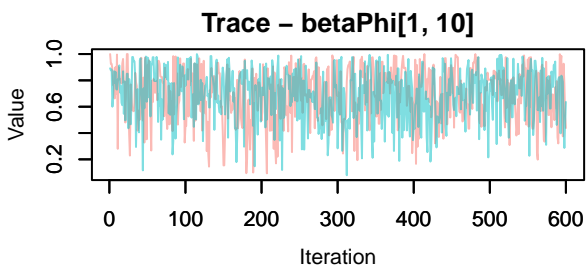


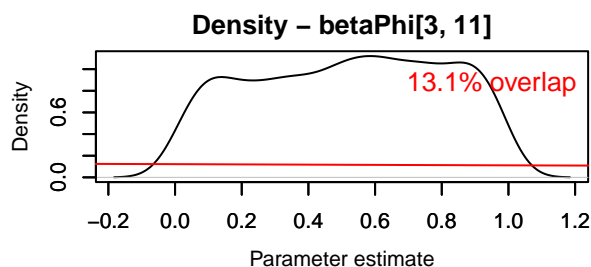
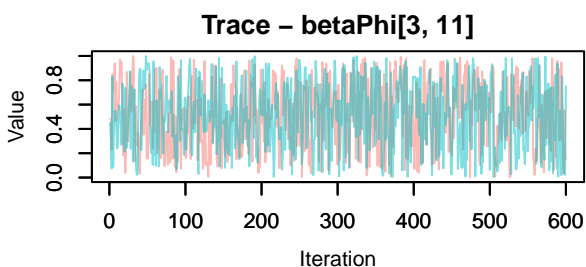
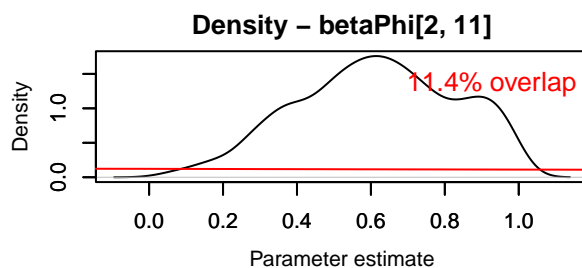
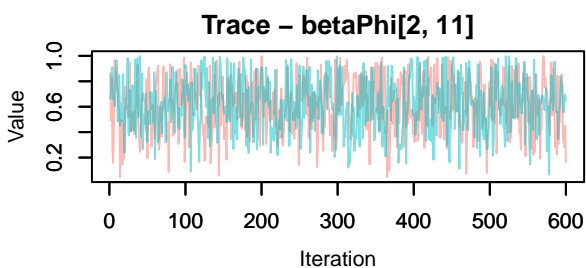
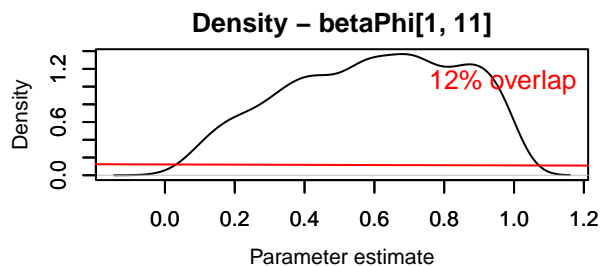
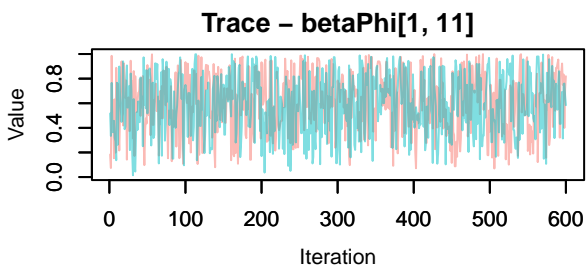












#>		mean	sd	2.5%	50%	97.5%	Rhat	n.eff
#>	betaP[1, 1]	0.140	0.134	0.003	0.100	0.494	1.01	589
#>	betaP[2, 1]	0.476	0.289	0.019	0.468	0.974	1.00	982
#>	betaP[3, 1]	0.490	0.288	0.027	0.477	0.975	1.00	927
#>	betaP[1, 2]	0.136	0.127	0.003	0.097	0.445	1.06	548
#>	betaP[2, 2]	0.488	0.292	0.026	0.503	0.971	1.03	810
#>	betaP[3, 2]	0.472	0.286	0.027	0.458	0.968	1.01	1017
#>	betaP[1, 3]	0.273	0.224	0.008	0.216	0.862	1.00	681
#>	betaP[2, 3]	0.479	0.286	0.030	0.465	0.974	1.00	973
#>	betaP[3, 3]	0.495	0.293	0.029	0.490	0.981	1.00	1055
#>	betaP[1, 4]	0.439	0.277	0.020	0.416	0.954	1.00	884
#>	betaP[2, 4]	0.436	0.277	0.018	0.401	0.951	1.00	842
#>	betaP[3, 4]	0.486	0.287	0.024	0.467	0.969	1.00	1090
#>	betaP[1, 5]	0.469	0.268	0.033	0.446	0.963	1.00	762
#>	betaP[2, 5]	0.533	0.283	0.038	0.548	0.981	1.00	936
#>	betaP[3, 5]	0.482	0.291	0.022	0.466	0.970	1.00	998
#>	betaP[1, 6]	0.294	0.222	0.011	0.247	0.811	1.00	580
#>	betaP[2, 6]	0.569	0.282	0.035	0.604	0.978	1.01	872
#>	betaP[3, 6]	0.506	0.293	0.031	0.509	0.974	1.00	955
#>	betaP[1, 7]	0.329	0.253	0.013	0.266	0.882	1.01	648
#>	betaP[2, 7]	0.429	0.277	0.025	0.403	0.959	1.00	918
#>	betaP[3, 7]	0.485	0.288	0.026	0.484	0.966	1.00	966
#>	betaP[1, 8]	0.581	0.274	0.044	0.625	0.976	1.01	1075
#>	betaP[2, 8]	0.468	0.285	0.020	0.460	0.957	1.00	832
#>	betaP[3, 8]	0.486	0.289	0.022	0.463	0.982	1.00	991
#>	betaP[1, 9]	0.556	0.287	0.040	0.588	0.984	1.01	984

```

#> betaP[2, 9]      0.515 0.284 0.028 0.524 0.977 1.00 1041
#> betaP[3, 9]      0.500 0.286 0.032 0.503 0.984 1.00 928
#> betaP[1, 10]     0.552 0.287 0.029 0.561 0.979 1.00 869
#> betaP[2, 10]     0.418 0.278 0.015 0.390 0.946 1.00 1136
#> betaP[3, 10]     0.494 0.287 0.027 0.499 0.972 1.00 1033
#> betaP[1, 11]     0.515 0.285 0.030 0.527 0.979 1.00 944
#> betaP[2, 11]     0.502 0.279 0.023 0.520 0.966 1.00 980
#> betaP[3, 11]     0.497 0.287 0.027 0.497 0.975 1.00 952
#> betaPhi[1, 1]    0.278 0.070 0.155 0.275 0.418 1.00 904
#> betaPhi[2, 1]    0.326 0.240 0.013 0.283 0.883 1.00 740
#> betaPhi[3, 1]    0.511 0.291 0.026 0.515 0.972 1.02 818
#> betaPhi[1, 2]    0.665 0.141 0.406 0.665 0.938 1.01 786
#> betaPhi[2, 2]    0.494 0.285 0.026 0.485 0.967 1.00 896
#> betaPhi[3, 2]    0.390 0.268 0.014 0.360 0.942 1.01 874
#> betaPhi[1, 3]    0.809 0.129 0.528 0.825 0.990 1.01 817
#> betaPhi[2, 3]    0.239 0.199 0.006 0.185 0.722 1.00 512
#> betaPhi[3, 3]    0.491 0.288 0.026 0.494 0.978 1.00 978
#> betaPhi[1, 4]    0.363 0.108 0.178 0.358 0.592 1.00 847
#> betaPhi[2, 4]    0.337 0.194 0.050 0.312 0.772 1.00 990
#> betaPhi[3, 4]    0.317 0.241 0.011 0.270 0.875 1.00 702
#> betaPhi[1, 5]    0.512 0.071 0.384 0.511 0.658 1.01 1251
#> betaPhi[2, 5]    0.862 0.114 0.594 0.890 0.995 1.00 711
#> betaPhi[3, 5]    0.473 0.285 0.021 0.478 0.969 1.02 1053
#> betaPhi[1, 6]    0.510 0.078 0.370 0.505 0.682 1.00 645
#> betaPhi[2, 6]    0.586 0.173 0.283 0.584 0.918 1.00 779
#> betaPhi[3, 6]    0.546 0.282 0.037 0.566 0.978 1.00 664
#> betaPhi[1, 7]    0.550 0.104 0.362 0.544 0.761 1.00 662
#> betaPhi[2, 7]    0.593 0.207 0.211 0.590 0.963 1.00 848
#> betaPhi[3, 7]    0.408 0.278 0.013 0.361 0.952 1.01 895
#> betaPhi[1, 8]    0.228 0.070 0.103 0.223 0.375 1.00 872
#> betaPhi[2, 8]    0.229 0.150 0.022 0.200 0.583 1.01 833
#> betaPhi[3, 8]    0.207 0.175 0.006 0.162 0.648 1.01 888
#> betaPhi[1, 9]    0.794 0.170 0.376 0.834 0.993 1.00 805
#> betaPhi[2, 9]    0.743 0.172 0.358 0.764 0.991 1.00 797
#> betaPhi[3, 9]    0.474 0.290 0.022 0.458 0.971 1.00 968
#> betaPhi[1, 10]   0.693 0.200 0.255 0.715 0.981 1.00 778
#> betaPhi[2, 10]   0.673 0.197 0.260 0.694 0.984 1.00 869
#> betaPhi[3, 10]   0.503 0.291 0.023 0.506 0.973 1.00 823
#> betaPhi[1, 11]   0.594 0.245 0.115 0.610 0.978 1.00 922
#> betaPhi[2, 11]   0.615 0.214 0.184 0.619 0.973 1.00 921
#> betaPhi[3, 11]   0.517 0.283 0.029 0.530 0.967 1.00 924
#> psi[1, 1, 1]     0.982 0.041 0.856 0.998 1.000 1.00 1200
#> psi[2, 1, 1]     0.102 0.218 0.000 0.001 0.849 1.00 960
#> psi[3, 1, 1]     0.110 0.226 0.000 0.001 0.879 1.01 697
#> psi[1, 2, 1]     0.009 0.031 0.000 0.000 0.089 1.01 1200
#> psi[2, 2, 1]     0.794 0.293 0.045 0.960 1.000 1.00 800
#> psi[3, 2, 1]     0.107 0.222 0.000 0.002 0.840 1.00 994
#> psi[1, 3, 1]     0.009 0.027 0.000 0.000 0.100 1.04 1200
#> psi[2, 3, 1]     0.104 0.222 0.000 0.001 0.850 1.00 839
#> psi[3, 3, 1]     0.783 0.293 0.041 0.945 1.000 1.00 841
#> psi[1, 1, 2]     0.973 0.077 0.817 0.998 1.000 1.24 771
#> psi[2, 1, 2]     0.095 0.210 0.000 0.001 0.814 1.00 819
#> psi[3, 1, 2]     0.093 0.204 0.000 0.001 0.798 1.01 882
#> psi[1, 2, 2]     0.016 0.071 0.000 0.000 0.129 1.26 728

```

```

#> psi[2, 2, 2] 0.803 0.281 0.054 0.957 1.000 1.00 676
#> psi[3, 2, 2] 0.110 0.226 0.000 0.001 0.864 1.00 886
#> psi[1, 3, 2] 0.011 0.031 0.000 0.000 0.107 1.03 1112
#> psi[2, 3, 2] 0.102 0.210 0.000 0.002 0.852 1.00 707
#> psi[3, 3, 2] 0.797 0.284 0.064 0.953 1.000 1.01 908
#> psi[1, 1, 3] 0.982 0.038 0.868 0.998 1.000 1.01 1096
#> psi[2, 1, 3] 0.099 0.214 0.000 0.001 0.788 1.01 939
#> psi[3, 1, 3] 0.088 0.197 0.000 0.001 0.780 1.00 982
#> psi[1, 2, 3] 0.010 0.029 0.000 0.000 0.095 1.11 1057
#> psi[2, 2, 3] 0.803 0.282 0.074 0.960 1.000 1.00 906
#> psi[3, 2, 3] 0.111 0.232 0.000 0.001 0.865 1.01 967
#> psi[1, 3, 3] 0.008 0.026 0.000 0.000 0.079 1.08 1200
#> psi[2, 3, 3] 0.098 0.207 0.000 0.001 0.785 1.00 880
#> psi[3, 3, 3] 0.800 0.287 0.067 0.962 1.000 1.01 875
#> psi[1, 1, 4] 0.371 0.182 0.076 0.356 0.730 1.01 1102
#> psi[2, 1, 4] 0.055 0.140 0.000 0.000 0.544 1.00 1036
#> psi[3, 1, 4] 0.105 0.214 0.000 0.001 0.797 1.00 886
#> psi[1, 2, 4] 0.615 0.183 0.260 0.630 0.921 1.01 1094
#> psi[2, 2, 4] 0.891 0.188 0.333 0.986 1.000 1.00 1107
#> psi[3, 2, 4] 0.096 0.207 0.000 0.001 0.823 1.04 987
#> psi[1, 3, 4] 0.013 0.038 0.000 0.000 0.130 1.00 1037
#> psi[2, 3, 4] 0.055 0.133 0.000 0.000 0.542 1.01 1079
#> psi[3, 3, 4] 0.799 0.280 0.075 0.952 1.000 1.02 899
#> psi[1, 1, 5] 0.995 0.012 0.961 1.000 1.000 1.02 1002
#> psi[2, 1, 5] 0.186 0.135 0.010 0.163 0.498 1.00 1069
#> psi[3, 1, 5] 0.102 0.212 0.000 0.002 0.806 1.00 1027
#> psi[1, 2, 5] 0.002 0.007 0.000 0.000 0.020 1.02 1115
#> psi[2, 2, 5] 0.762 0.152 0.414 0.784 0.982 1.01 682
#> psi[3, 2, 5] 0.110 0.218 0.000 0.001 0.808 1.02 942
#> psi[1, 3, 5] 0.003 0.010 0.000 0.000 0.029 1.02 1200
#> psi[2, 3, 5] 0.052 0.086 0.000 0.007 0.295 1.06 454
#> psi[3, 3, 5] 0.787 0.286 0.052 0.944 1.000 1.01 911
#> psi[1, 1, 6] 0.965 0.031 0.877 0.973 0.998 1.01 1096
#> psi[2, 1, 6] 0.023 0.061 0.000 0.000 0.224 1.01 900
#> psi[3, 1, 6] 0.079 0.179 0.000 0.001 0.691 1.00 899
#> psi[1, 2, 6] 0.032 0.030 0.001 0.023 0.118 1.01 1092
#> psi[2, 2, 6] 0.864 0.147 0.503 0.913 1.000 1.00 616
#> psi[3, 2, 6] 0.072 0.176 0.000 0.001 0.721 1.00 1022
#> psi[1, 3, 6] 0.003 0.010 0.000 0.000 0.033 1.00 759
#> psi[2, 3, 6] 0.113 0.139 0.000 0.057 0.467 1.01 526
#> psi[3, 3, 6] 0.849 0.240 0.129 0.974 1.000 1.00 812
#> psi[1, 1, 7] 0.992 0.019 0.934 0.999 1.000 1.00 1200
#> psi[2, 1, 7] 0.051 0.122 0.000 0.000 0.474 1.00 892
#> psi[3, 1, 7] 0.124 0.235 0.000 0.003 0.863 1.00 876
#> psi[1, 2, 7] 0.005 0.016 0.000 0.000 0.040 1.00 1200
#> psi[2, 2, 7] 0.923 0.137 0.516 0.988 1.000 1.00 864
#> psi[3, 2, 7] 0.097 0.211 0.000 0.001 0.786 1.00 719
#> psi[1, 3, 7] 0.004 0.011 0.000 0.000 0.034 1.00 1022
#> psi[2, 3, 7] 0.026 0.066 0.000 0.000 0.239 1.00 1041
#> psi[3, 3, 7] 0.779 0.292 0.062 0.939 1.000 1.00 826
#> psi[1, 1, 8] 0.488 0.158 0.195 0.481 0.780 1.00 1282
#> psi[2, 1, 8] 0.052 0.133 0.000 0.000 0.530 1.00 1200
#> psi[3, 1, 8] 0.100 0.216 0.000 0.001 0.820 1.02 856
#> psi[1, 2, 8] 0.499 0.158 0.210 0.504 0.799 1.00 1286

```

```

#> psi[2, 2, 8]    0.885 0.199 0.265 0.985 1.000 1.00    1111
#> psi[3, 2, 8]    0.112 0.224 0.000 0.002 0.842 1.00     901
#> psi[1, 3, 8]    0.012 0.036 0.000 0.000 0.120 1.00    1132
#> psi[2, 3, 8]    0.063 0.155 0.000 0.000 0.588 1.00    1029
#> psi[3, 3, 8]    0.789 0.293 0.069 0.958 1.000 1.00    1039
#> psi[1, 1, 9]    0.838 0.188 0.361 0.913 1.000 1.00     587
#> psi[2, 1, 9]    0.019 0.055 0.000 0.000 0.168 1.01     942
#> psi[3, 1, 9]    0.098 0.207 0.000 0.001 0.816 1.00     836
#> psi[1, 2, 9]    0.138 0.180 0.000 0.048 0.612 1.00     506
#> psi[2, 2, 9]    0.964 0.073 0.755 0.996 1.000 1.00     869
#> psi[3, 2, 9]    0.108 0.220 0.000 0.001 0.832 1.09     860
#> psi[1, 3, 9]    0.025 0.074 0.000 0.000 0.272 1.00     842
#> psi[2, 3, 9]    0.017 0.047 0.000 0.000 0.170 1.06    1120
#> psi[3, 3, 9]    0.794 0.282 0.078 0.948 1.000 1.03     742
#> psi[1, 1, 10]   0.810 0.207 0.298 0.886 1.000 1.00     567
#> psi[2, 1, 10]   0.024 0.070 0.000 0.000 0.242 1.00     662
#> psi[3, 1, 10]   0.096 0.206 0.000 0.001 0.804 1.00    1029
#> psi[1, 2, 10]   0.160 0.196 0.000 0.067 0.649 1.00     622
#> psi[2, 2, 10]   0.953 0.092 0.687 0.994 1.000 1.01    1018
#> psi[3, 2, 10]   0.110 0.225 0.000 0.002 0.847 1.00     767
#> psi[1, 3, 10]   0.030 0.081 0.000 0.000 0.319 1.00    1133
#> psi[2, 3, 10]   0.023 0.063 0.000 0.000 0.231 1.05    1065
#> psi[3, 3, 10]   0.794 0.281 0.050 0.943 1.000 1.00     725
#> psi[1, 1, 11]   0.904 0.167 0.382 0.987 1.000 1.00    1042
#> psi[2, 1, 11]   0.035 0.091 0.000 0.000 0.316 1.01    1060
#> psi[3, 1, 11]   0.098 0.215 0.000 0.001 0.843 1.01     732
#> psi[1, 2, 11]   0.044 0.116 0.000 0.000 0.443 1.00    1200
#> psi[2, 2, 11]   0.934 0.123 0.550 0.991 1.000 1.03     922
#> psi[3, 2, 11]   0.122 0.238 0.000 0.002 0.876 1.00     778
#> psi[1, 3, 11]   0.052 0.127 0.000 0.000 0.463 1.00    1046
#> psi[2, 3, 11]   0.031 0.085 0.000 0.000 0.305 1.08     860
#> psi[3, 3, 11]   0.780 0.298 0.045 0.945 1.000 1.00     877
#kable(modSummary %>%
#       add_column(data.frame(year = rep(years[1:15], 2), dateYM = rep(occs[1:15], 2)) )

# d %>%
# summarize(unique(data.frame(dateYM, occ)))

```

## phiT\_pT\_psiT

Main and trib together

```

### Read the model run into global memory
if (tar_exist_objects(c("ttt_modelOut"))) {
  mod_ttt <- tar_read(ttt_modelOut)

  #MCMCplot(object = mod$mcmc)

  modSummary_ttt <- MCMCsummary(object = mod_ttt$mcmc, round = 3)
}
#kable(modSummary %>%
#       add_column(data.frame(year = rep(years[1:15], 2), dateYM = rep(occs[1:15], 2)) )

```



```

# d %>%
# summarize(unique(data.frame(dateYM, occ)))

# nS <- tar_read(ttt_nStates)
# nT <- tar_read(ttt_myConstants)$T
#
# modSummaryPhi_ttt <- modSummary_ttt %>%
#   filter(substr(row.names(modSummary), 1, 10) == "betaPhiOut") %>%
#   add_column(data.frame(state = 1:nS, dateYM = rep(1:(nT - 1), each = nS))) %>%
#   mutate(mainTrib = ifelse(state < 4, "Main", "Trib"),
#          size = paste0("Size", (state - 1) %% 3 + 1))
#
# ggplot(modSummaryPhi_ttt, aes(dateYM, mean)) +
#   geom_point() +
#   geom_line() +
#   facet_grid(mainTrib ~ size)
#
#
# # modSummaryYears <- modSummary %>%
# #   filter(substr(row.names(modSummary), 1, 3) == "betaPhiout") %>%
# #   add_column(data.frame(year = years[1:15], dateYM = occs[1:15], occ = 1:15)) %>%
# #   group_by(year) %>%
# #   mutate(maxSampPerYear = occ == max(occ))
# #
# # kable(
# #   modSummaryYears %>%
# #   group_by(year) %>%
# #   filter(!maxSampPerYear) %>%
# #   summarize(phiProd = prod(mean),
# #             dateRange = range(dateYM)) %>%
# #   as.data.frame()
# # )
# MCMCplot(object = mod$mcmc, params = "betaPhiRiverOut")
#
# priors <- rnorm(tar_read(ttt_runData)$nIter * tar_read(ttt_runData)$nChains, 0, 1/sqrt(.1))
# MCMCtrace(object = mod$mcmc,
#           #ISB = FALSE,
#           #exact = TRUE,
#           params = c("betaPhiRiverOut"),
#           pdf = FALSE,
#           priors = priors
#           )
#
# MCMCtrace(object = mod$mcmc,
#           #ISB = FALSE,
#           #exact = TRUE,
#           params = c("betaPhiOut"),
#           pdf = FALSE,
#           priors = priors
#           )

```

```
#create data frame for summarizing p results
```

```
# modSummaryYearsP <- modSummary %>%  
#   filter(substr(row.names(modSummary), 1, 2) == "p[") %>%  
#   add_column(data.frame(year = years[1:15], dateYM = occs[1:15], occ = 1:15)) %>%  
#   group_by(year) %>%  
#   mutate(maxSampPerYear = occ == max(occ))  
#  
# kable(  
#   modSummaryYearsP %>%  
#   group_by(year) %>%  
#   summarize(pMean = mean(mean),  
#             dateRange = range(dateYM))  
# )
```

```
# # modSummaryYearsP <- modSummary %>%  
# #   filter(substr(row.names(modSummary), 1, 2) == "p[") %>%  
# #   add_column(data.frame(year = years[1:15], dateYM = occs[1:15], occ = 1:15)) %>%  
# #   group_by(year) %>%  
# #   mutate(maxSampPerYear = occ == max(occ))  
#  
# modSummaryPsi_ttt <- modSummary_ttt %>%  
#   filter(substr(row.names(modSummary_ttt), 1, 3) == "psi") %>%  
#   add_column(data.frame(state = 1:nS, state2 = rep(1:nS, each = nS), dateYM = rep(1:(nT - 1), each = nS)))  
#   mutate(mainTrib = ifelse(state < 4, "Main", "Trib"),  
#          size = paste0("Size", (state - 1) %% 3 + 1))  
#  
# ggplot(modSummaryPsi_ttt, aes(dateYM, mean, color = factor(state2))) +  
#   geom_point() +  
#   facet_grid(mainTrib ~ size)  
#
```