Delaware River PIT tag data analysis

Ben Letcher

2022-09-23

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.

'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 and has some other helpful functions.

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#  tar_watch(seconds = 10, outdated = FALSE, targets_only = TRUE)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

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


#str(d)
```

Load raw data for exploration
Without a postscript ('_main' or '_trib'), 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   97    0    0    0    0
#>   Roods Creek                 0    0  161    0    0    0
#>   Sands Creek                 0    0    0  140    0    0
#>   Shehawken Creek             0    0    0    0   91    0
#>   West Br Delaware River      0    0    0    0    0 7617
```

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	128
2018-06-12	99
2018-06-13	88
2018-07-16	212
2018-07-17	176
2018-07-18	141
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	128
2019-07-15	211
2019-07-16	311
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	145
2020-07-20	249
2020-07-21	29
2020-08-10	87
2020-08-11	41
2020-08-17	145
2020-08-20	110
2020-09-10	187
2020-09-14	251
2020-09-15	47
2020-10-13	368
2020-10-14	55
2020-10-15	132
2021-04-05	210
2021-04-08	67
2021-05-03	130
2021-05-10	87
2021-06-07	119
2021-06-08	171
2021-07-12	36
2021-07-15	117
2021-08-09	142
2021-08-12	137
2021-08-16	143
2021-08-17	40
2021-09-07	114
2021-09-09	108
2021-09-20	53
2021-09-22	11
2021-10-18	114
2021-10-19	121
2021-10-21	77

```
#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	7402
rainbow trout	730

```
### Number of unique tags
```

```
length(unique(d$tag))
#> [1] 5915
```

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	315
2018-07	529
2018-08	11
2018-09	110
2018-10	345
2019-04	369
2019-05	261
2019-06	297
2019-07	547
2019-08	505
2019-09	456
2019-10	367
2020-07	423
2020-08	383
2020-09	485
2020-10	555
2021-04	277
2021-05	217
2021-06	290
2021-07	153
2021-08	462
2021-09	286
2021-10	312

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   10 
#> 4622  773  297  126   53   24   13    7    2    1 

### Number of times individual fish were observed by river
(table(tagN$Water, tagN$n))
#>                          
#>                              1    2    3    4    5    6    7    8    9   10
#>   Balls Creek               30    4    1    0    0    0    0    0    0    0
#>   Cold Spring Creek         39    9    4    2    1    0    1    1    0    0
#>   Roods Creek               78   21    6    4    0    0    1    0    0    0
#>   Sands Creek               97   16    1    2    0    0    0    0    0    0
#>   Shehawken Creek           45   10    6    2    0    0    0    0    0    0
#>   West Br Delaware River 4333  713  279  116   52   24   11    6    2    1
```

Grouped by state

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

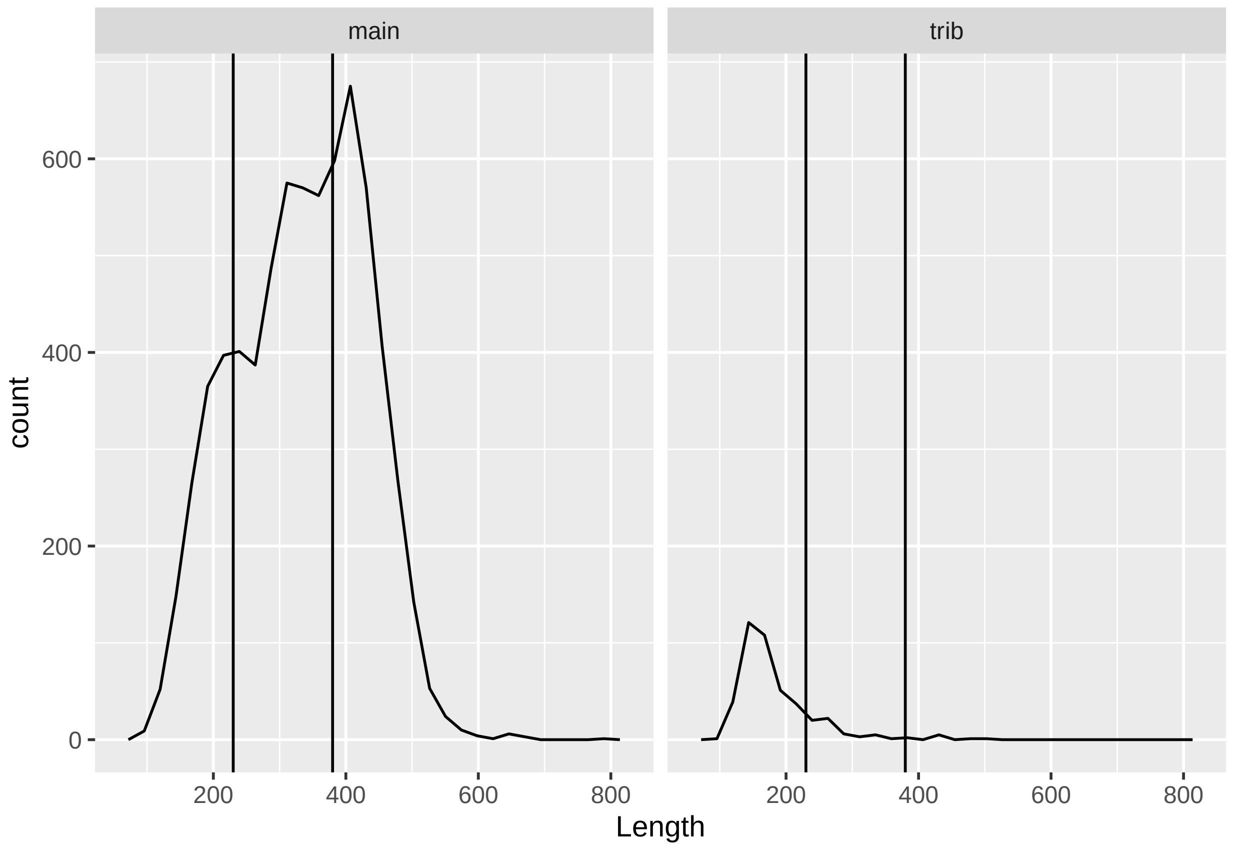
```
tagN_s <- d %>%
  group_by(tag, state) %>%
  summarize(n = n()) %>%
  filter(tag != "") %>%
  arrange(desc(n))
#> `summarise()` has grouped output by 'tag'. You can override using the `.groups`
#> argument.

### Number of times individual fish were observed
table(tagN_s$n)
#> 
#>    1    2    3    4    5    6    7    8    9   10 
#> 5043  820  242   82   40   18    6    4    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    8    9   10
#>   1 1161   79    5    0    0    0    0    0    0    0
#>   2 2214  405  109   34    9    3    2    0    0    0
#>   3 1367  270  106   43   30   14    4    4    2    1
#>   4  264   59   15    5    1    1    0    0    0    0
#>   5   29    7    7    0    0    0    0    0    0    0
#>   6    8    0    0    0    0    0    0    0    0    0
```

Grouped by main/trib   This is what is used for the primary analysis

```
tagN_mt <- d %>%
  group_by(tag, mainTrib, species) %>%
  summarize(n = n()) %>%
  filter(tag != "") %>%
  arrange(desc(n))
#> `summarise()` has grouped output by 'tag', 'mainTrib'. 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   10 
#> 4637  769  296  125   53   24   13    7    2    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   10
#>   main 4344  711  278  115   52   24   11    6    2    1
#>   trib  293   58   18   10    1    0    2    1    0    0

(table(tagN_mt$species))
#> 
#>   brook trout   brown trout rainbow trout 
#>            11          5258           656
(table(tagN_mt$mainTrib, tagN_mt$species))
#>       
#>        brook trout brown trout rainbow trout
#>   main           0        4970           572
#>   trib          11         288            84
```

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$eh)
#>  num [1:5062, 1:24] 1 1 1 1 1 1 1 1 1 1 ...
#>  - attr(*, "dimnames")=List of 2
#>   ..$ : NULL
#>   ..$ : chr [1:24] "date_2018-05" "date_2018-06" "date_2018-07" "date_2018-08" ...
kable(head(eh$eh,8))
```

date_2018-05	date_2018-06	date_2018-07	date_2018-08	date_2018-09	date_2018-10	date_2019-04	date_2019-05	date_2019-06	date_2019-07	date_2019-08	date_2019-09	date_2019-10	date_2020-07	date_2020-08	date_2020-09	date_2020-10	date_2021-04	date_2021-05	date_2021-06	date_2021-07	date_2021-08	date_2021-09	date_2021-10
1	0	0	0	0	0	0	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	0	0	0	0	0	0

date_2018-05	date_2018-06	date_2018-07	date_2018-08	date_2018-09	date_2018-10	date_2019-04	date_2019-05	date_2019-06	date_2019-07	date_2019-08	date_2019-09	date_2019-10	date_2020-07	date_2020-08	date_2020-09	date_2020-10	date_2021-04	date_2021-05	date_2021-06	date_2021-07	date_2021-08	date_2021-09	date_2021-10
1	1	0	0	0	0	0	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	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	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	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	0	0	0	0	0	0

```r
table(paste(eh$first, eh$last, sep="_"))
#> 
#>  1_24 10_24 11_24 12_24 13_24 14_24 15_24 16_24 17_24 18_24 19_24  2_24 20_24 
#>   173   359   295   264   257   296   244   291   382   185   120   263   181 
#> 21_24 22_24 23_24  3_24  4_24  5_24  6_24  7_24  8_24  9_24 
#>    82   288   157   373     8    62   231   233   130   188
```

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

```r
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      11222 
#>        964         66          4          1          6          3          1 
#>      11223         12        122       1222    1222222     122233    1223333 
#>          1         44         21          5          1          1          1 
#>        123         13      13233          2         22        222       2221 
#>          1          4          1       1645        286         81          1 
#>       2222      22222     222222    2222222     222223     222233       2223 
#>         33          8          2          2          1          1          7 
#>      22233     222333        223       2232       2233      22333     223333 
#>          7          2         35          1         15          5          2 
#>    2233333   22333333         23        233       2333      23333     233333 
#>          3          2         74         39         11          5          3 
#>       2533          3         32         33        333       3333      33333 
#>          1       1022          2        189         83         35         24 
#>     333333    3333333   33333333  333333333 3333333333          4         42 
#>         12          4          4          2          1        182          1 
#>         44        444       4444   44444455    4444455      44445    4444555 
#>         47         12          3          1          1          1          1 
#>       4445       4455         45        455       4555          5       5455 
#>          1          2          2          1          3         17          1 
#>         55        555        556          6         63 
#>          1          2          1          6          1
```

Summary info for years and occasions

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

months <- colnames(eh$eh) %>%
  substr(11,12) %>%
  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. 'psi' = probability of transitioning from one state to another. Here, states are size bins.

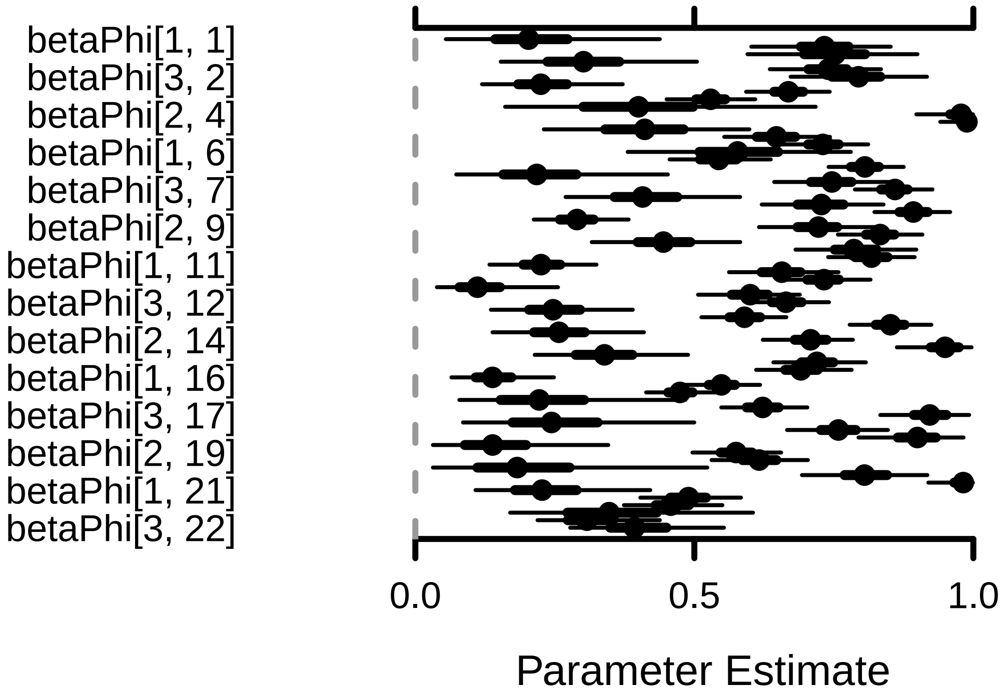
phiT__pT__psiT__main

Load data 'main' for analysis

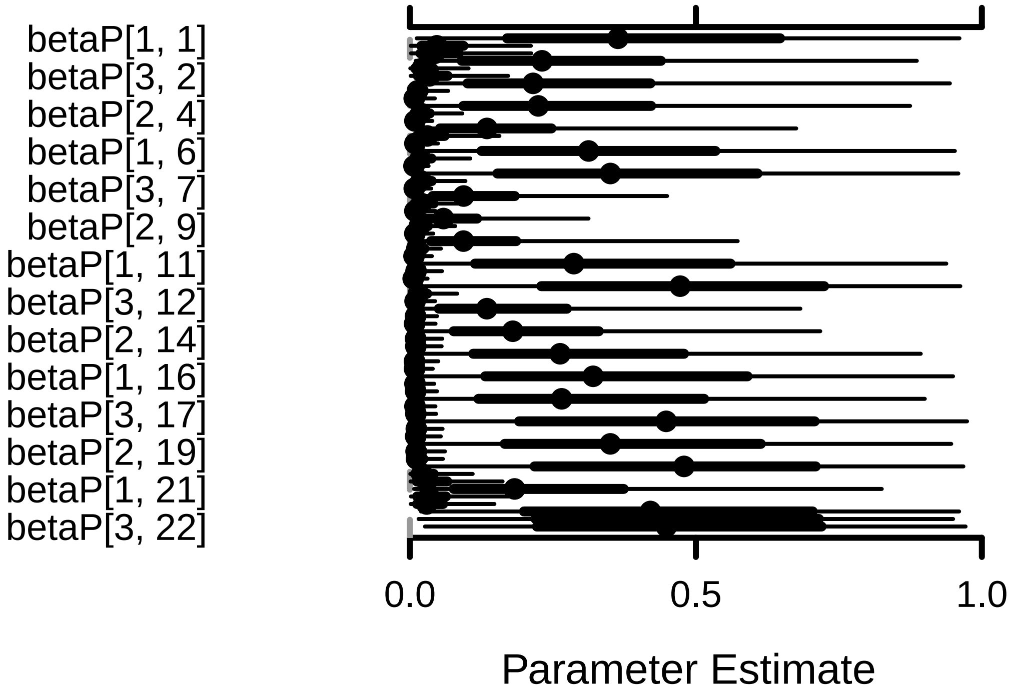
```
#d_tt <- tar_read(target_d_trib)
eh_main <- tar_read(target_eh_main)

str(eh_main$eh)
#>  num [1:4777, 1:23] 1 1 1 1 1 1 1 1 1 1 ...
#>  - attr(*, "dimnames")=List of 2
#>   ..$ : NULL
#>   ..$ : chr [1:23] "date_2018-05" "date_2018-06" "date_2018-07" "date_2018-09" ...
#kable(eh_main$eh[1:8,1:10])

table(paste(eh_main$first, eh_main$last, sep="_"))
#>
#>  1_23 10_23 11_23 12_23 13_23 14_23 15_23 16_23 17_23 18_23 19_23  2_23 20_23
#>   173   242   244   238   296   244   291   382   185   120   181   263    82
#> 21_23 22_23  3_23  4_23  5_23  6_23  7_23  8_23  9_23
#>   288   157   313    51   216   233   130   188   260
```

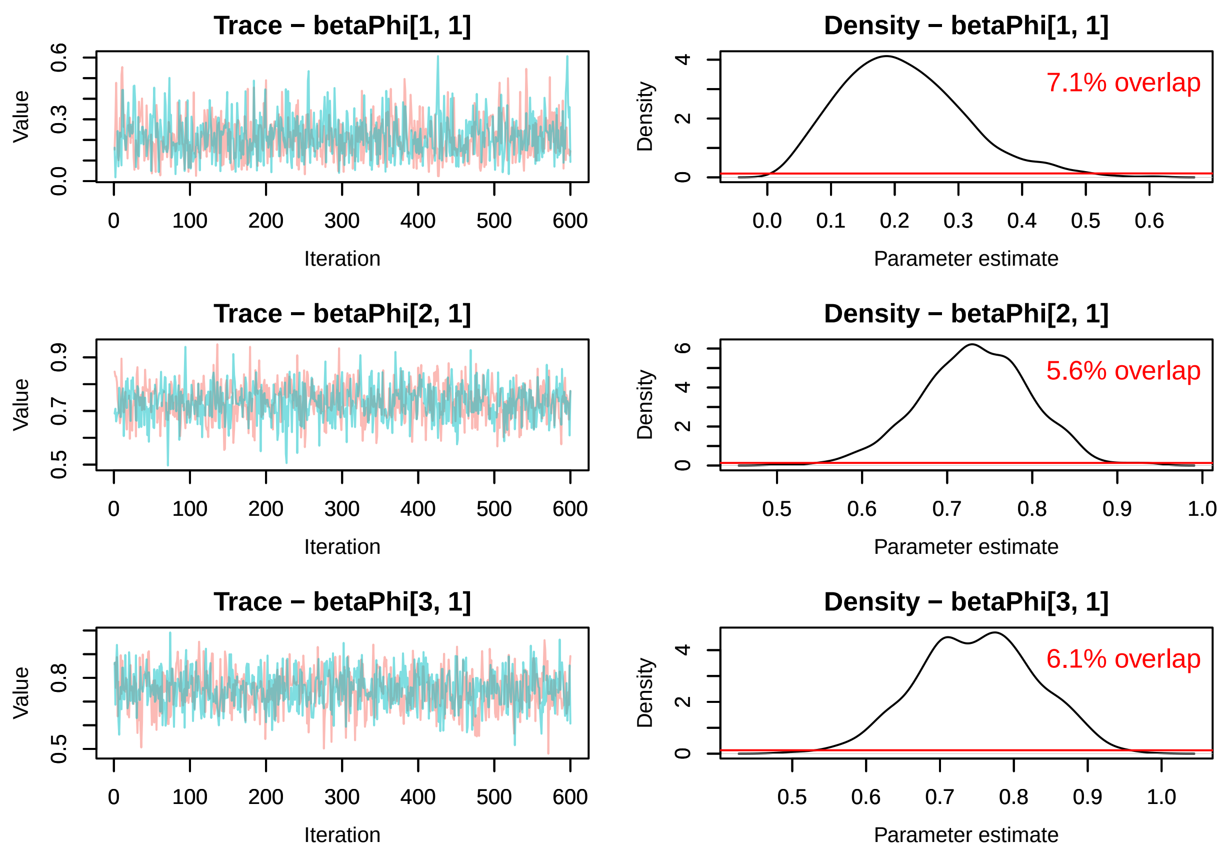
```
### Read the model run into global memory
if (tar_exist_objects(c("tt_modelOut_main"))) {
  mod_tt_main <- tar_read(tt_modelOut_main)

  MCMCplot(object = mod_tt_main$mcmc, params = "betaPhi")
  MCMCplot(object = mod_tt_main$mcmc, params = "betaP")

   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[1, 1]", "betaPhi[2, 1]", "betaPhi[3, 1]",
                        "betaPhi[1, 2]", "betaPhi[2, 2]", "betaPhi[3, 2]"
                        #"betaPhi[1, 3]", "betaPhi[2, 3]", "betaPhi[3, 3]"
```

```
                      ),
              pdf = FALSE,
              priors = priors
              )

       MCMCtrace(object = mod_tt_main$mcmc,
              ISB = FALSE,
              exact = TRUE,
              params = c("betaP[1, 1]", "betaP[2, 1]", "betaP[3, 1]",
                         "betaP[1, 2]", "betaP[2, 2]", "betaP[3, 2]"
                         #"betaP[1, 3]", "betaP[2, 3]", "betaP[3, 3]"
                         ),
              pdf = FALSE,
              priors = priors
              )

     modSummary_tt_main <- MCMCsummary(object = mod_tt_main$mcmc, round = 3) %>%
       rename(lo = '2.5%', med = '50%', hi = '97.5%')
}
```

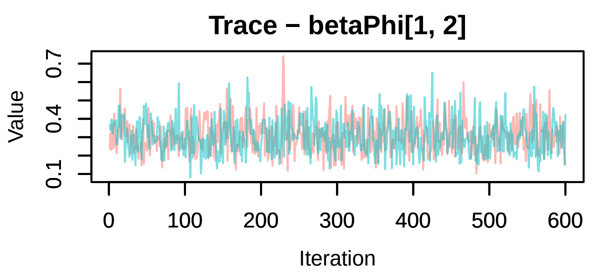


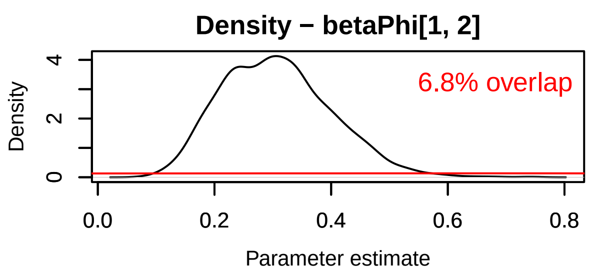
```
#> Warning in MCMCtrace(object = mod_tt_main$mcmc, ISB = FALSE, exact = TRUE, :
#> Only one prior specified for > 1 parameter. Using a single prior for all
#> parameters.
#> Warning in MCMCtrace(object = mod_tt_main$mcmc, ISB = FALSE, exact = TRUE, :
#> Number of samples in prior is greater than number of total or specified
#> iterations (for all chains) for specified parameter. Only last 1200 iterations
#> will be used.
```

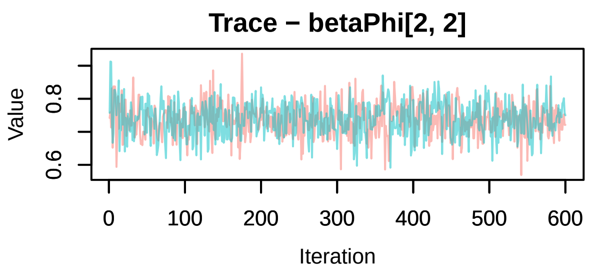


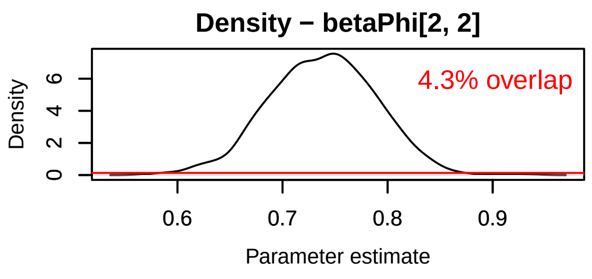


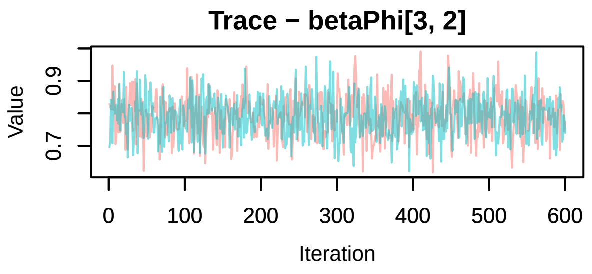
```
#> Warning in MCMCtrace(object = mod_tt_main$mcmc, ISB = FALSE, exact = TRUE, :
#> Only one prior specified for > 1 parameter. Using a single prior for all
#> parameters.
#> Warning in MCMCtrace(object = mod_tt_main$mcmc, ISB = FALSE, exact = TRUE, :
#> Number of samples in prior is greater than number of total or specified
#> iterations (for all chains) for specified parameter. Only last 1200 iterations
#> will be used.
```

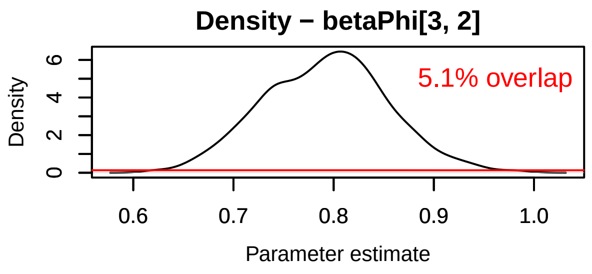

Trace − betaPhi[1, 2]
Value
0.1
0.4
0.7
0
100
200
300
400
500
600
Iteration

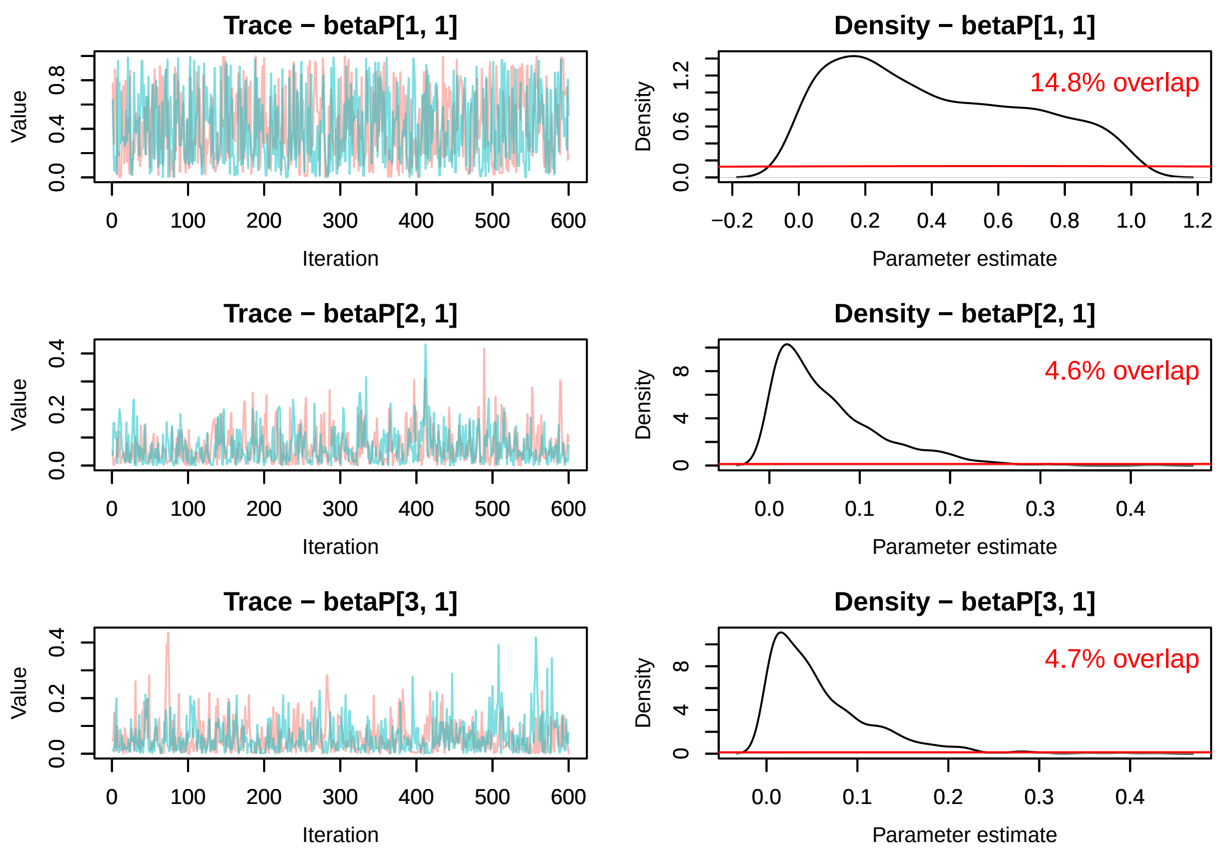

Density − betaPhi[1, 2]
Density
0
2
4
6.8% overlap
0.0
0.2
0.4
0.6
0.8
Parameter estimate

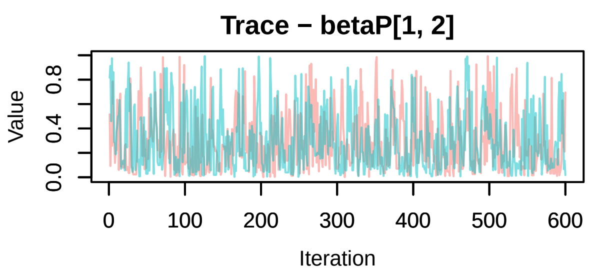

Trace − betaPhi[2, 2]
Value
0.6
0.8
0
100
200
300
400
500
600
Iteration

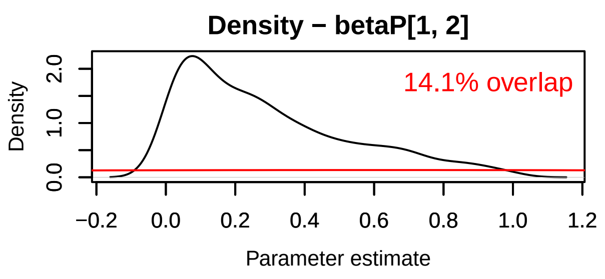

Density − betaPhi[2, 2]
Density
0
2
4
6
4.3% overlap
0.6
0.7
0.8
0.9
Parameter estimate

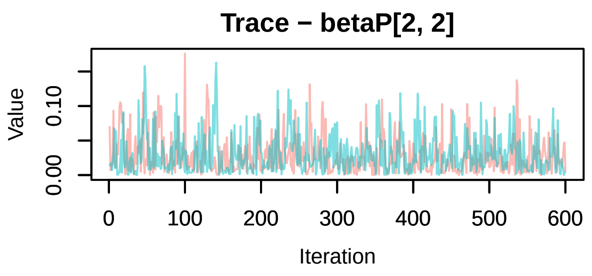

Trace − betaPhi[3, 2]
Value
0.7
0.9
0
100
200
300
400
500
600
Iteration

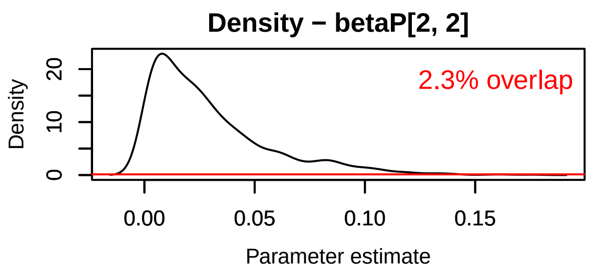

Density − betaPhi[3, 2]
Density
0
2
4
6
5.1% overlap
0.6
0.7
0.8
0.9
1.0
Parameter estimate

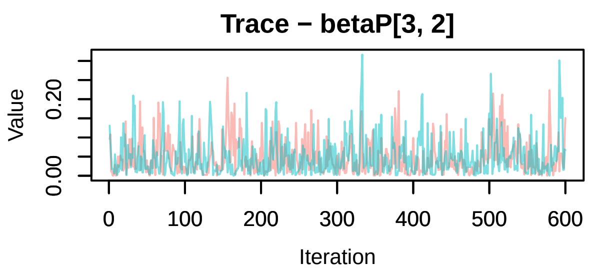

Trace − betaP[1, 1]
Density − betaP[1, 1]
14.8% overlap
Trace − betaP[2, 1]
Density − betaP[2, 1]
4.6% overlap
Trace − betaP[3, 1]
Density − betaP[3, 1]
4.7% overlap
Value
Iteration
Density
Parameter estimate

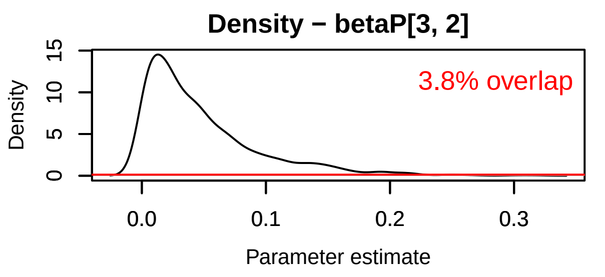








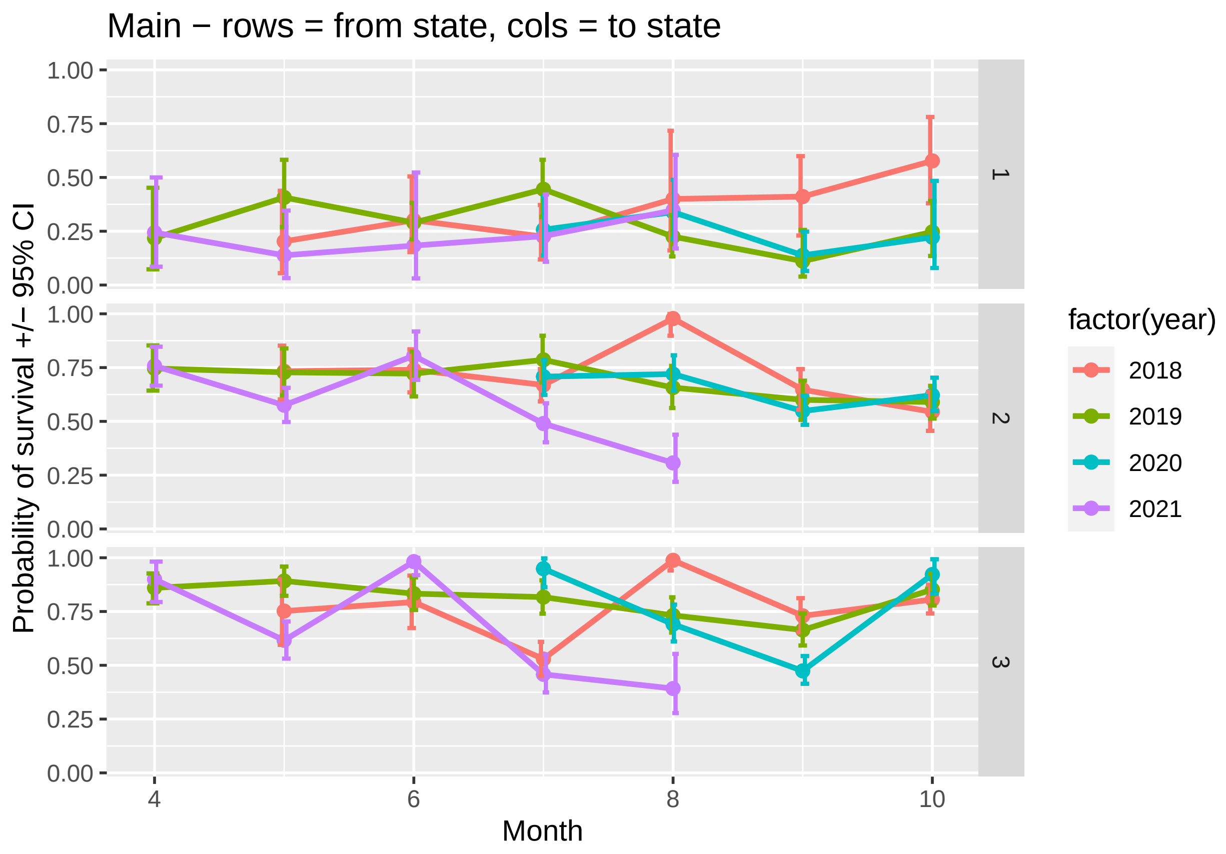




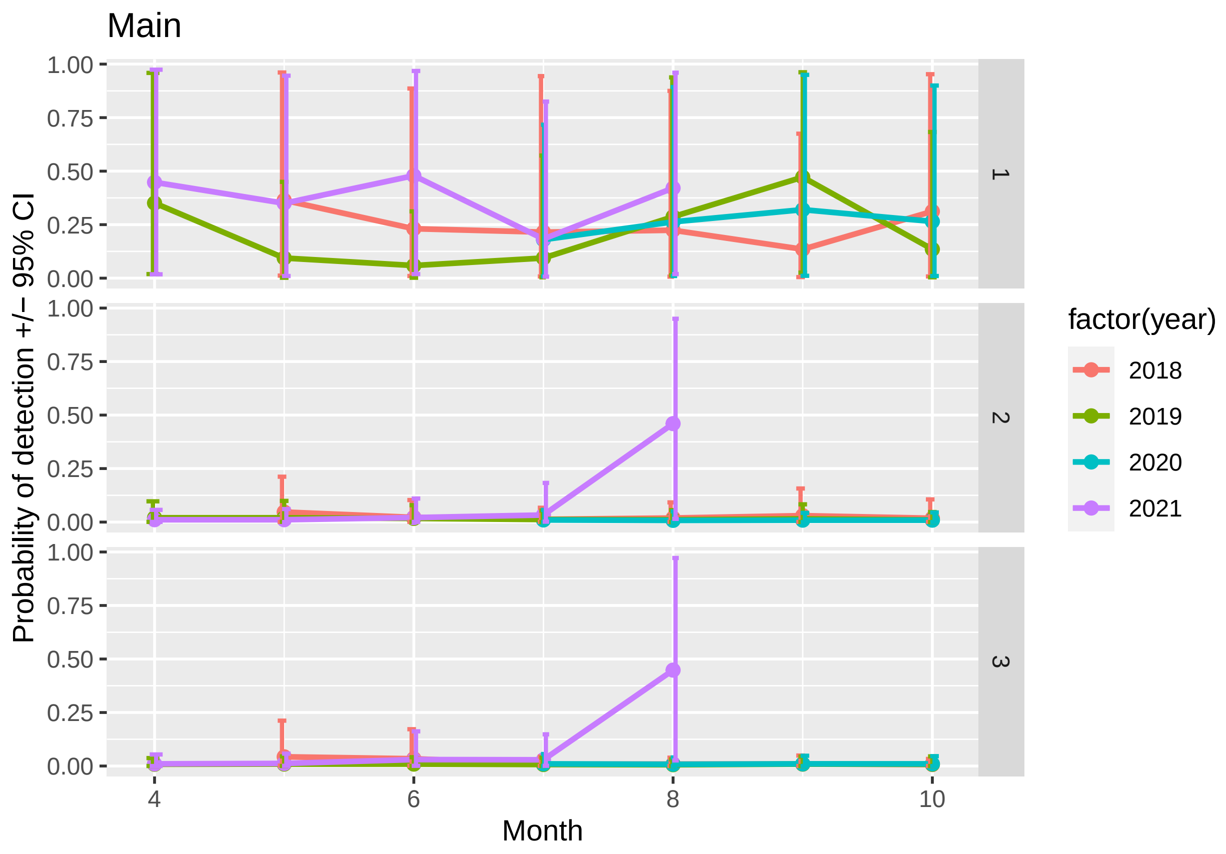
Main, phi, p and psi

```
nS <- mod_tt_main$myConstants$nStates
nT <- mod_tt_main$myConstants$T

phi_tt_main <- modSummary_tt_main %>%
  filter(substr(row.names(modSummary_tt_main), 1, 7) == "betaPhi") %>%
  add_column(data.frame(state = 1:nS, dateYM = rep(1:(nT - 1), each = nS))) %>%
  mutate(year = years[dateYM],
         month = months[dateYM],
         river = "main")

p_tt_main <- modSummary_tt_main %>%
  filter(substr(row.names(modSummary_tt_main), 1, 6) == "betaP[") %>%
  add_column(data.frame(state = 1:nS, dateYM = rep(1:(nT - 1), each = nS))) %>%
  mutate(year = years[dateYM],
         month = months[dateYM],
         river = "main")

psi_tt_main <- modSummary_tt_main %>%
  filter(substr(row.names(modSummary_tt_main), 1, 3) == "psi") %>%
  add_column(data.frame(stateFrom = 1:nS, stateTo = rep(1:nS, each = nS), dateYM = rep(1:(nT - 1), ea
  mutate(year = years[dateYM],
         month = months[dateYM],
           river = "main")
```

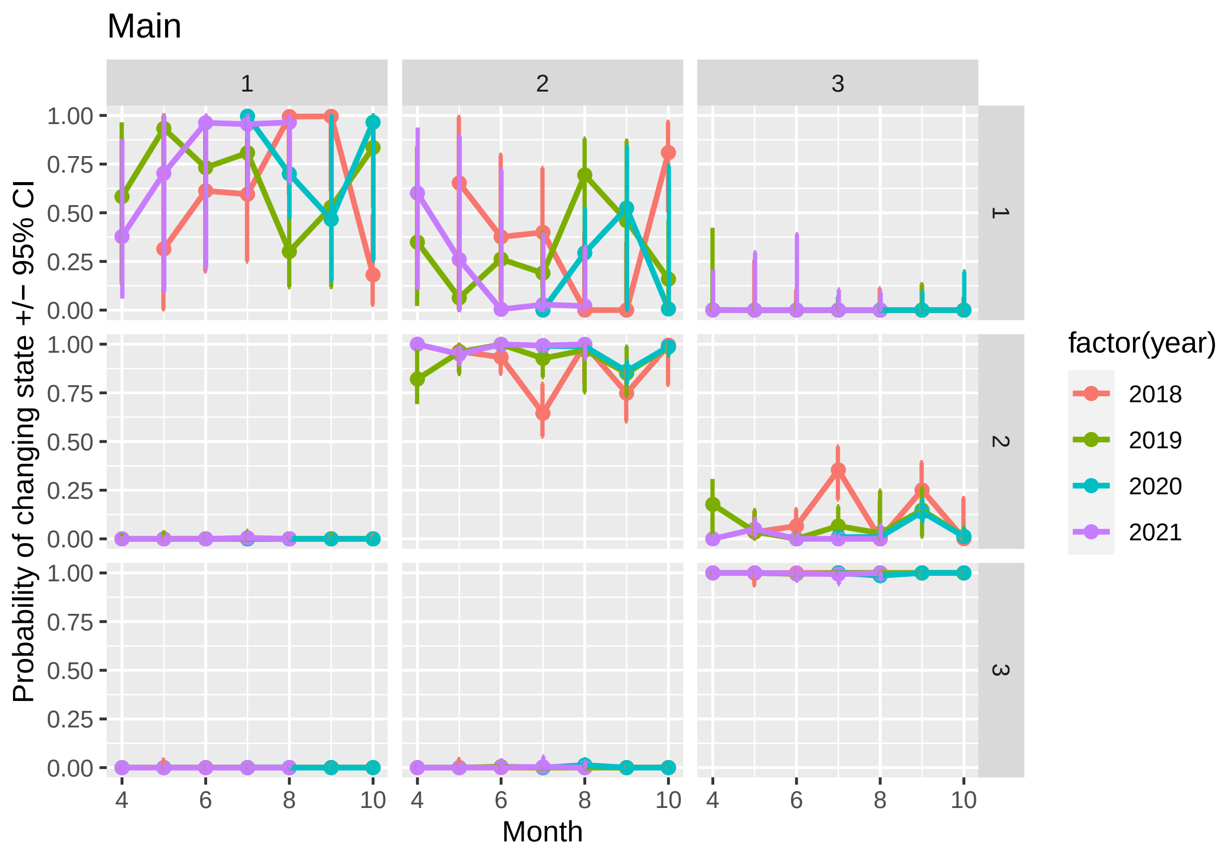
```
#phi
ggplot(phi_tt_main, aes(month, med, color = factor(year))) +
  geom_point(size = 2) +
  geom_line(size = 1) +
  geom_errorbar(aes(ymin = lo, ymax = hi),
                width = 0.2,
                position = position_dodge(0.05),
                size = 0.75) +
  ylab("Probability of survival +/- 95% CI") +
  xlab("Month") +
  ggtitle("Main - rows = from state, cols = to state") +
  facet_grid(rows = vars(state))
```



```
# p
ggplot(p_tt_main, aes(month, med, color = factor(year))) +
  geom_point(size = 2) +
  geom_line(size = 1) +
  geom_errorbar(aes(ymin = lo, ymax = hi),
                width = 0.2,
                position = position_dodge(0.05),
                size = 0.75) +
  ylab("Probability of detection +/- 95% CI") +
  xlab("Month") +
  ggtitle("Main") +
  facet_grid(rows = vars(state))
```



```
#psi
ggplot(psi_tt_main, aes(month, med, color = factor(year))) +
  geom_point(size = 2) +
  geom_line(size = 1) +
  geom_errorbar(aes(ymin = lo, ymax = hi),
              width = 0.2,
              position = position_dodge(0.03),
              size = 0.75) +
  ylab("Probability of changing state +/- 95% CI") +
  xlab("Month") +
  ggtitle("Main") +
  facet_grid(rows = vars(stateFrom),
             cols = vars(stateTo))
```



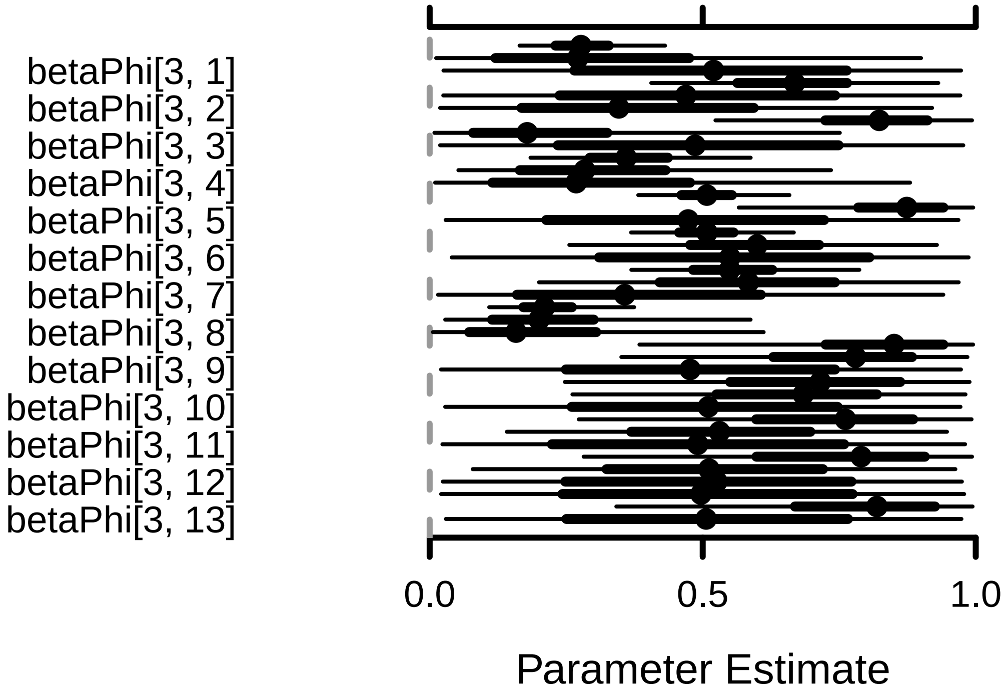
phiT_pT_psiT_trib

Load data 'trib' for analysis

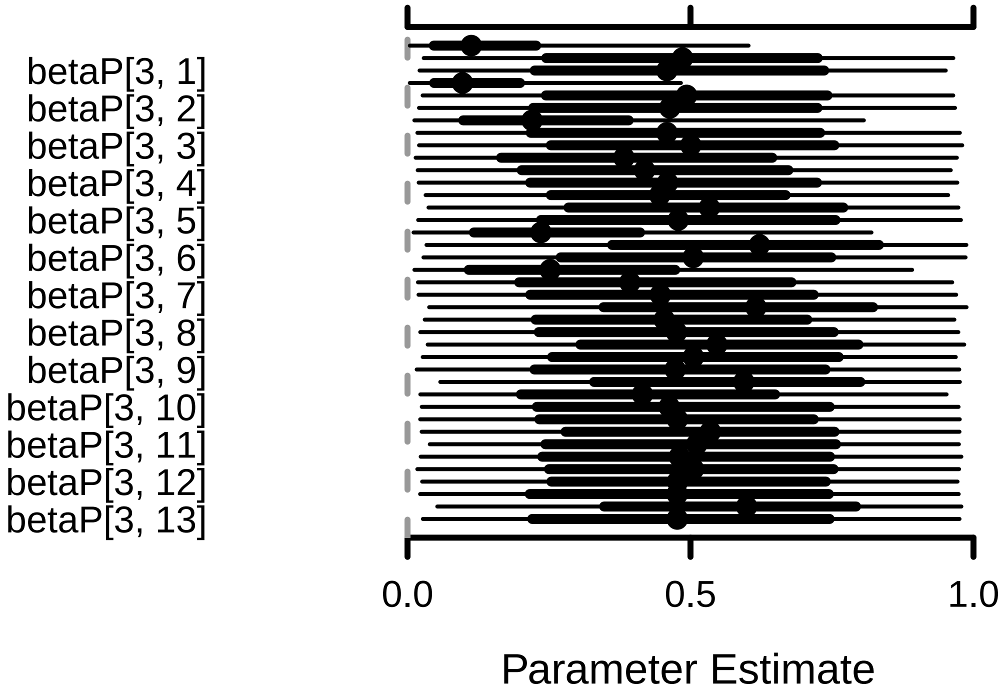
```
#d_tt <- tar_read(target_d_trib)
eh_trib <- tar_read(target_eh_trib)

str(eh_trib$eh)
#>  num [1:288, 1:14] 1 1 1 1 1 1 1 1 1 1 ...
#>  - attr(*, "dimnames")=List of 2
#>   ..$ : NULL
#>   ..$ : chr [1:14] "date_2018-07" "date_2018-08" "date_2018-09" "date_2018-10" ...
#kable(eh_trib$eh[1:8,1:10])

table(paste(eh_trib$first, eh_trib$last, sep="_"))
#>
#> 1_14 2_14 3_14 4_14 5_14 6_14 7_14 8_14
#>   60    8   12   15  100   53   20   20
```

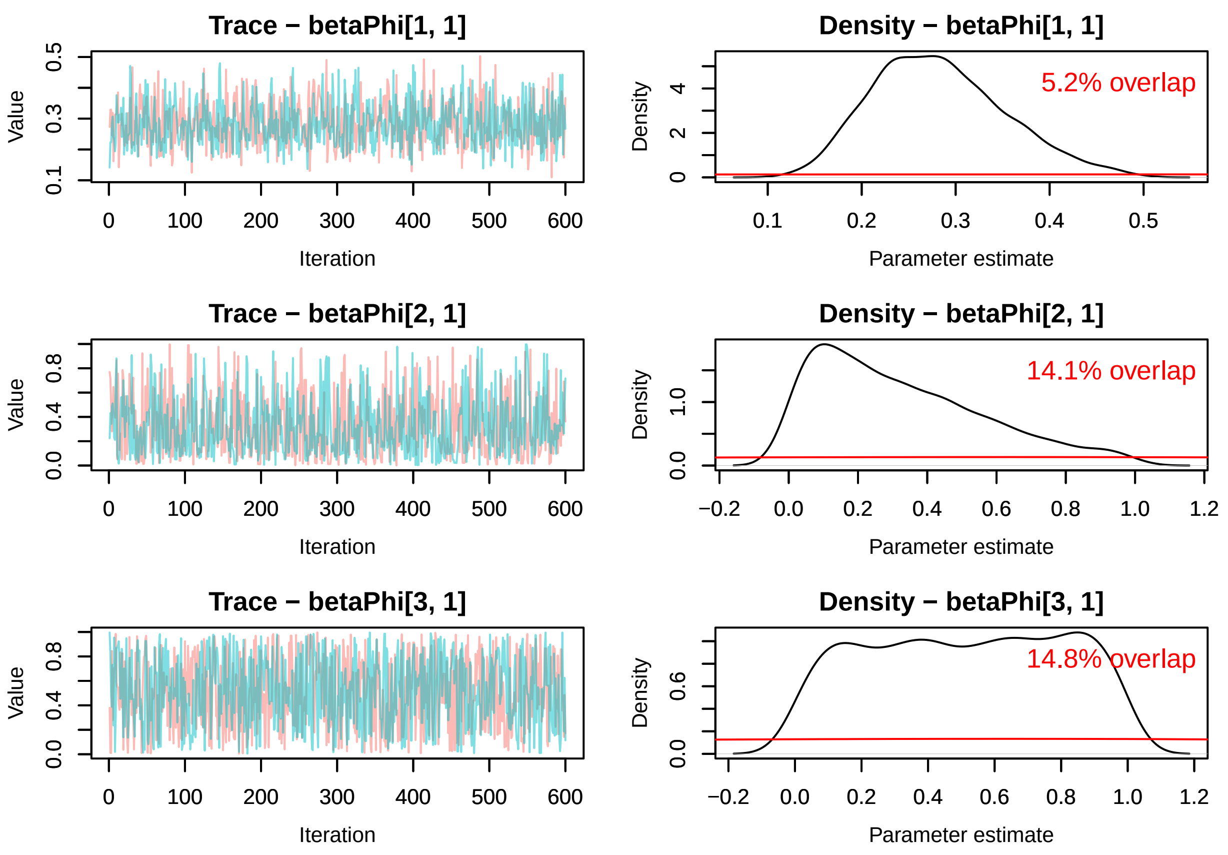
Plot trib model estimates

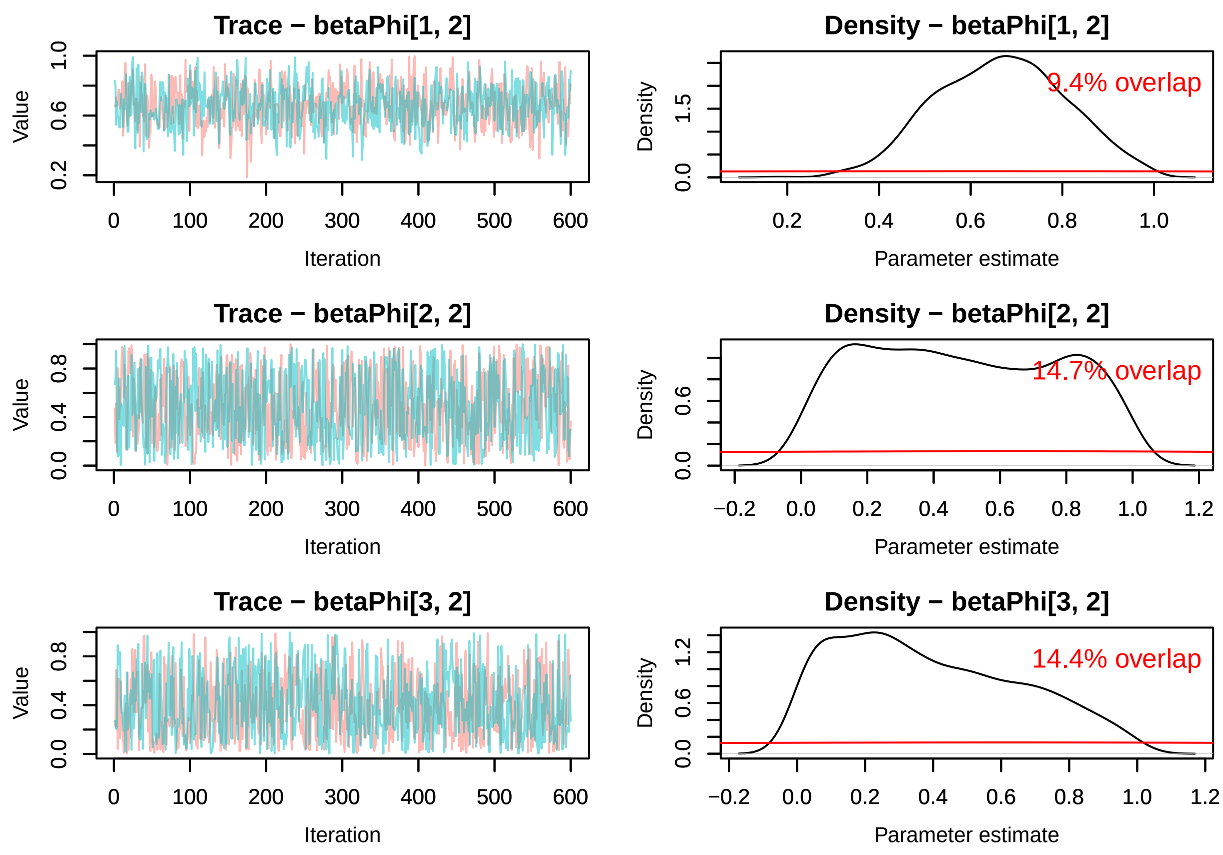
```
### Read the model run into global memory
if (tar_exist_objects(c("tt_modelOut_trib"))) {
  mod_tt_trib <- tar_read(tt_modelOut_trib)
  MCMCplot(object = mod_tt_trib$mcmc, params = "betaPhi")
  MCMCplot(object = mod_tt_trib$mcmc, params = "betaP")

   MCMCtrace(object = mod_tt_trib$mcmc,
             ISB = FALSE,
             exact = TRUE,
             params = c("betaPhi[1, 1]", "betaPhi[2, 1]", "betaPhi[3, 1]",
                        "betaPhi[1, 2]", "betaPhi[2, 2]", "betaPhi[3, 2]",
                        "betaPhi[1, 3]", "betaPhi[2, 3]", "betaPhi[3, 3]"),
             pdf = FALSE,
             priors = priors
             )

    MCMCtrace(object = mod_tt_trib$mcmc,
           ISB = FALSE,
           exact = TRUE,
           params = c("betaP[1, 1]", "betaP[2, 1]", "betaP[3, 1]",
                      "betaP[1, 2]", "betaP[2, 2]", "betaP[3, 2]",
                      "betaP[1, 3]", "betaP[2, 3]", "betaP[3, 3]"),
           pdf = FALSE,
           priors = priors
           )

    modSummary_tt_trib <- MCMCsummary(object = mod_tt_trib$mcmc, round = 3) %>%
      rename(lo = '2.5%', med = '50%', hi = '97.5%')
}
```



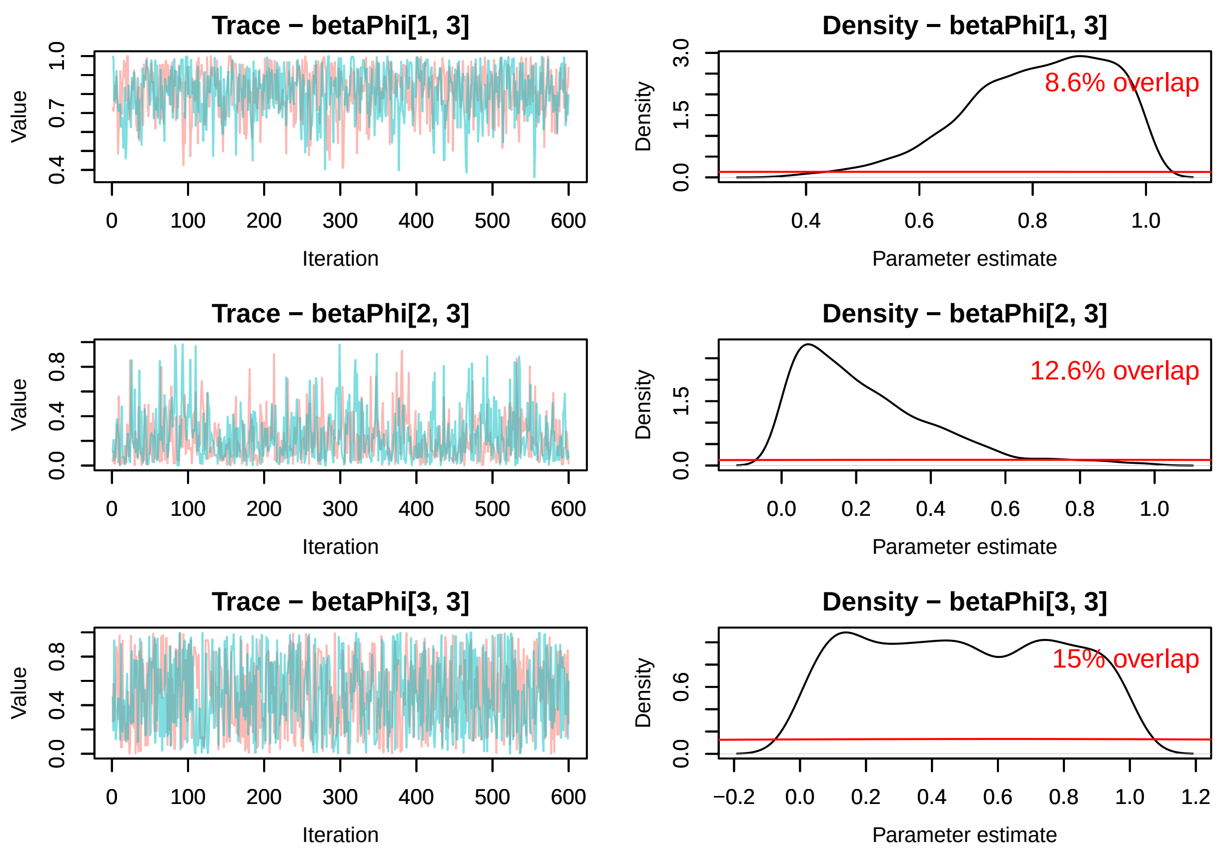
```
#> Warning in MCMCtrace(object = mod_tt_trib$mcmc, ISB = FALSE, exact = TRUE, :
#> Only one prior specified for > 1 parameter. Using a single prior for all
#> parameters.
#> Warning in MCMCtrace(object = mod_tt_trib$mcmc, ISB = FALSE, exact = TRUE, :
#> Number of samples in prior is greater than number of total or specified
#> iterations (for all chains) for specified parameter. Only last 1200 iterations
#> will be used.
```

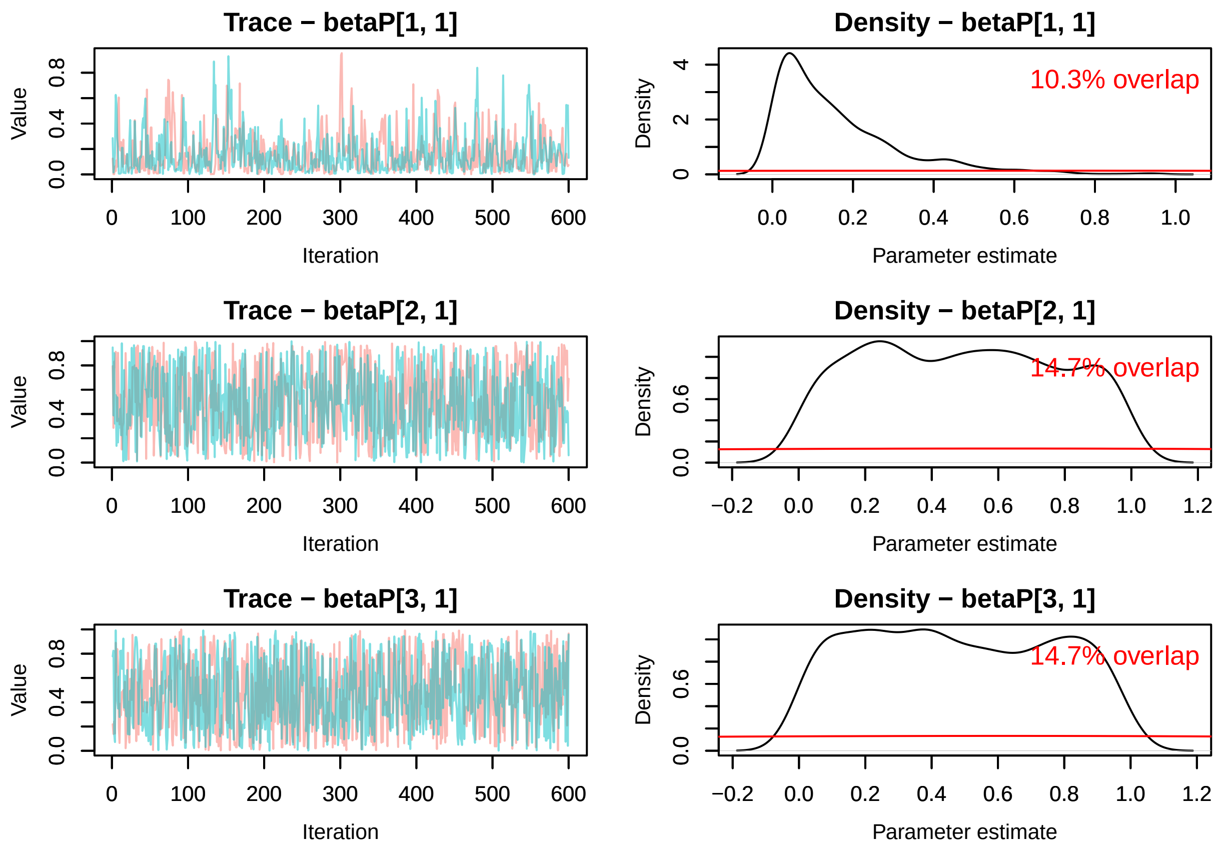

betaP[3, 1]
betaP[3, 2]
betaP[3, 3]
betaP[3, 4]
betaP[3, 5]
betaP[3, 6]
betaP[3, 7]
betaP[3, 8]
betaP[3, 9]
betaP[3, 10]
betaP[3, 11]
betaP[3, 12]
betaP[3, 13]
0.0
0.5
1.0
Parameter Estimate

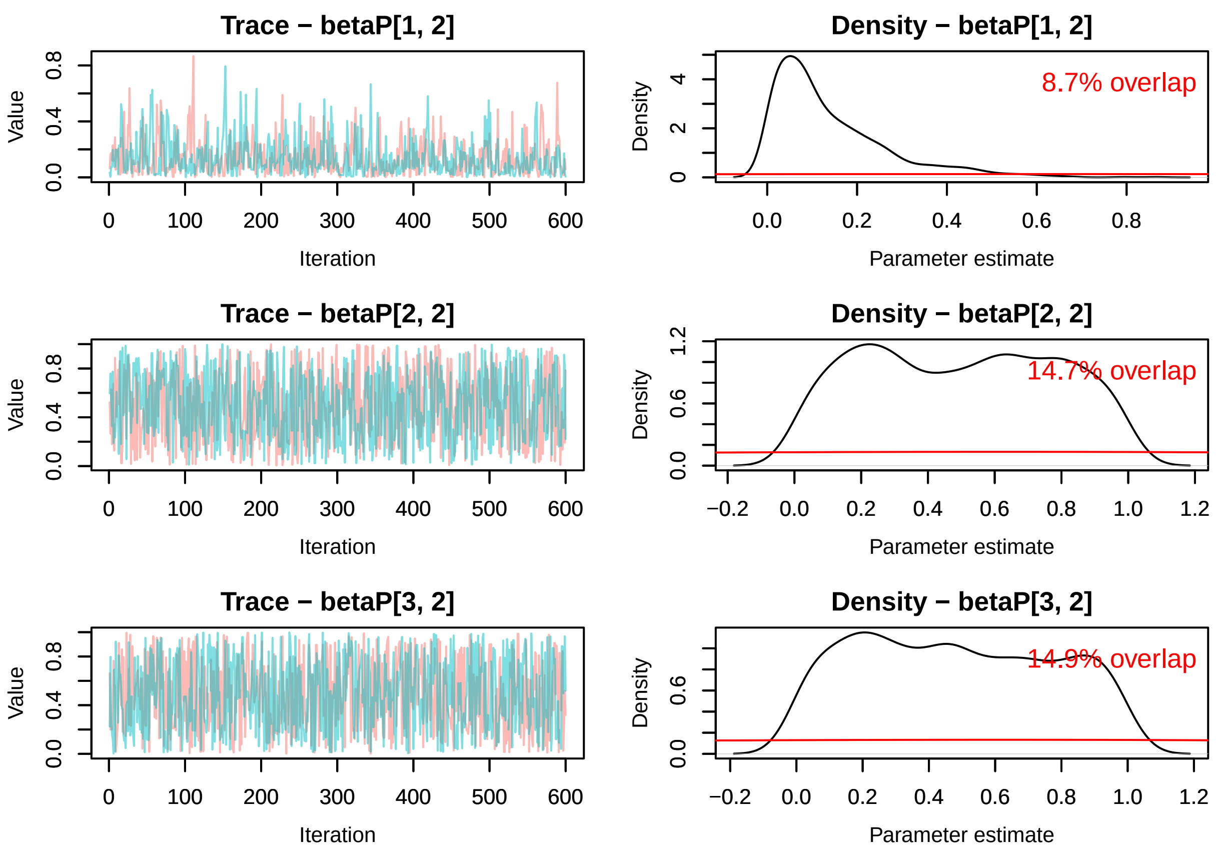

Trace − betaPhi[1, 1]
Density − betaPhi[1, 1]
5.2% overlap
Trace − betaPhi[2, 1]
Density − betaPhi[2, 1]
14.1% overlap
Trace − betaPhi[3, 1]
Density − betaPhi[3, 1]
14.8% overlap
Value
Density
Iteration
Parameter estimate

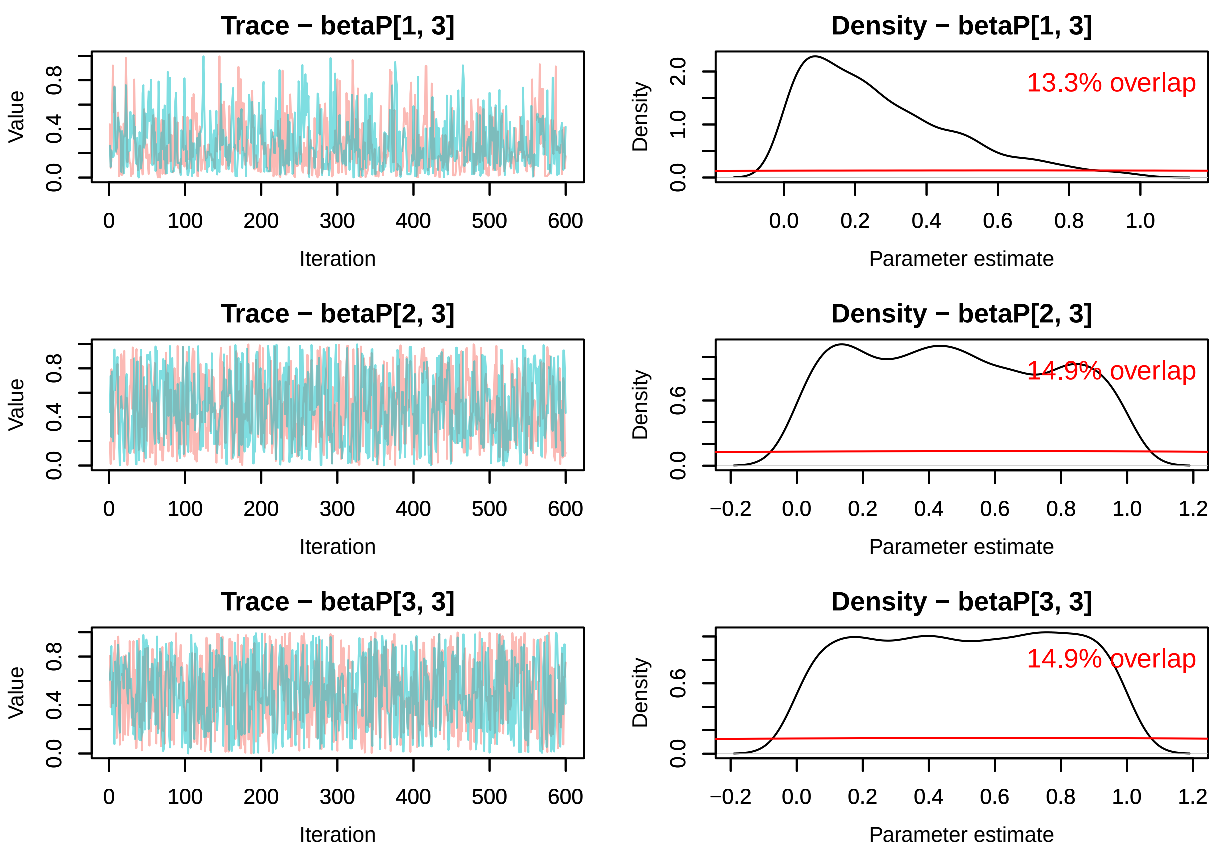


```
#> Warning in MCMCtrace(object = mod_tt_trib$mcmc, ISB = FALSE, exact = TRUE, :
#> Only one prior specified for > 1 parameter. Using a single prior for all
#> parameters.
#> Warning in MCMCtrace(object = mod_tt_trib$mcmc, ISB = FALSE, exact = TRUE, :
#> Number of samples in prior is greater than number of total or specified
#> iterations (for all chains) for specified parameter. Only last 1200 iterations
#> will be used.
```


Trace − betaPhi[1, 3]
Value
Iteration
Density − betaPhi[1, 3]
Density
Parameter estimate
8.6% overlap
Trace − betaPhi[2, 3]
Value
Iteration
Density − betaPhi[2, 3]
Density
Parameter estimate
12.6% overlap
Trace − betaPhi[3, 3]
Value
Iteration
Density − betaPhi[3, 3]
Density
Parameter estimate
15% overlap

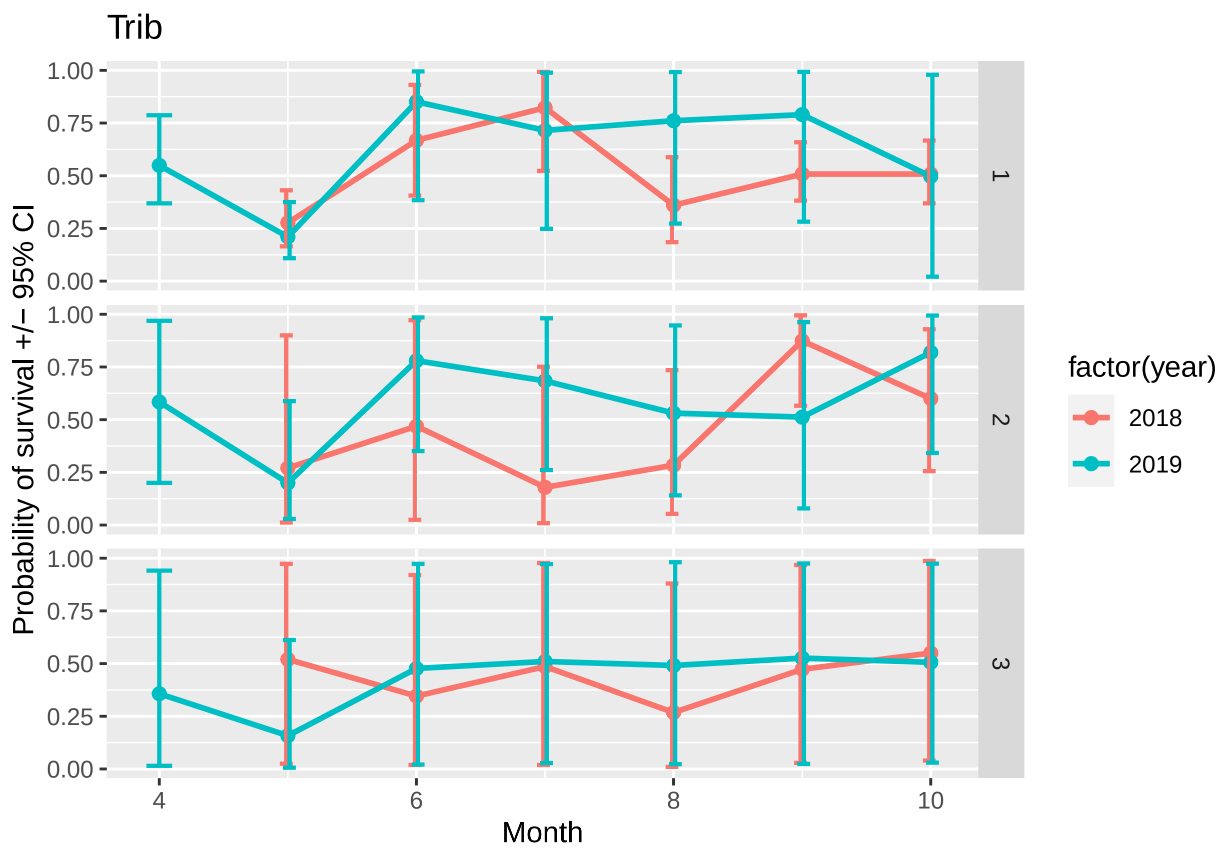

Trace − betaP[1, 1]
Density − betaP[1, 1]
10.3% overlap
Trace − betaP[2, 1]
Density − betaP[2, 1]
14.7% overlap
Trace − betaP[3, 1]
Density − betaP[3, 1]
14.7% overlap
Value
Iteration
Density
Parameter estimate


Trace − betaP[1, 2]
Value
Iteration
Density − betaP[1, 2]
Density
Parameter estimate
8.7% overlap
Trace − betaP[2, 2]
Value
Iteration
Density − betaP[2, 2]
Density
Parameter estimate
14.7% overlap
Trace − betaP[3, 2]
Value
Iteration
Density − betaP[3, 2]
Density
Parameter estimate
14.9% overlap

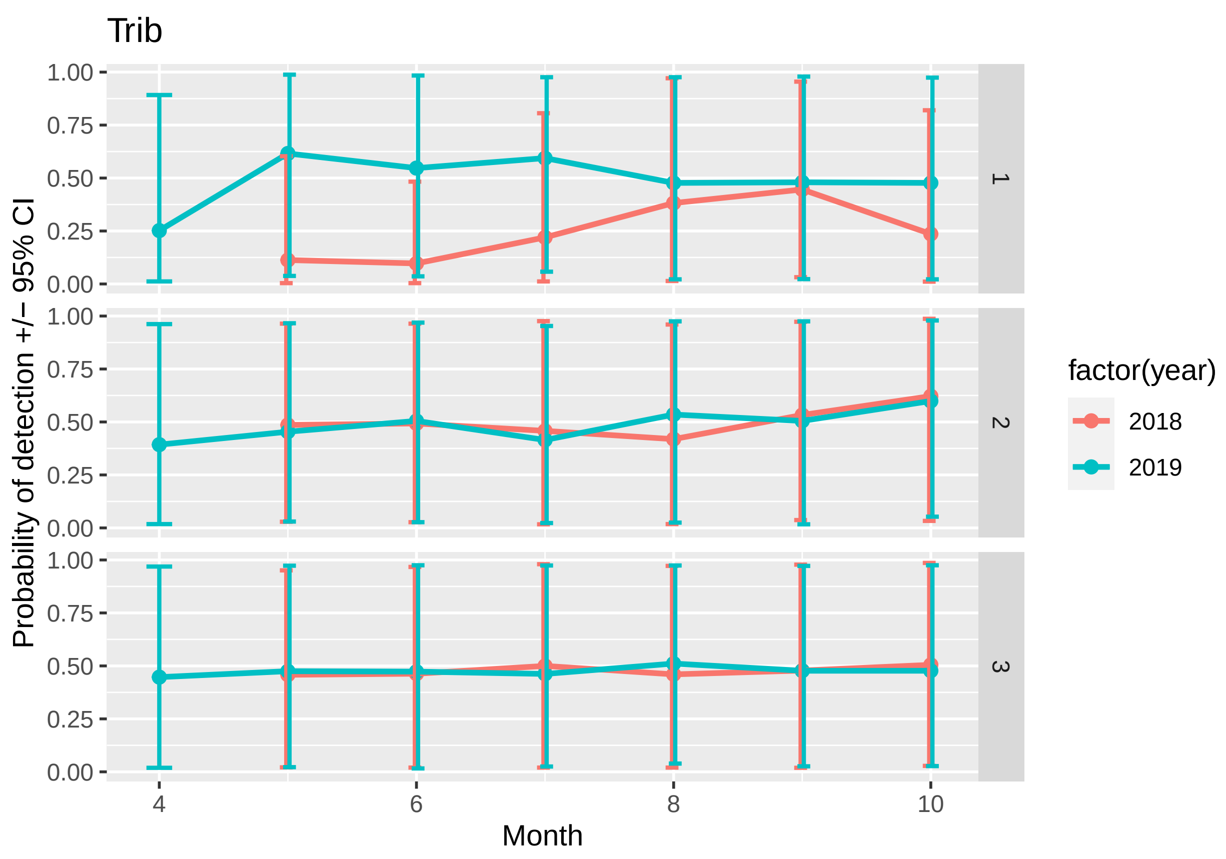


```
nS <- mod_tt_trib$myConstants$nStates
nT <- mod_tt_trib$myConstants$T

phi_tt_trib <- modSummary_tt_trib %>%
  filter(substr(row.names(modSummary_tt_trib), 1, 7) == "betaPhi") %>%
  add_column(data.frame(state = 1:nS, dateYM = rep(1:(nT - 1), each = nS))) %>%
  mutate(year = years[dateYM],
         month = months[dateYM],
         river = "trib")

p_tt_trib <- modSummary_tt_trib %>%
  filter(substr(row.names(modSummary_tt_trib), 1, 6) == "betaP[") %>%
  add_column(data.frame(state = 1:nS, dateYM = rep(1:(nT - 1), each = nS))) %>%
  mutate(year = years[dateYM],
         month = months[dateYM],
         river = "trib")

psi_tt_trib <- modSummary_tt_trib %>%
  filter(substr(row.names(modSummary_tt_trib), 1, 3) == "psi") %>%
  add_column(data.frame(stateFrom = 1:nS, stateTo = rep(1:nS, each = nS), dateYM = rep(1:(nT - 1), ea
  mutate(year = years[dateYM],
         month = months[dateYM],
         river = "trib")
```

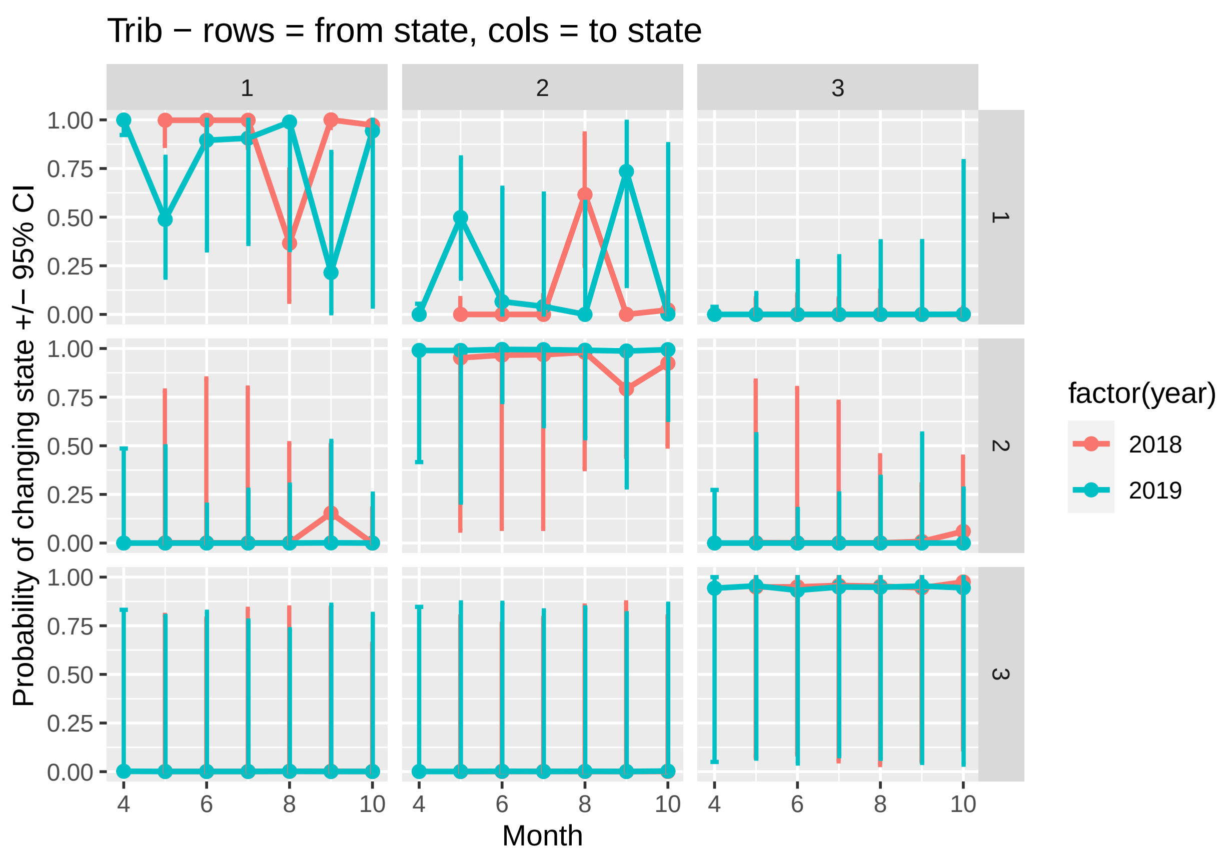
```
#phi
ggplot(phi_tt_trib, aes(month, med, color = factor(year))) +
  geom_point(size = 2) +
  geom_line(size = 1) +
  geom_errorbar(aes(ymin = lo, ymax = hi),
                width = 0.2,
                position = position_dodge(0.05),
                size = 0.75) +
  ylab("Probability of survival +/- 95% CI") +
  xlab("Month") +
  ggtitle("Trib") +
  facet_grid(rows = vars(state))
```



```
# p
ggplot(p_tt_trib, aes(month, med, color = factor(year))) +
  geom_point(size = 2) +
  geom_line(size = 1) +
  geom_errorbar(aes(ymin = lo, ymax = hi),
                width = 0.2,
                position = position_dodge(0.05),
                size = 0.75) +
  ylab("Probability of detection +/- 95% CI") +
  xlab("Month") +
  ggtitle("Trib") +
  facet_grid(rows = vars(state))
```



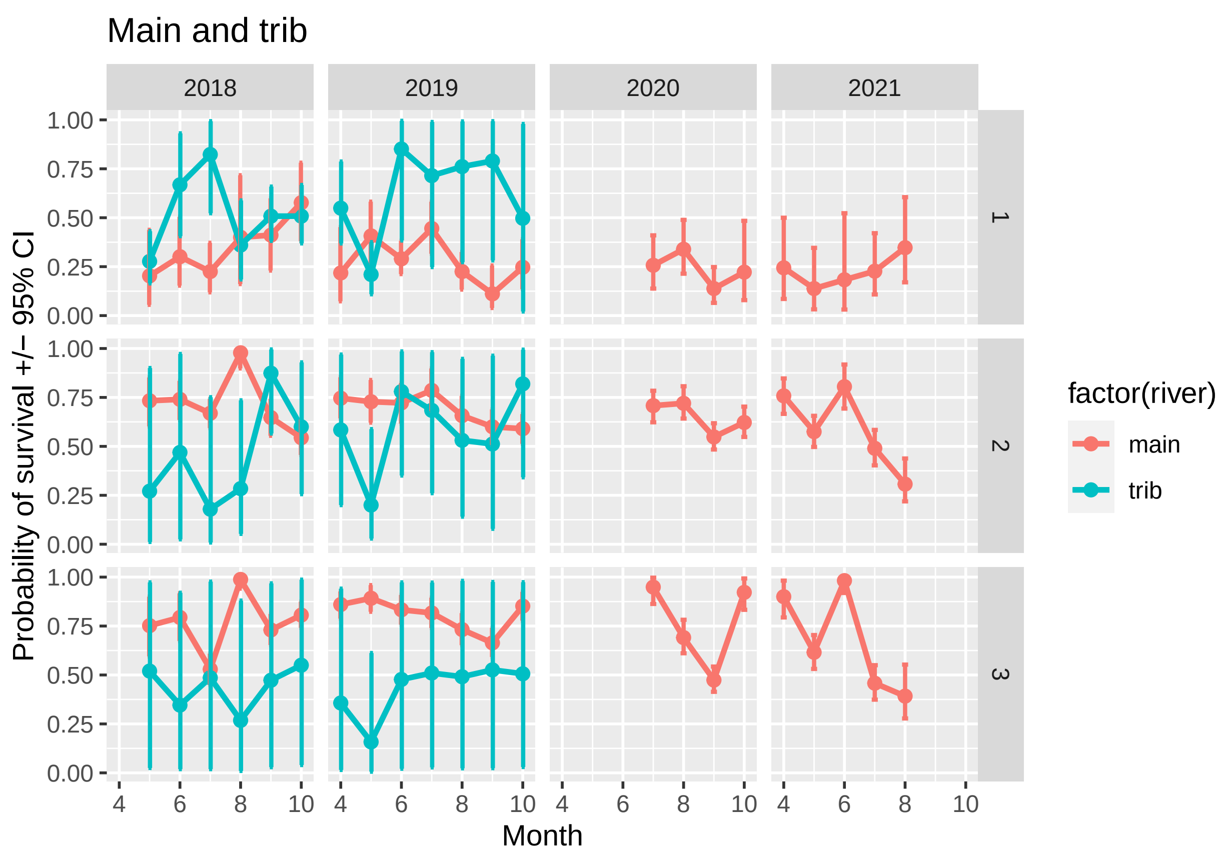
```
#psi
ggplot(psi_tt_trib, aes(month, med, color = factor(year))) +
  geom_point(size = 2) +
  geom_line(size = 1) +
  geom_errorbar(aes(ymin = lo, ymax = hi),
                width = 0.2,
                position = position_dodge(0.03),
                size = 0.75) +
  ylab("Probability of changing state +/- 95% CI") +
  xlab("Month") +
  ggtitle("Trib - rows = from state, cols = to state") +
  facet_grid(rows = vars(stateFrom),
             cols = vars(stateTo))
```



Combine main and trib estimates

```
phi_tt_mainTrib <- add_row(phi_tt_main, phi_tt_trib)
psi_tt_mainTrib <- add_row(psi_tt_main, psi_tt_trib)
```

```
ggplot(phi_tt_mainTrib, aes(month, med, color = factor(river))) +
  geom_point(size = 2) +
  geom_line(size = 1) +
  geom_errorbar(aes(ymin = lo, ymax = hi),
                width = 0.2,
                position = position_dodge(0.05),
                size = 0.75) +
  ylab("Probability of survival +/- 95% CI") +
  xlab("Month") +
  ggtitle("Main and trib") +
  facet_grid(rows = vars(state),
             cols = vars(year))
```



Calculate means for broad comparisons

```
  # Overall means across occasions
# Main
  (
    phi_tt_main_means <- phi_tt_main %>%
    group_by(state) %>%
    summarise(meanPhi = mean(mean),
              sdPhi = sd(mean),
              meanLo = mean(lo),
              meanMed = mean(med),
              meanHi = mean(hi)) %>%
    mutate(river = "main") %>%
    ungroup()
  )
#> # A tibble: 3 x 7
#>   state meanPhi sdPhi meanLo meanMed meanHi river
#>   <int>   <dbl> <dbl>  <dbl>   <dbl>  <dbl> <chr>
#> 1     1   0.287 0.112  0.145   0.280  0.473 main
#> 2     2   0.667 0.132  0.575   0.667  0.759 main
#> 3     3   0.756 0.171  0.673   0.756  0.836 main

  (
    psi_tt_main_means <- psi_tt_main %>%
    group_by(stateFrom, stateTo) %>%
    summarise(meanPhi = mean(mean),
              sdPhi = sd(mean),
              meanLo = mean(lo),
              meanMed = mean(med),
              meanHi = mean(hi)) %>%
    mutate(river = "main") %>%
    ungroup()
    )
#> `summarise()` has grouped output by 'stateFrom'. You can override using the
#> `.groups` argument.
#> # A tibble: 9 x 8
#>   stateFrom stateTo meanPhi    sdPhi    meanLo   meanMed  meanHi river
#>       <int>   <int>   <dbl>    <dbl>     <dbl>     <dbl>   <dbl> <chr>
#> 1         1       1 0.687   0.226    0.349     0.705     0.931   main
#> 2         1       2 0.298   0.225    0.0653    0.280     0.637   main
#> 3         1       3 0.0147  0.0128   0         0.0000455 0.135   main
#> 4         2       1 0.00232 0.00186  0         0.000227  0.0169  main
#> 5         2       2 0.928   0.0901   0.845     0.935     0.977   main
#> 6         2       3 0.0702  0.0904   0.0222    0.0634    0.152   main
#> 7         3       1 0.00109 0.000750 0         0         0.00968 main
#> 8         3       2 0.003   0.00405  0.0000909 0.00105   0.0175  main
#> 9         3       3 0.996   0.00420  0.978     0.999     1.00    main

# Trib
  (
    phi_tt_trib_means <- phi_tt_trib %>%
    group_by(state) %>%
    summarise(meanPhi = mean(mean),
              sdPhi = sd(mean),
              meanLo = mean(lo),
              meanMed = mean(med),
              meanHi = mean(hi)) %>%
    mutate(river = "trib") %>%
    ungroup()
    )
#> # A tibble: 3 x 7
#>   state meanPhi  sdPhi meanLo meanMed meanHi river
#>   <int>   <dbl>  <dbl>  <dbl>   <dbl>  <dbl> <chr>
#> 1     1   0.568 0.194  0.286    0.578  0.798 trib
#> 2     2   0.526 0.206  0.179    0.522  0.901 trib
#> 3     3   0.449 0.0964 0.0222   0.436  0.933 trib

  (psi_tt_trib_means <- psi_tt_trib %>%
    group_by(stateFrom, stateTo) %>%
    summarise(meanPhi = mean(mean),
              sdPhi = sd(mean),
              meanLo = mean(lo),
              meanMed = mean(med),
              meanHi = mean(hi)) %>%
    mutate(river = "trib") %>%
    ungroup()
    )
```

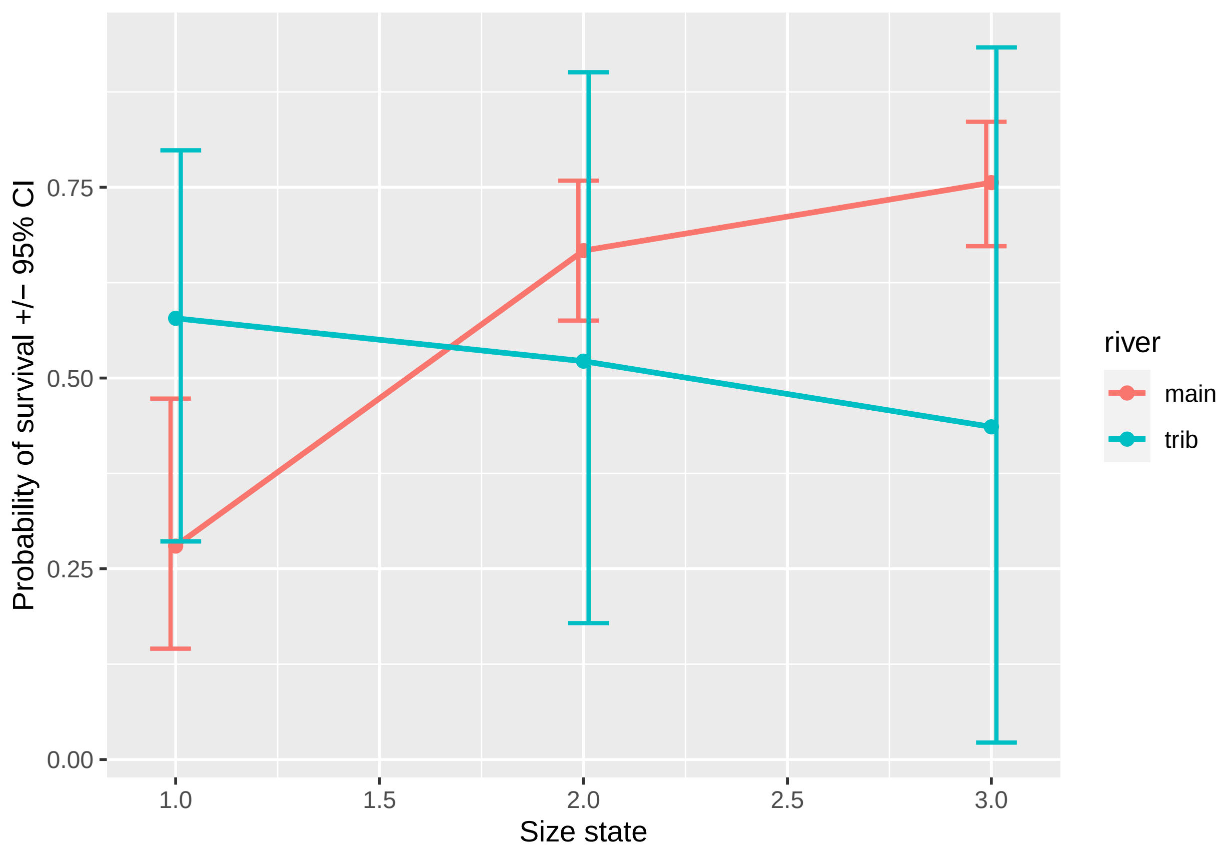
```
#> `summarise()` has grouped output by 'stateFrom'. You can override using the
#> `.groups` argument.
#> # A tibble: 9 x 8
#>   stateFrom stateTo meanPhi  sdPhi   meanLo   meanMed meanHi river
#>       <int>   <int>   <dbl>  <dbl>    <dbl>     <dbl>  <dbl> <chr>
#> 1         1       1  0.799  0.252  0.512    0.828      0.953 trib
#> 2         1       2  0.178  0.247  0.0446   0.152      0.456 trib
#> 3         1       3  0.0225 0.0263 0        0.0000769  0.209 trib
#> 4         2       1  0.0631 0.0465 0.000692 0.0122     0.474 trib
#> 5         2       2  0.880  0.0653 0.380    0.963      0.999 trib
#> 6         2       3  0.0566 0.0319 0        0.00562    0.461 trib
#> 7         3       1  0.1    0.0111 0        0.00131    0.800 trib
#> 8         3       2  0.104  0.0104 0        0.00138    0.831 trib
#> 9         3       3  0.797  0.0177 0.0618   0.950      1     trib
```

Combine mean main and trib estimates

```
phi_tt_mainTrib_means <- add_row(phi_tt_main_means, phi_tt_trib_means)
psi_tt_mainTrib_means <- add_row(psi_tt_main_means, psi_tt_trib_means)
```

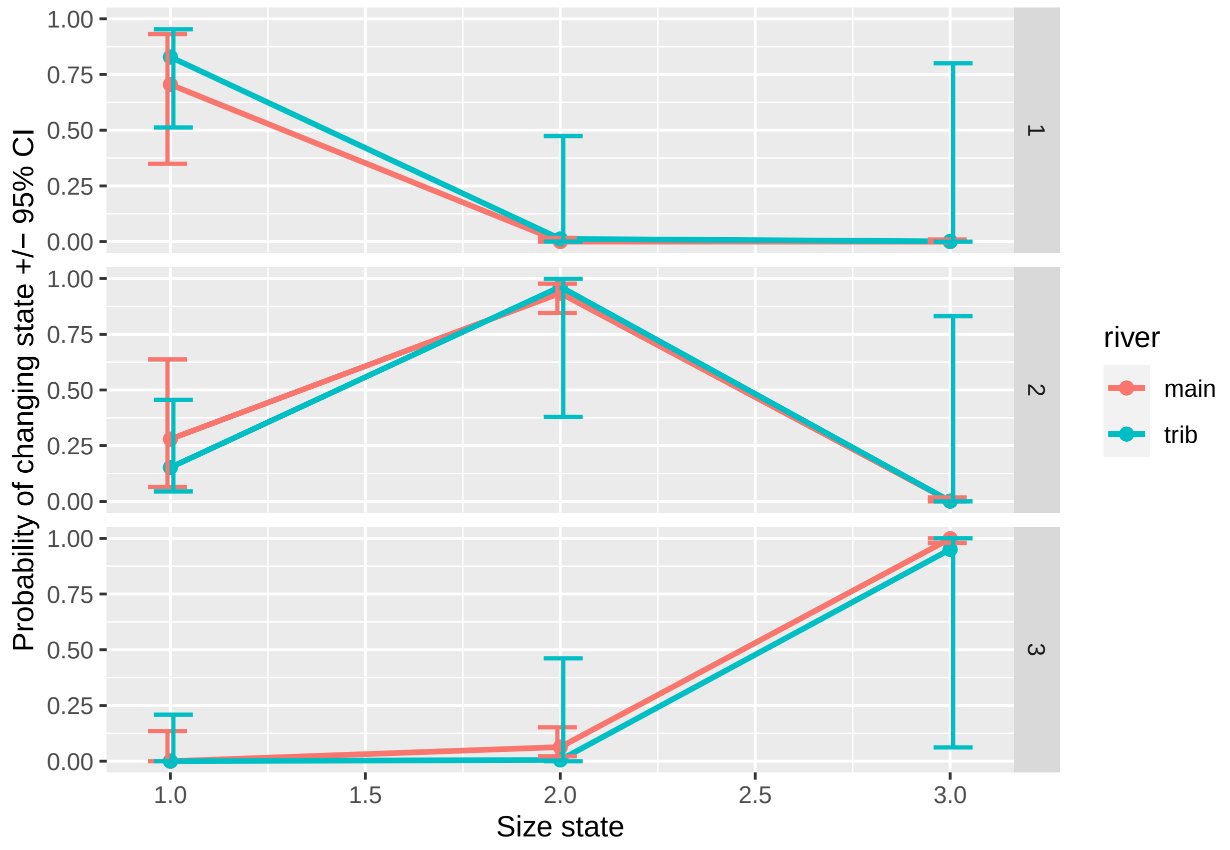
Probability of survival for each size state

```
ggplot(phi_tt_mainTrib_means, aes(state, meanMed, color = river)) +
  geom_point(size = 2) +
  geom_line(size = 1) +
  geom_errorbar(aes(ymin = meanLo, ymax = meanHi),
              width = 0.2,
              position = position_dodge(0.05),
              size = 0.75) +
  ylab("Probability of survival +/- 95% CI") +
  xlab("Size state")
```



Probability of changing state. 'From' size states are on the x-axis, 'to' size states are in the rows of the facets

```r
ggplot(psi_tt_mainTrib_means, aes(stateFrom, meanMed, color = river)) +
  geom_point(size = 2) +
  geom_line(size = 1) +
  geom_errorbar(aes(ymin = meanLo, ymax = meanHi),
             width = 0.2,
             position = position_dodge(0.03),
             size = 0.75) +
  ylab("Probability of changing state +/- 95% CI") +
  xlab("Size state") +
  facet_grid(rows = vars(stateTo))
```



Matrix model

```
# mMain <- matrix(
#   c()
# )
```

phiT__pT__psiT__mainTrib

Main and trib modeled together. Not using this as of 7-27-22.

```
### 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 =
#   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)
#
```