Delaware River PIT tag data analysis

Ben Letcher

2022-07-14

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#  tar_watch(seconds = 10, outdated = FALSE, targets_only = TRUE)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

```
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 main/trib and size

```
tagN_mt_s <- d %>%
  group_by(tag, mainTrib, sizeState) %>%
  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_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_mt_s$mainTrib, tagN_mt_s$sizeState, tagN_mt_s$n)
#> , ,  = 1
#> 
#>       
#>           1    2    3
#>   main  963 1660 1025
#>   trib  264   29    8
#> 
#> , ,  = 2
#> 
#>       
#>           1    2    3
#>   main   69  274  232
#>   trib   59    6    0
#> 
#> , ,  = 3
#> 
#>       
#>           1    2    3
#>   main    4   58   87
#>   trib   15    6    0
#> 
#> , ,  = 4
#> 
```

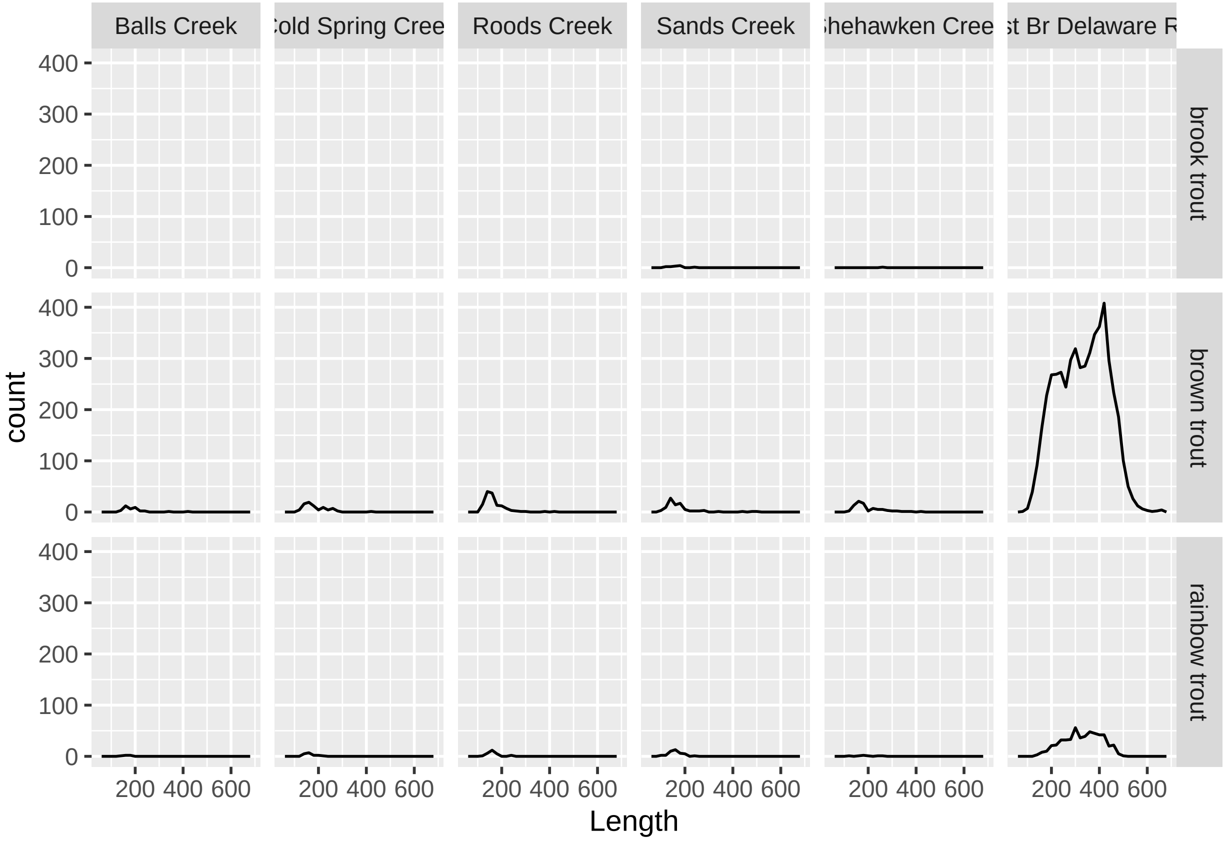
```
#> 
#>           1    2    3
#>   main    0   17   35
#>   trib    5    0    0
#> 
#> , ,  = 5
#> 
#> 
#>           1    2    3
#>   main    0    7   15
#>   trib    1    0    0
#> 
#> , ,  = 6
#> 
#> 
#>           1    2    3
#>   main    0    0    8
#>   trib    1    0    0
#> 
#> , ,  = 7
#> 
#> 
#>           1    2    3
#>   main    0    0    2
#>   trib    0    0    0
#> 
#> , ,  = 9
#> 
#> 
#>           1    2    3
#>   main    0    0    1
#>   trib    0    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$state, tagN_mt_s$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(!is.na(species)), aes(Length)) +
  geom_freqpoly() +
  facet_grid(species ~ Water)
#> `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	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
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$first, eh$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
```

Summary info for years and occasions

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

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

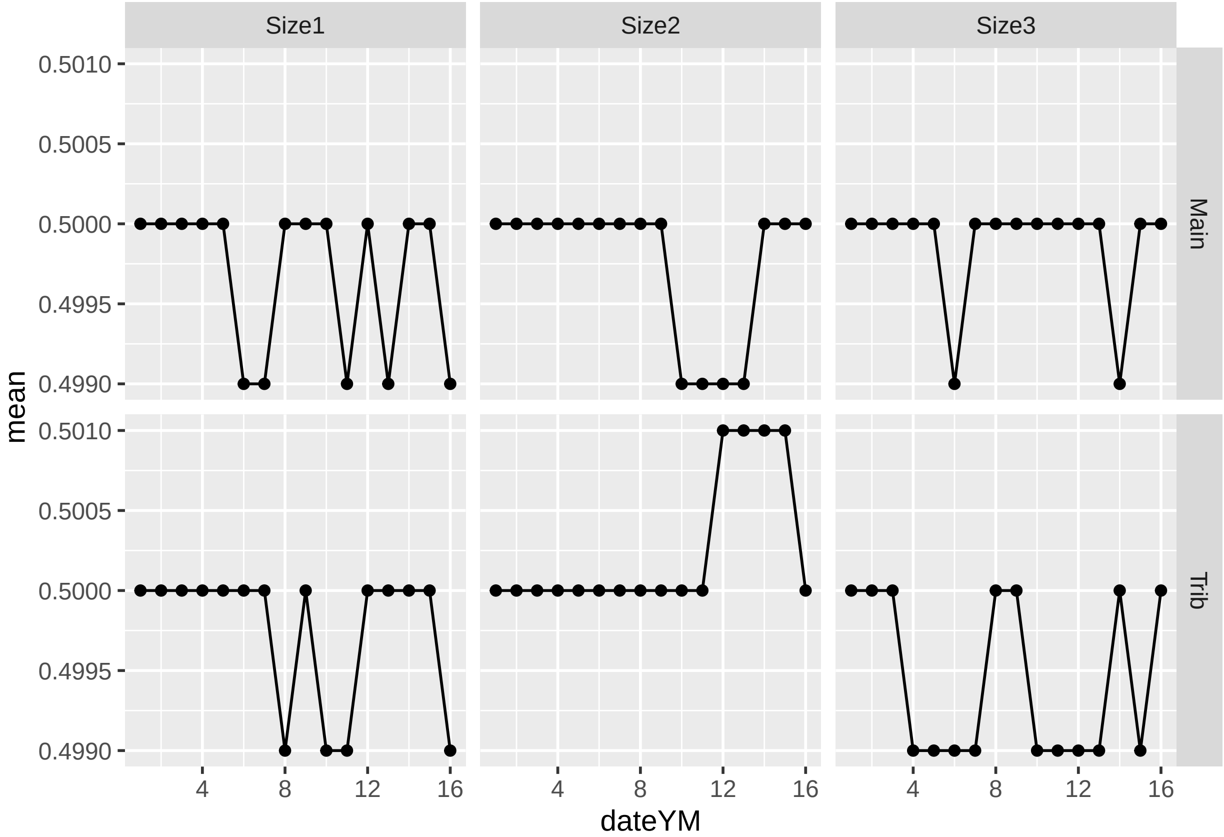
Models

'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.

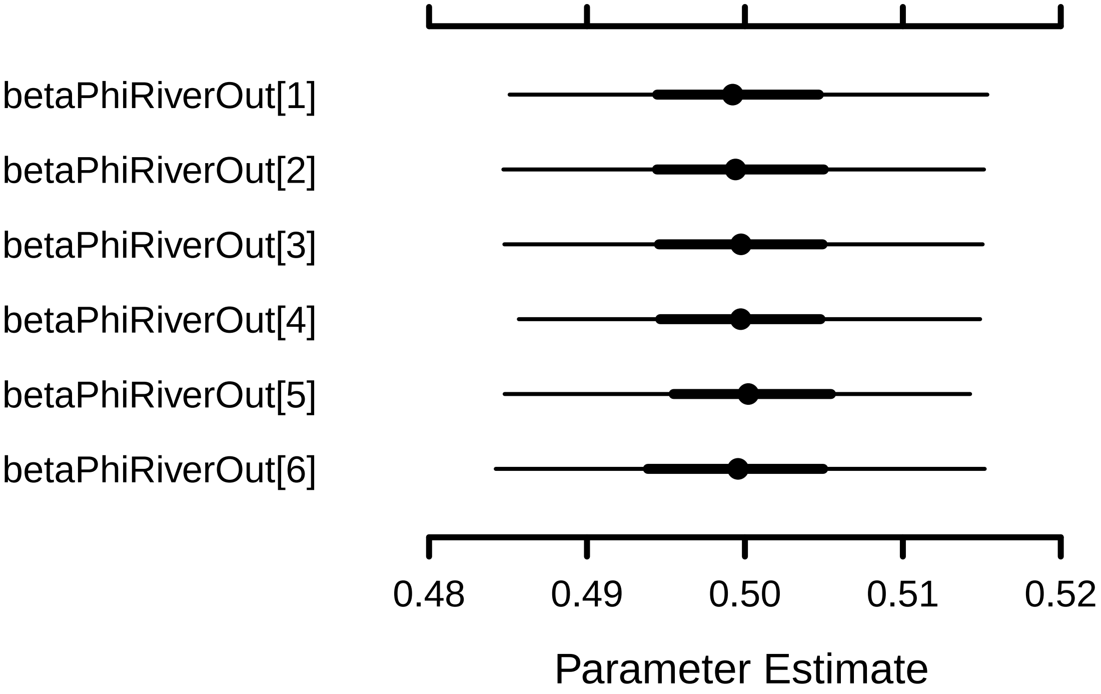
```
### Read the model run into global memory
mod <- tar_read(ttt_modelOut)

#MCMCplot(object = mod$mcmc)

modSummary <- MCMCsummary(object = mod$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)))
```

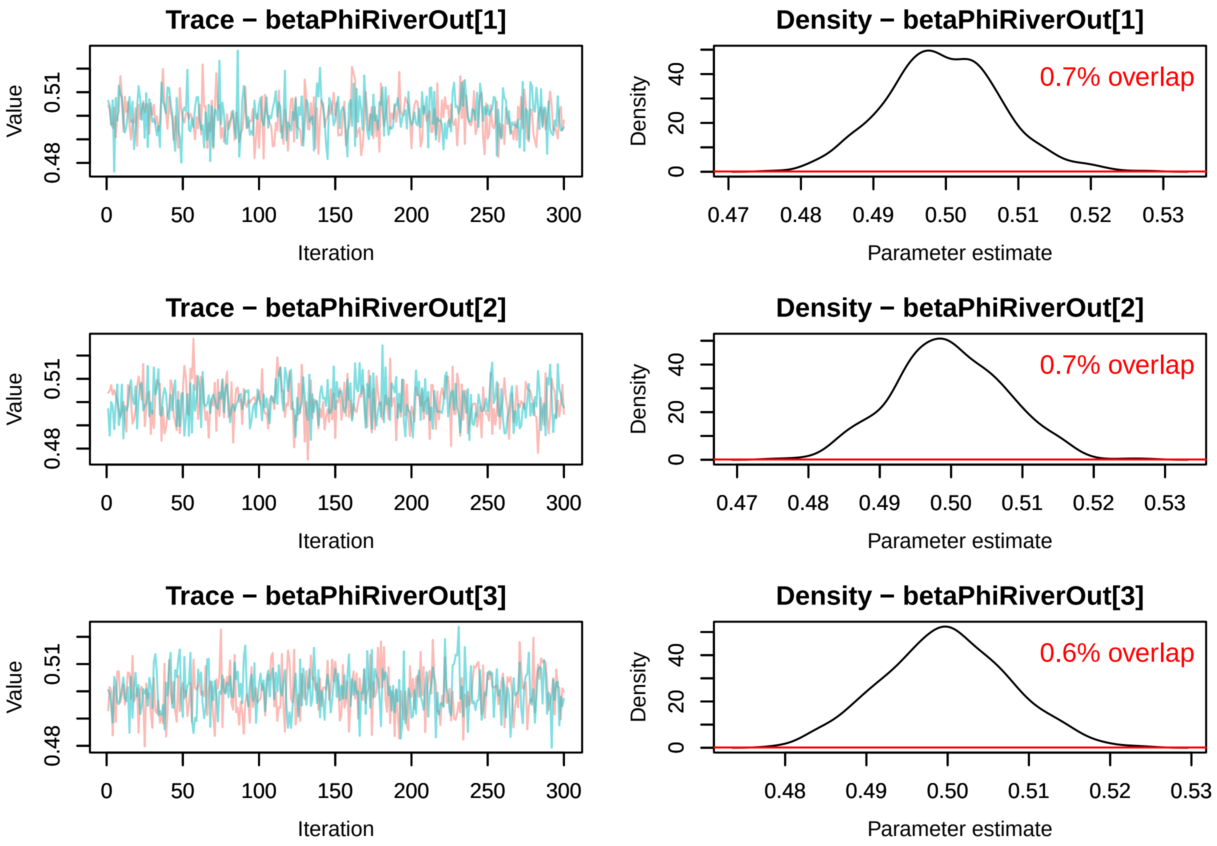
```
nS <- tar_read(nStates)
nT <- tar_read(ttt_myConstants)$T

modSummaryPhi <- modSummary %>%
  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, aes(dateYM, mean)) +
  geom_point() +
  geom_line() +
  facet_grid(mainTrib ~ size)
```

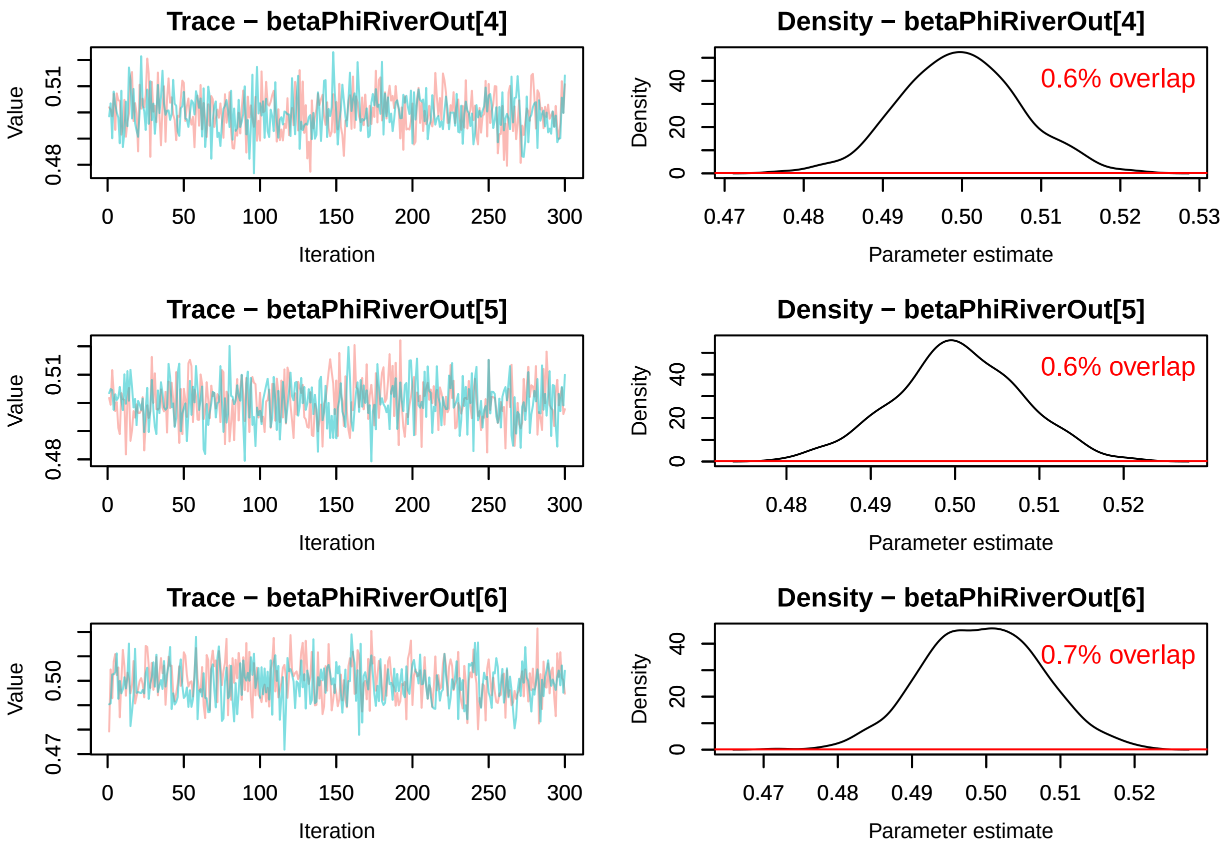


```
# modSummaryYears <- modSummary %>%
#   filter(substr(row.names(modSummary), 1, 3) == "betaPhiout") %>%
#   add_column(data.frame(year = years[1:15], dateYM = occs[1:15], occ = 1:15)) %>%
#   group_by(year) %>%
#   mutate(maxSampPerYear = occ == max(occ))
#
# kable(
#   modSummaryYears %>%
#   group_by(year) %>%
#   filter(!maxSampPerYear) %>%
#   summarize(phiProd = prod(mean),
#             dateRange = range(dateYM)) %>%
#     as.data.frame()
# )
  MCMCplot(object = mod$mcmc, params = "betaPhiRiverOut")
```



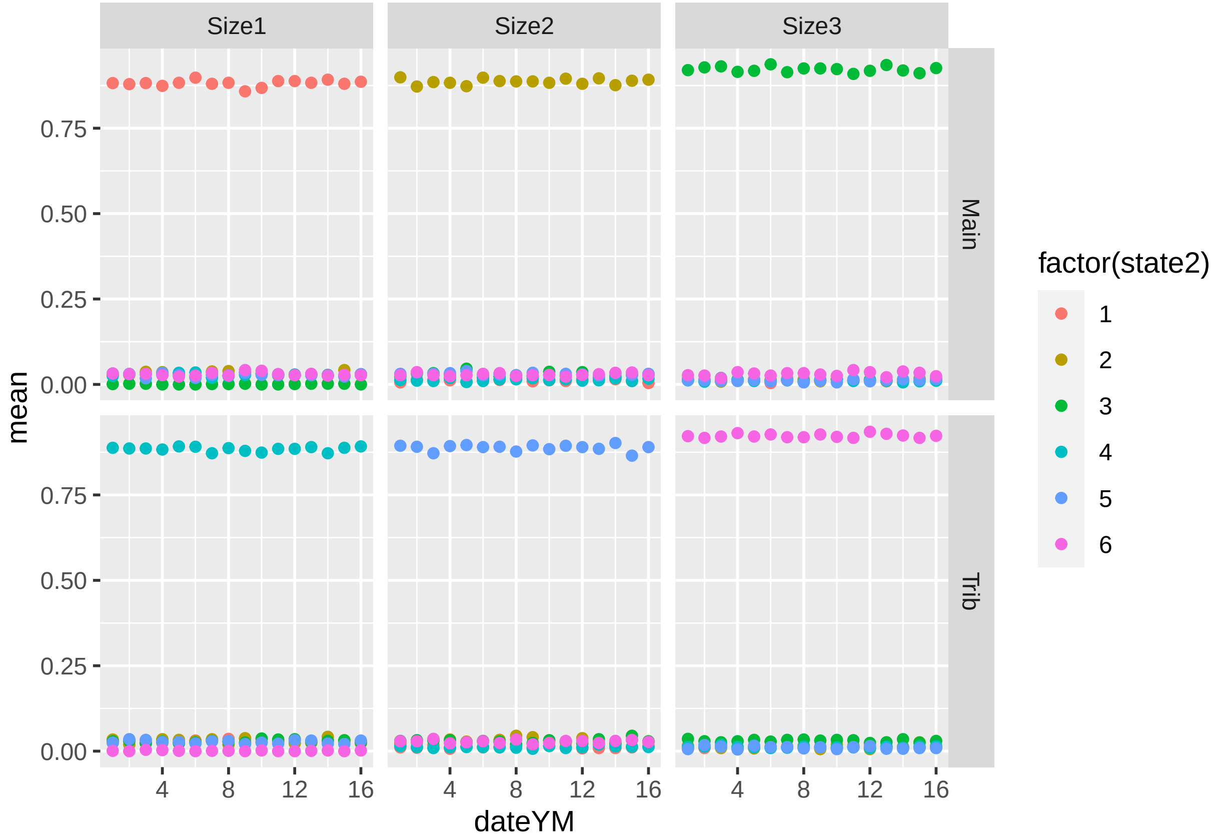
```
  priors <- rnorm(tar_read(ttt_runData)$nIter * tar_read(ttt_runData)$nChains, 0, 1/sqrt(.1))
  MCMCtrace(object = mod$mcmc,
            #ISB = FALSE,
            #exact = TRUE,
            params = c("betaPhiRiverOut"),
            pdf = FALSE,
            priors = priors
            )
#> Warning in MCMCtrace(object = mod$mcmc, params = c("betaPhiRiverOut"), pdf =
#> FALSE, : Only one prior specified for > 1 parameter. Using a single prior for
#> all parameters.
#> Warning in MCMCtrace(object = mod$mcmc, params = c("betaPhiRiverOut"), pdf =
#> FALSE, : Number of samples in prior is greater than number of total or specified
#> iterations (for all chains) for specified parameter. Only last 600 iterations
#> will be used.
```





```
#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 <- modSummary %>%
  filter(substr(row.names(modSummary), 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, aes(dateYM, mean, color = factor(state2))) +
  geom_point() +
  facet_grid(mainTrib ~ size)
```


Size1
Size2
Size3
Main
Trib
mean
dateYM
factor(state2)