

Environmental Drivers of Golden Tilefish Landings*

Vyacheslav Lyubchich and Geneviève Nesslage

Chesapeake Biological Laboratory, University of Maryland Center for Environmental Science,
Solomons, MD, 20688, USA

2020-04-07

Contents

1 Packages and functions	2
2 Data pre-processing	3
2.1 Load	3
2.2 Create lagged series, remove FMP period	3
2.3 Tests and visualizations	4
3 Random forest	8
3.1 Cross-validation	12
4 Generalized additive modeling (GAM)	13
4.1 Cross-validation	20
References	20

*These data and code support the results presented in (recommended citation):

Nesslage G, Lyubchich V, Nitschke P, Williams E, Grimes C, Wiedenmann J (2020) Environmental drivers of golden tilefish (*Lopholatilus chamaeleonticeps*) landings and catch per unit effort. In progress.

Citation for the current data and code:

Lyubchich V, Nesslage G (2020) github.com/vlyubchich/tilefish: Environmental Drivers of Golden Tilefish Landings. Zenodo. DOI 10.5281/zenodo.3732840. <https://doi.org/10.5281/zenodo.3732840>.

1 Packages and functions

```
rm(list = ls())
#Packages and custom functions used
#All packages are needed, but some are not loaded fully to avoid conflicts
library(Boruta)
# library(dplyr)
# library(forecast)
library(funtimes)
library(GGally)
# library(Hmisc)
library(mgcv)
library(ranger)
library(randomForest)
library(xtable)

summaryVL = function(X){
  out = sapply(c(1:ncol(X)), function(i) summary(X[,i])[1:6])
  colnames(out) = names(X)
  if(any(is.na(X))){
    Missing = apply(is.na(X), 2, sum)
    out = rbind(out, Missing)
  }
  out
}
```

Some colors to start with

```
COL = c(black = "black"
,red = rgb(100, 38, 33, maxColorValue = 100)
,green = rgb(38, 77, 19, maxColorValue = 100)
,blue = rgb(28, 24, 61, maxColorValue = 100)
,purple = rgb(76, 32, 72, maxColorValue = 100)
,cyan = rgb(21, 75, 87, maxColorValue = 100)
,dark_cyan = rgb(0, 47, 59, maxColorValue = 100)
,yellow = rgb(99, 90, 13, maxColorValue = 100)
)
par(mar = c(0, 0, 0, 0), mgp = c(0, 0, 0))
barplot(rep(1, length(COL)), col = COL, border = NA, axes = FALSE, space = c(0, 0))
```



2 Data pre-processing

2.1 Load

```
D = read.csv("./dataraw/landings.csv", stringsAsFactors = FALSE)
D = D[order(D$Year),]
rownames(D) = D$Year
#convert to the (unordered) factor:
D$Time_block = factor(D$Time_block, levels = unique(D$Time_block))
```

Table 1 gives a summary of the numeric variables (the last row is the number of missing values, i.e., NA's).

```
tmp = summaryVL(D[, -3])
```

Table 1: Summary statistics for numeric variables in the dataset

	Year	Landings	AMO_annual	AMO_DJFMA	NAO_DJF_st	NAO_DJF_PC	NAO_DJFMA_PC	NAO_DJFMA_st
Min.	1915.00	5.00	-0.43	-0.41	-5.20	-2.97	-2.14	-2.82
1st Qu.	1940.50	450.00	-0.15	-0.15	-1.30	-0.68	-0.55	-0.53
Median	1966.00	767.50	0.01	-0.02	0.40	-0.02	-0.02	0.16
Mean	1966.00	983.18	-0.00	-0.03	0.10	-0.03	-0.04	0.05
3rd Qu.	1991.50	1208.25	0.14	0.13	1.45	0.72	0.49	0.66
Max.	2017.00	4501.00	0.36	0.33	4.60	2.44	1.87	2.36
Missing	0.00	3.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 2 provides a summary of the categorical variables.

```
tmp = summary(D[, 3, drop = FALSE])
rownames(tmp) = NULL
```

Table 2: Summary statistics for categorical variables in the dataset

Time_block
1 Campaign: 6
2 Early :20
3 WWII : 5
4 PostWWII:25
5 Longline:30
6 FMP :17

2.2 Create lagged series, remove FMP period

Fisher et al (2014) suggest that landings are correlated with the Dec–Feb station-based NAO index lagged by up to seven years, so we add those lagged values to the analysis (and lagged values of the other climate indices).

```
n = names(D)
v = grep("AMO|NAO_", n) #variables to lag
L = 7 #number of lags to add to variables v
for (i in v) { #i=4
  tmp = sapply(1:L, function(x) dplyr::lag(D[,i], x))
  colnames(tmp) = paste(n[i], "_1", 1:L, sep = "")
  D = cbind(D, tmp)
}
```

Landings data from 2001–2017 (the terminal time block FMP) were removed from the analysis to exclude landings that were quota-limited.

```
#Save a copy of the full dataset as D0, cut off the FMP period with fishing restrictions
D0 = D
D = D[D$Time_block != "FMP", ]
D$Time_block = droplevels(D$Time_block)
```

2.3 Tests and visualizations

The landings time series has few missing values that can be reasonably guessed with linear interpolation (see Figure 1):

```
indNA = is.na(D$Landings) #save the location of what we fill-in for later
D$Landings = forecast::na.interp(D$Landings) #interpolate NAs
(TBend = tapply(D$Year, D$Time_block, max)) #end year of each time block

## Campaign    Early      WWII PostWWII Longline
##    1920      1940      1945      1970      2000

(TBcenter = round(tapply(D$Year, D$Time_block, mean))) #year-center of each time block

## Campaign    Early      WWII PostWWII Longline
##    1918      1930      1943      1958      1986
```

Try to remove the right skewness using a power transformation (log is too much, use square root):

```
D$sqrtLandings = sqrt(D$Landings)
```

Apply a non-parametric test for trends (Lyubchich et al, 2013) that is suitable for autocorrelated and conditionally heteroskedastic data. Specifically, we test the null hypothesis that the unknown trend function $\mu(t)$ in the observed time series is constant (i.e., the hypothesis of no trend) vs. the alternative of some other trend (including the cases of a linear, monotonic, or non-linear trend):

$$H_0 : \mu(t) \equiv 0 \quad \text{vs.} \quad H_1 : \mu(t) \neq 0. \quad (1)$$

```
set.seed(111)
wavk.test(D$Landings ~ 1,
          factor.length = "adaptive.selection", ar.method = "yw")

##
## Trend test by Wang, Akritas, and Van Keilegom (bootstrap p-values)
##
## data: D$Landings
## WAVK test statistic = 5, adaptively selected window = 6, p-value = 0.002
## alternative hypothesis: trend is not of the form D$Landings ~ 1.

wavk.test(D$sqrtLandings ~ 1,
          factor.length = "adaptive.selection", ar.method = "yw")

##
## Trend test by Wang, Akritas, and Van Keilegom (bootstrap p-values)
##
## data: D$sqrtLandings
## WAVK test statistic = 3, adaptively selected window = 6, p-value = 0.006
## alternative hypothesis: trend is not of the form D$sqrtLandings ~ 1.

set.seed(222)
wavk.test(DO$AMO_DJFMA ~ 1,
          factor.length = "adaptive.selection", ar.method = "yw")

##
## Trend test by Wang, Akritas, and Van Keilegom (bootstrap p-values)
##
## data: DO$AMO_DJFMA
## WAVK test statistic = -0.4, adaptively selected window = 4, p-value = 0.8
## alternative hypothesis: trend is not of the form DO$AMO_DJFMA ~ 1.

wavk.test(DO$AMO_annual ~ 1,
          factor.length = "adaptive.selection", ar.method = "yw")

##
## Trend test by Wang, Akritas, and Van Keilegom (bootstrap p-values)
##
## data: DO$AMO_annual
## WAVK test statistic = -0.5, adaptively selected window = 4, p-value = 0.8
## alternative hypothesis: trend is not of the form DO$AMO_annual ~ 1.
```

```

par(mar = c(4, 4, 0.1, 1) + 0.1, mgp = c(3, 0.7, 0), mfrow = c(3, 1))
attach(D)
plot(x = Year, y = Landings, las = 1, type = "o", xlab = "", ylim = c(0, 5000))
points(x = Year[indNA], y = Landings[indNA], bg = COL["red"], pch = 21)
abline(v = TBend + 0.5, lty = 2, col = COL["green"])
text(x = c(1916, TBcenter[-1]), y = 4900, col = COL["green"], labels = names(TBcenter))
plot(x = Year, y = AMO_DJFMA, las = 1, type = "o", xlab = "")
plot(x = Year, y = NAO_DJFMA_PC, las = 1, type = "o")
detach(D)

```

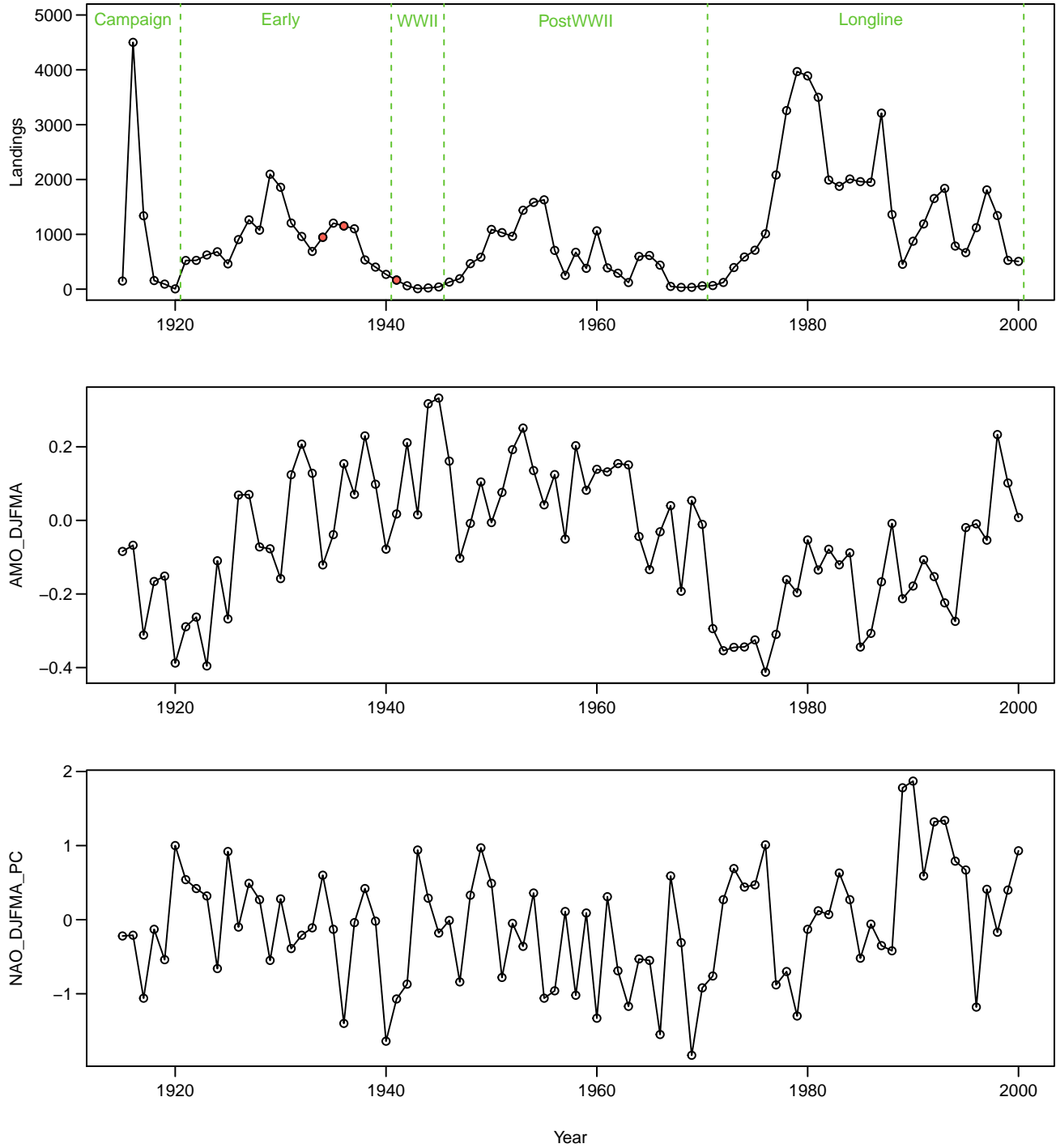


Figure 1: Plots of the main time series, 1915–2000. Filled circles represent filled-in (interpolated) values. Dashed vertical lines denote the time blocks.

```

set.seed(333)
wavk.test(DO$NAO_DJF_PC ~ 1,
           factor.length = "adaptive.selection", ar.method = "yw")

##
## Trend test by Wang, Akritas, and Van Keilegom (bootstrap p-values)
##
## data: DO$NAO_DJF_PC
## WAVK test statistic = 3, adaptively selected window = 4, p-value = 0.04
## alternative hypothesis: trend is not of the form DO$NAO_DJF_PC ~ 1.

wavk.test(DO$NAO_DJF_st ~ 1,
           factor.length = "adaptive.selection", ar.method = "yw")

##
## Trend test by Wang, Akritas, and Van Keilegom (bootstrap p-values)
##
## data: DO$NAO_DJF_st
## WAVK test statistic = 3, adaptively selected window = 4, p-value = 0.02
## alternative hypothesis: trend is not of the form DO$NAO_DJF_st ~ 1.

wavk.test(DO$NAO_DJFMA_PC ~ 1,
           factor.length = "adaptive.selection", ar.method = "yw")

##
## Trend test by Wang, Akritas, and Van Keilegom (bootstrap p-values)
##
## data: DO$NAO_DJFMA_PC
## WAVK test statistic = 3, adaptively selected window = 4, p-value = 0.02
## alternative hypothesis: trend is not of the form DO$NAO_DJFMA_PC ~ 1.

wavk.test(DO$NAO_DJFMA_st ~ 1,
           factor.length = "adaptive.selection", ar.method = "yw")

##
## Trend test by Wang, Akritas, and Van Keilegom (bootstrap p-values)
##
## data: DO$NAO_DJFMA_st
## WAVK test statistic = 2, adaptively selected window = 4, p-value = 0.1
## alternative hypothesis: trend is not of the form DO$NAO_DJFMA_st ~ 1.

```

It is reasonable that the test rejects the H_0 for the time series of landings because there are important time blocks of the fisheries that affect the average landings; to account for that we will use the time block variable in the regression models.

See some of the scatterplots in [Figure 2](#).

```
tmp = c(#grep("Time_block", names(D)),
  grep("14", names(D)),
  grep("Land", names(D)))
ggpairs(D[,tmp])
```

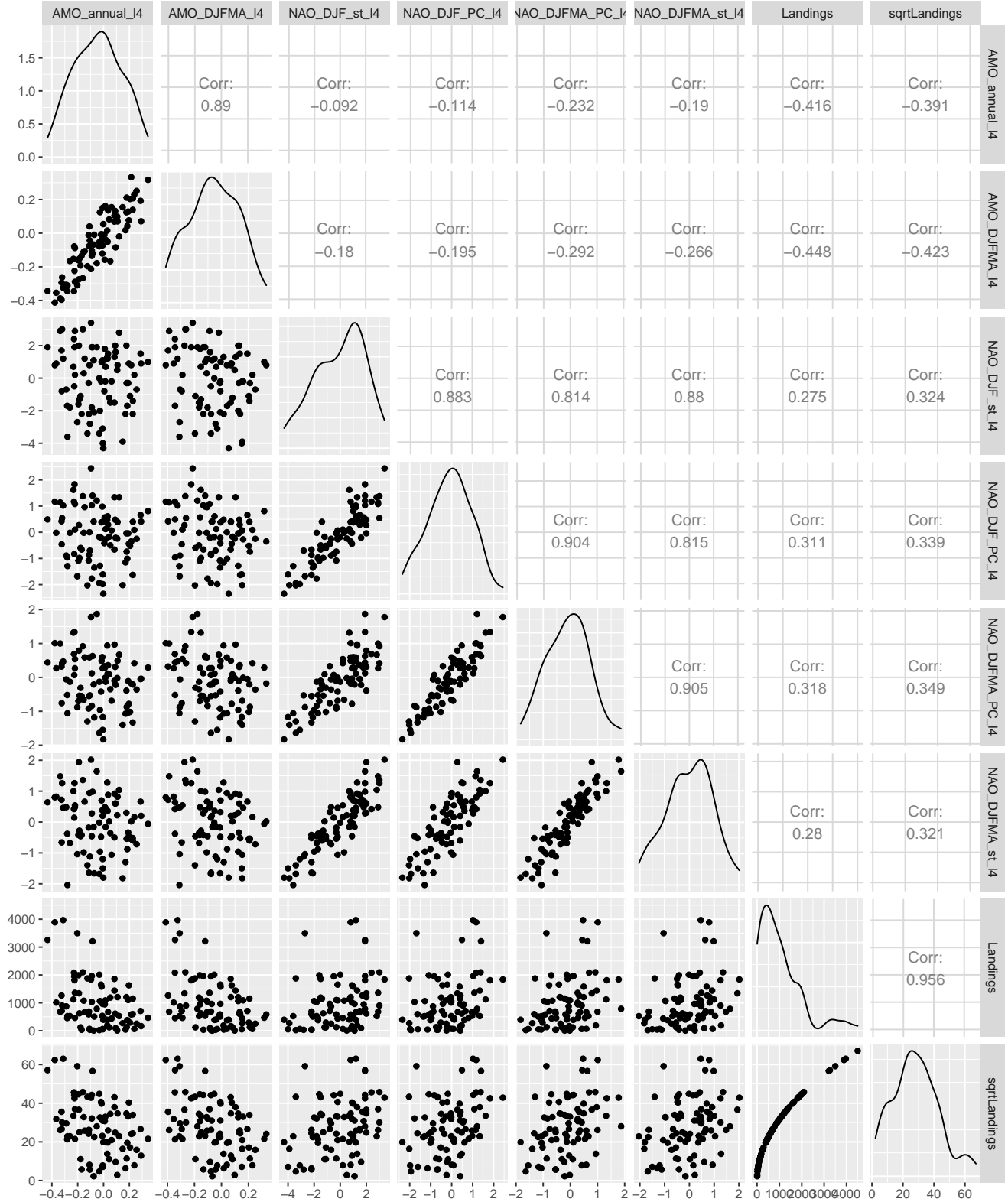


Figure 2: Matrix of scatterplots, univariate distributions (main diagonal), and pairwise correlations for the period of 1915–2000. Only some of the lagged variables are plotted for visibility.

3 Random forest

References about the methods: [Breiman \(2001\)](#); [Hastie et al \(2009\)](#); [Kursa and Rudnicki \(2010\)](#); [Wright and Ziegler \(2017\)](#).

Random forest inputs:

```
NTREE = 500
Nnode = 3
RESPONSE = "sqrtLandings"
RESPONSEprint = "sqrt(Landings)"
#Predictors:
n = names(D)
vnot = grep("Landings|Year", n, value = TRUE) #variables to exclude
predictors = sort(setdiff(n, vnot))
DATA = D
DATAnoNA = na.omit(DATA[,c(RESPONSE, predictors)])
```

Variable selection:

```
set.seed(444)
B = Boruta::Boruta(as.formula(paste(RESPONSE, ".", sep = " ~ ")),
  doTrace = 1, maxRuns = 5000,
  data = DATAnoNA)

print(B)

## Boruta performed 4492 iterations in 1.86 mins.
## 14 attributes confirmed important: AMO_annual_15, AMO_annual_16, AMO_annual_17, AMO_DJFMA_15,
## AMO_DJFMA_16 and 9 more;
## 35 attributes confirmed unimportant: AMO_annual, AMO_annual_11, AMO_annual_12, AMO_annual_13,
## AMO_annual_14 and 30 more;

st = attStats(B) # print(st)
# v = rownames(st)[st$decision != "Rejected"]
v = rownames(st)[st$decision == "Confirmed"]
```

The selected variables in alphabetic order (green in Figure 3):

```
sort(v)

## [1] "AMO_annual_15" "AMO_annual_16" "AMO_annual_17" "AMO_DJFMA_15" "AMO_DJFMA_16"
## [6] "AMO_DJFMA_17" "NAO_DJF_PC_11" "NAO_DJF_PC_13" "NAO_DJF_PC_14" "NAO_DJF_st_13"
## [11] "NAO_DJF_st_14" "NAO_DJFMA_st_14" "NAO_DJFMA_st_16" "Time_block"
```

That is, from the original 49 variables we selected 14 variables. Get the final random forest with those 14 variables:

```
set.seed(555)
DATAnoNA = na.omit(DATA[, c(RESPONSE, v)])
rf_allv = ranger(dependent.variable.name = RESPONSE, data = DATAnoNA,
  #importance = 'impurity_corrected',
  min.node.size = Nnode,
  respect.unordered.factors = 'partition',
  num.trees = NTREE)

print(rf_allv)

## Ranger result
##
## Call:
## ranger(dependent.variable.name = RESPONSE, data = DATAnoNA, min.node.size = Nnode,      respect.unordered.factors = "partition")
##
## Type:                                Regression
## Number of trees:                      500
## Sample size:                          79
## Number of independent variables:      14
## Mtry:                                  3
## Target node size:                      3
## Variable importance mode:              none
## Splitrule:                             variance
## OOB prediction error (MSE):            101
## R squared (OOB):                       0.486
```



```

par(mar = c(4, 9, 0.1, 1) + 0.1)
plot(B, horizontal = TRUE,
     colCode = COL[c("green", "yellow", "red", "blue")],
     las = 1, ylab = "", xlab = "Importance")

```

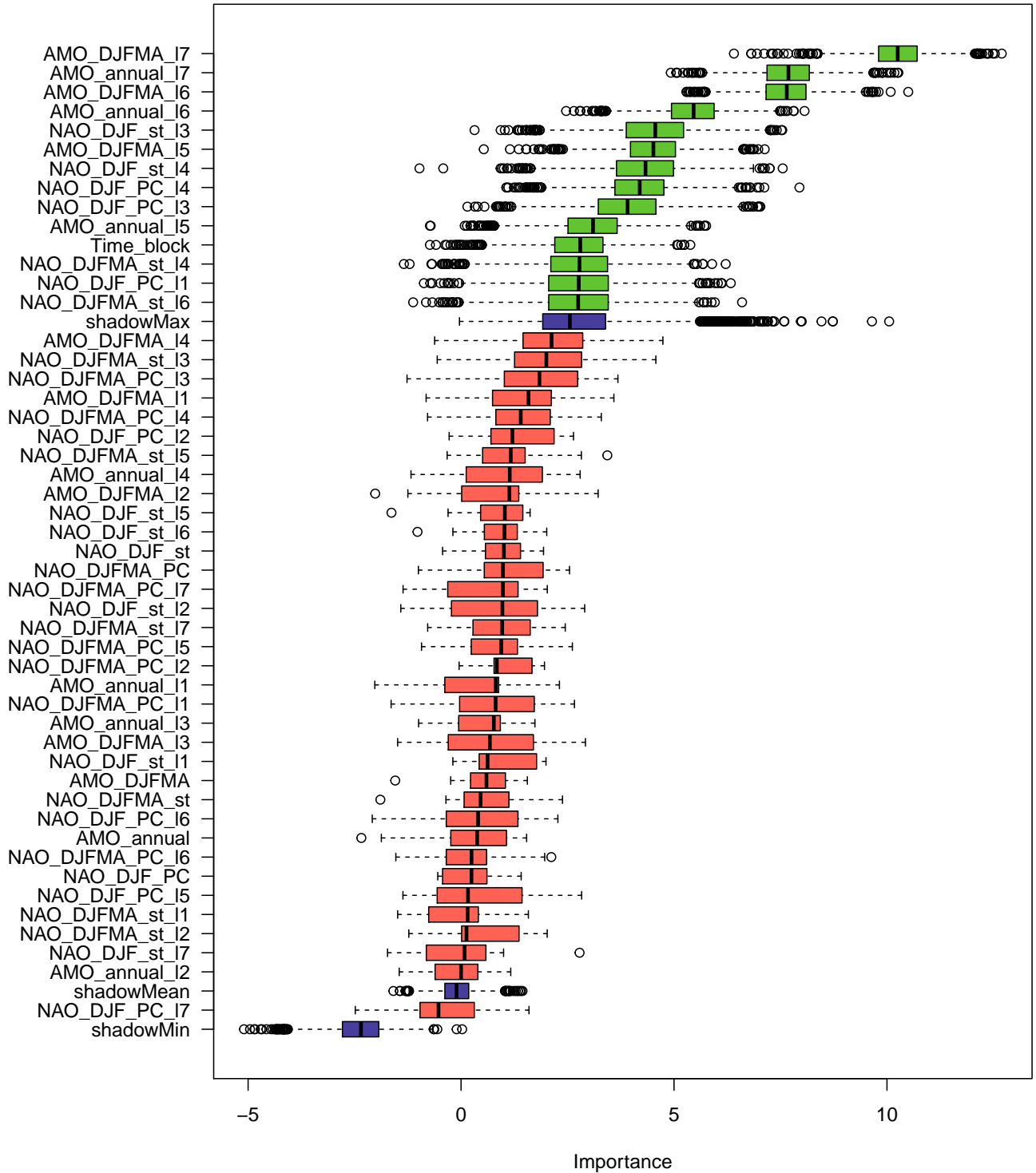


Figure 3: Boxplots of predictor importance over runs of the Boruta algorithm. Green boxes correspond to predictors confirmed as important; red boxes correspond to rejected predictors, and blue boxes combine importance information (min, mean, and max) from shadow predictors obtained by permuting the original ones.

```
par(mar = c(4, 4, 0.1, 1) + 0.1)
plot(rf_allv2, las = 1, main = "")
```

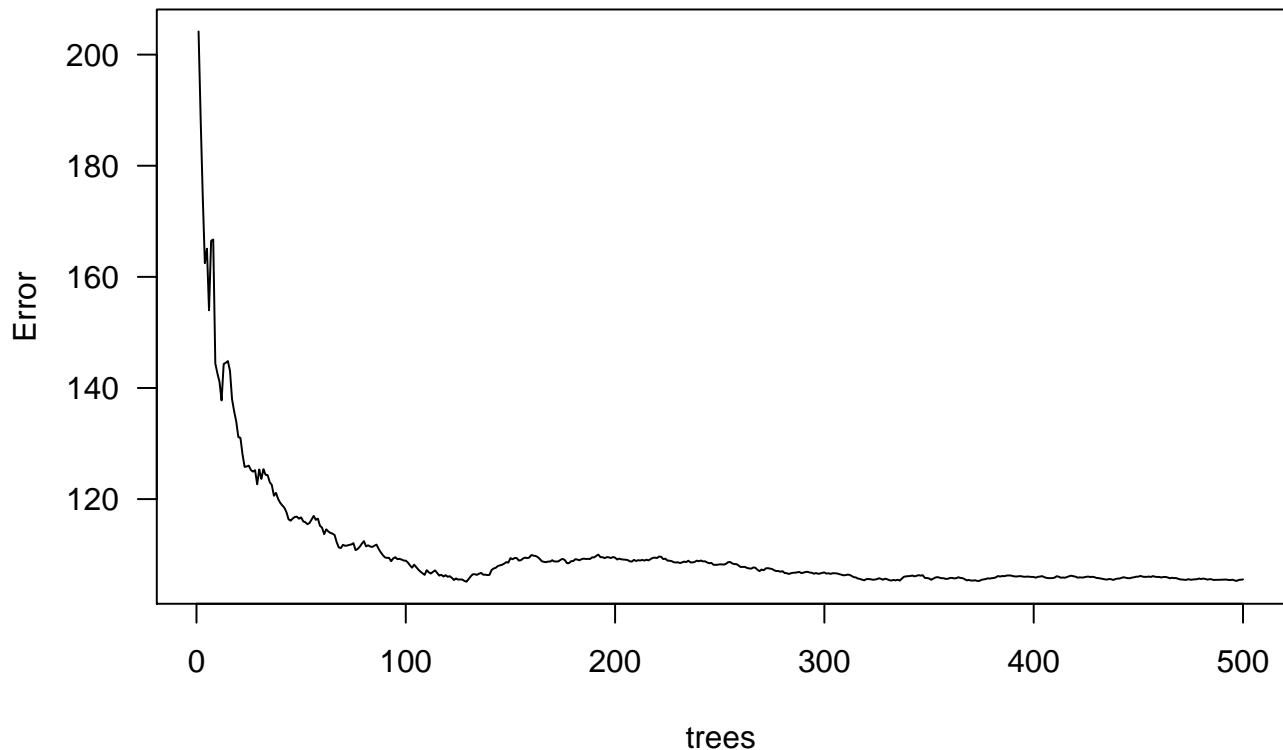


Figure 4: Random forest error based on the number of trees.

Non-adjusted R^2 :

```
# Not really useful for us
y = DATAnoNA[, RESPONSE]
e = y - predict(rf_allv, data = DATAnoNA)$predictions
1 - sum(e^2) / sum((y - mean(y))^2)

## [1] 0.923
```

Final random forest output from another package (see error plot in Figure 4 and partial dependence plots in Figure 5):

```
set.seed(666)
rf_allv2 = randomForest(y = DATAnoNA[,RESPONSE],
                        x = DATAnoNA[, v],
                        nodesize = rf_allv$min.node.size,
                        mtry = rf_allv$mtry,
                        ntree = rf_allv$num.trees)

print(rf_allv2)

##
## Call:
## randomForest(x = DATAnoNA[, v], y = DATAnoNA[, RESPONSE], ntree = rf_allv$num.trees, mtry = rf_allv$mtry, nodesize = rf_al
##           Type of random forest: regression
##           Number of trees: 500
##           No. of variables tried at each split: 3
##
##           Mean of squared residuals: 106
##           % Var explained: 45.7
```

```

RF = rf_allv2
preds = sort(v)
par(mfrow = c(ceiling(length(preds)/4), 4))
par(bty = "L", mar = c(5, 4, 1, 1) + 0.1, mgp = c(2, 0.7, 0))
for(i in 1:length(preds)) {
  l = ifelse(is.factor(DATANoNA[,preds[i]]), 2, 1) #rotate labels for categorical predictor
  x = ifelse(is.factor(DATANoNA[,preds[i]]), "", preds[i]) #x-label
  partialPlot(RF, pred.data = DATANoNA, x.var = preds[i],
             las = l, xlab = x, ylab = "", main = "", xpd = F
             ,ylim = c(25, 34) #fix y-scale across plots
             )
  mtext(RESPONSEprint, side = 2, line = 2, cex = 0.7)
  mtext(paste("(", letters[i], ")", sep = ""), side = 3, line = 0.1, cex = 0.8, adj = -0.37)
}

```

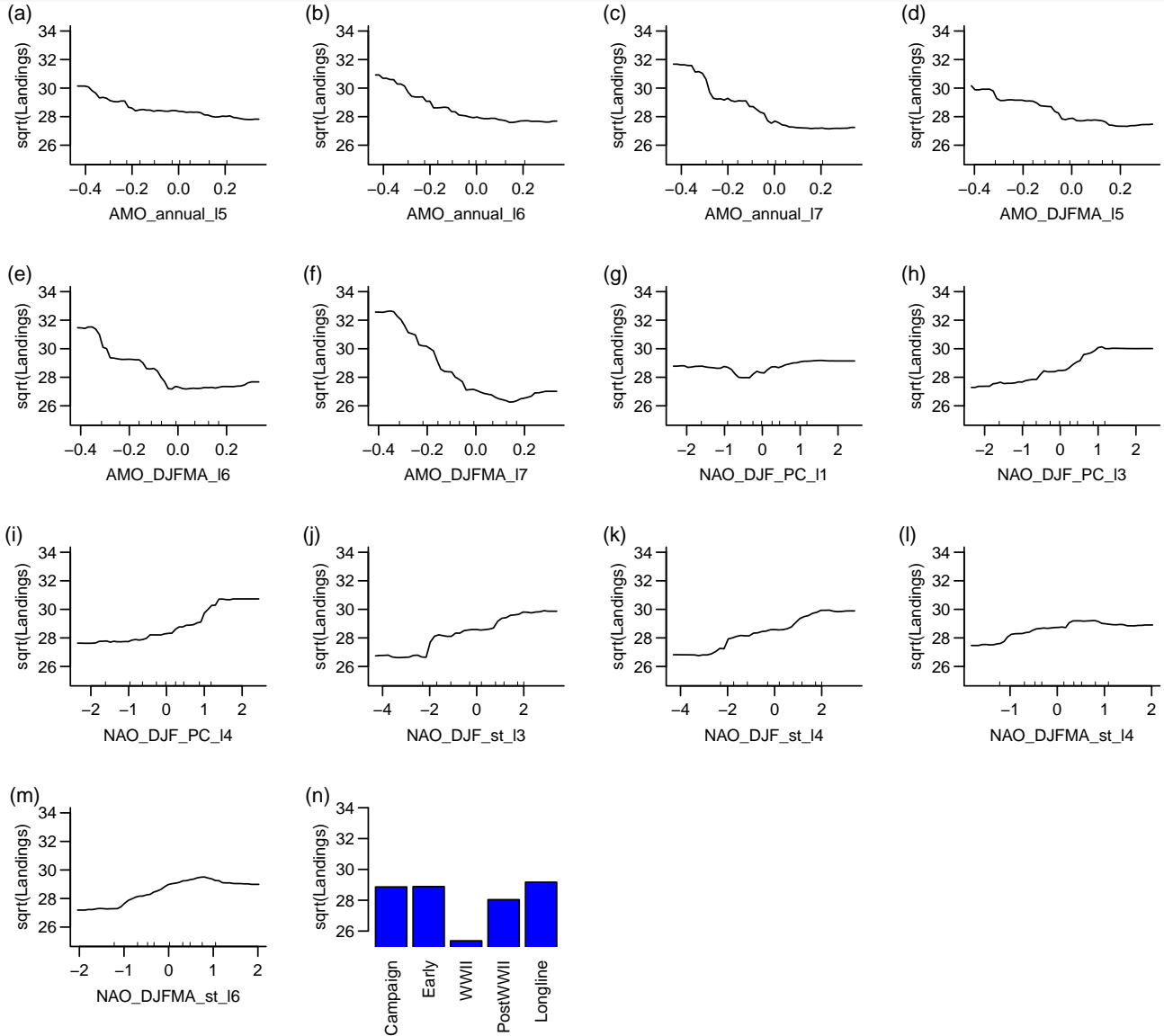


Figure 5: Partial dependence plots for the random forest for $\sqrt{\text{Landings}}$ of golden tilefish. The plots are in alphabetic order by the name of the explanatory variable. The inner tickmarks on the horizontal axis denote deciles of the respective explanatory variable.

3.1 Cross-validation

Scheme:

1. Train model using data up to the middle of the PostWWII period.
2. Forecast until the end of that period.
3. Repeat the above steps for the period Longline.

```
set.seed(777)
CVperiods = c("PostWWII", "Longline")
PRED = as.list(rep(NA, length(CVperiods)))
yrs = as.numeric(rownames(DATANoNA))
for (cv in seq_along(CVperiods)){ # cv=1
  indtrain = yrs[1]:TBcenter[CVperiods[cv]]
  indtest = (TBcenter[CVperiods[cv]] + 1):TBend[CVperiods[cv]]
  rf_cv = ranger(dependent.variable.name = RESPONSE,
                 data = DATANoNA[as.character(indtrain),],
                 #importance = 'impurity_corrected',
                 min.node.size = Nnode,
                 respect.unordered.factors = 'partition',
                 num.trees = NTREE)
  obs = DATANoNA[as.character(indtest), RESPONSE] #observed in the testing set
  pred = predict(rf_cv, data = DATANoNA[as.character(indtest),])$predictions #predictions
  PRED[[cv]] = list(obs = obs, pred = pred)
}
```

Count number of test samples per cross-validation run

```
(tmp = sapply(PRED, function(x) length(x[[1]])))
## [1] 12 14
```

and total

```
sum(tmp)
## [1] 26
```

Prediction mean absolute error (PMAE) and prediction root mean square error (PRMSE).

```
err = lapply(PRED, function(x) x$obs - x$pred)
err = unlist(err)
```

PMAE

```
mean(abs(err))
## [1] 8.04
```

PRMSE

```
sqrt(mean(err^2))
## [1] 10.1
```

4 Generalized additive modeling (GAM)

References about the methods: [Wood \(2006\)](#); [Zuur et al \(2009\)](#).

Full model:

```
K = 5
gam_0 = gamfit = mgcv::gam(sqrtLandings ~
  Time_block
  + s(AMO_annual_15, k = K)
  + s(AMO_annual_16, k = K)
  + s(AMO_annual_17, k = K)
  + s(AMO_DJFMA_15, k = K)
  + s(AMO_DJFMA_16, k = K)
  + s(AMO_DJFMA_17, k = K)
  + s(NAO_DJF_PC_11, k = K)
  + s(NAO_DJF_PC_13, k = K)
  + s(NAO_DJF_PC_14, k = K)
  + s(NAO_DJF_st_13, k = K)
  + s(NAO_DJF_st_14, k = K)
  + s(NAO_DJFMA_st_14, k = K)
  + s(NAO_DJFMA_st_16, k = K)
  , select = TRUE
  , bs = "cr"
  , method = "REML"
  , data = DATAoNA)

summary(gamfit)

##
## Family: gaussian
## Link function: identity
##
## Formula:
## sqrtLandings ~ Time_block + s(AMO_annual_15, k = K) + s(AMO_annual_16,
## k = K) + s(AMO_annual_17, k = K) + s(AMO_DJFMA_15, k = K) +
## s(AMO_DJFMA_16, k = K) + s(AMO_DJFMA_17, k = K) + s(NAO_DJF_PC_11,
## k = K) + s(NAO_DJF_PC_13, k = K) + s(NAO_DJF_PC_14, k = K) +
## s(NAO_DJF_st_13, k = K) + s(NAO_DJF_st_14, k = K) + s(NAO_DJFMA_st_14,
## k = K) + s(NAO_DJFMA_st_16, k = K)
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    26.905      1.785   15.07 < 2e-16 ***
## Time_blockWWII -14.424      4.071   -3.54 0.00076 ***
## Time_blockPostWWII 0.168      2.861    0.06 0.95349
## Time_blockLongline 6.600      2.425    2.72 0.00843 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##              edf Ref.df    F p-value
## s(AMO_annual_15)  9.81e-05     4 0.00 0.6757
## s(AMO_annual_16)  1.02e-04     4 0.00 0.7289
## s(AMO_annual_17)  1.17e-04     4 0.00 0.7546
## s(AMO_DJFMA_15)  1.05e-04     4 0.00 0.5613
## s(AMO_DJFMA_16)  1.89e+00     4 2.86 0.0020 **
## s(AMO_DJFMA_17)  2.16e+00     4 4.61 9.1e-05 ***
## s(NAO_DJF_PC_11)  1.69e-04     4 0.00 0.4299
## s(NAO_DJF_PC_13)  1.72e+00     4 2.84 0.0013 **
## s(NAO_DJF_PC_14)  1.40e+00     4 1.05 0.0248 *
## s(NAO_DJF_st_13)  1.16e-04     4 0.00 0.6117
## s(NAO_DJF_st_14)  1.14e+00     4 0.50 0.0980 .
## s(NAO_DJFMA_st_14) 1.46e+00     4 1.04 0.0310 *
## s(NAO_DJFMA_st_16) 3.09e+00     4 6.03 4.2e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) = 0.737 Deviance explained = 79%
## -REML = 275.8 Scale est. = 51.863 n = 79
```

```
concurvity(gamfit)[3,-1]
```

```
##      s(AMO_annual_15)      s(AMO_annual_16)      s(AMO_annual_17)      s(AMO_DJFMA_15)      s(AMO_DJFMA_16)
##      0.915              0.959              0.940              0.943              0.947
##      s(AMO_DJFMA_17)      s(NAO_DJF_PC_11)      s(NAO_DJF_PC_13)      s(NAO_DJF_PC_14)      s(NAO_DJF_st_13)
##      0.919              0.639              0.916              0.919              0.901
##      s(NAO_DJF_st_14)      s(NAO_DJFMA_st_14)      s(NAO_DJFMA_st_16)
##      0.944              0.901              0.598
```

Spearman correlations:

```
Hmisc::rccorr(as.matrix(DATANoNA[,c(RESPONSE, v[-length(v)])]),
              type = "spearman")[[1]]
```

```
##      sqrtLandings AMO_annual_15 AMO_annual_16 AMO_annual_17 AMO_DJFMA_15 AMO_DJFMA_16 AMO_DJFMA_17
## sqrtLandings      1.000      -0.440      -0.540      -0.574      -0.485      -0.558      -0.601
## AMO_annual_15      -0.440      1.000      0.733      0.560      0.891      0.613      0.498
## AMO_annual_16      -0.540      0.733      1.000      0.747      0.823      0.895      0.639
## AMO_annual_17      -0.574      0.560      0.747      1.000      0.574      0.822      0.885
## AMO_DJFMA_15      -0.485      0.891      0.823      0.574      1.000      0.682      0.560
## AMO_DJFMA_16      -0.558      0.613      0.895      0.822      0.682      1.000      0.690
## AMO_DJFMA_17      -0.601      0.498      0.639      0.885      0.560      0.690      1.000
## NAO_DJF_PC_11      0.200      -0.140      -0.212      -0.118      -0.225      -0.230      -0.104
## NAO_DJF_PC_13      0.369      -0.162      -0.232      -0.172      -0.267      -0.263      -0.227
## NAO_DJF_PC_14      0.355      -0.109      -0.167      -0.230      -0.112      -0.268      -0.263
## NAO_DJF_st_13      0.356      -0.232      -0.221      -0.166      -0.280      -0.236      -0.203
## NAO_DJF_st_14      0.393      -0.089      -0.226      -0.243      -0.123      -0.281      -0.246
## NAO_DJFMA_st_14     0.360      -0.170      -0.252      -0.266      -0.148      -0.293      -0.248
## NAO_DJFMA_st_16     0.273      -0.104      -0.241      -0.133      -0.179      -0.294      -0.126
##      NAO_DJF_PC_11 NAO_DJF_PC_13 NAO_DJF_PC_14 NAO_DJF_st_13 NAO_DJF_st_14 NAO_DJFMA_st_14
## sqrtLandings      0.2005      0.369      0.3554      0.3557      0.3928      0.360
## AMO_annual_15      -0.1397      -0.162      -0.1094      -0.2317      -0.0890      -0.170
## AMO_annual_16      -0.2122      -0.232      -0.1668      -0.2212      -0.2263      -0.252
## AMO_annual_17      -0.1183      -0.172      -0.2301      -0.1657      -0.2432      -0.266
## AMO_DJFMA_15      -0.2251      -0.267      -0.1125      -0.2805      -0.1232      -0.148
## AMO_DJFMA_16      -0.2297      -0.263      -0.2681      -0.2363      -0.2811      -0.293
## AMO_DJFMA_17      -0.1044      -0.227      -0.2629      -0.2029      -0.2464      -0.248
## NAO_DJF_PC_11      1.0000      0.118      0.0147      0.0903      0.0891      0.118
## NAO_DJF_PC_13      0.1178      1.000      0.1984      0.8868      0.2420      0.228
## NAO_DJF_PC_14      0.0147      0.198      1.0000      0.1898      0.8844      0.811
## NAO_DJF_st_13      0.0903      0.887      0.1898      1.0000      0.2231      0.219
## NAO_DJF_st_14      0.0891      0.242      0.8844      0.2231      1.0000      0.861
## NAO_DJFMA_st_14     0.1183      0.228      0.8112      0.2185      0.8611      1.000
## NAO_DJFMA_st_16     0.0907      0.121      0.0916      0.0917      0.0897      0.148
##      NAO_DJFMA_st_16
## sqrtLandings      0.2731
## AMO_annual_15      -0.1036
## AMO_annual_16      -0.2414
## AMO_annual_17      -0.1334
## AMO_DJFMA_15      -0.1791
## AMO_DJFMA_16      -0.2940
## AMO_DJFMA_17      -0.1260
## NAO_DJF_PC_11      0.0907
## NAO_DJF_PC_13      0.1206
## NAO_DJF_PC_14      0.0916
## NAO_DJF_st_13      0.0917
## NAO_DJF_st_14      0.0897
## NAO_DJFMA_st_14     0.1478
## NAO_DJFMA_st_16     1.0000
```

High concurvity. Most of the concurvity is due to duplicated information (e.g., annual and winter form of a climate index). From each pair of concurvity terms, select such term to remove, which is less correlated with the response (use Spearman's correlation coefficient, which allows to account for possibly non-linear monotonic relationship). Based on this approach, we favor the winter form of AMO (keep in the model), and remove the annual AMO version; favor PC-based/station-based NAO and remove station-based/PC-based NAO.

Further variable selection is done by stepwise removal of terms with the largest non-significant

p -value, until all remaining terms are statistically significant (also see `?mgcv::gam.selection`).

To additionally incorporate autocorrelation into the model (to get rid of autocorrelation in the model residuals), fit a generalized additive mixed model (GAMM) with autocorrelation structure specified as AR(1) – autoregression of the order 1 – and the autoregression coefficient ϕ estimated from the data. See the autocorrelation function (ACF) plot of the residuals from the final (reduced) model in Figure 7.

Reduced model:

```
#cc removed because of concurvity
#1# after that, removed first because of non-significance
#1# after that, switched to gamm and AR1 structure and continue removal
set.seed(111111)
K = 5
gam_1 = gamfit = mgcv::gamm(sqrtLandings ~
  ##1 Time_block
  #cc + s(AMO_annual_l5, k = K)
  #cc + s(AMO_annual_l6, k = K)
  #cc + s(AMO_annual_l7, k = K)
  #2# + s(AMO_DJFMA_l5, k = K)
  ##3 + s(AMO_DJFMA_l6, k = K)
  + s(AMO_DJFMA_l7, k = K)
  #1# + s(NAO_DJF_PC_l1, k = K)
  + s(NAO_DJF_PC_l3, k = K)
  #cc + s(NAO_DJF_PC_l4, k = K)
  #cc + s(NAO_DJF_st_l3, k = K)
  + s(NAO_DJF_st_l4, k = K)
  #cc + s(NAO_DJFMA_st_l4, k = K)
  ##2 + s(NAO_DJFMA_st_l6, k = K)
  , select = TRUE
  , bs = "cr"
  , method = "REML"
  , correlation = corAR1()
  , data = DATAoNA)
```

GAM part:

```
summary(gamfit$gam)

##
## Family: gaussian
## Link function: identity
##
## Formula:
## sqrtLandings ~ +s(AMO_DJFMA_l7, k = K) + s(NAO_DJF_PC_l3, k = K) +
## s(NAO_DJF_st_l4, k = K)
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    27.97      4.55     6.14 3.7e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##              edf Ref.df    F p-value
## s(AMO_DJFMA_l7) 1.00   1.00 5.62 0.0204 *
## s(NAO_DJF_PC_l3) 1.00   1.00 6.46 0.0131 *
## s(NAO_DJF_st_l4) 1.98   1.98 4.80 0.0085 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) = 0.254
## Scale est. = 153.82    n = 79
```

LME part:

```
summary(gamfit$lme)

## Linear mixed-effects model fit by REML
## Data: strip.offset(mf)
```

```
## AIC BIC logLik
## 530 551 -256
##
## Random effects:
## Formula: ~Xr - 1 | g
## Structure: pdIdnot
##      Xr1      Xr2      Xr3
## StdDev: 0.00182 0.00182 0.00182
##
## Formula: ~Xr.0 - 1 | g.0 %in% g
## Structure: pdIdnot
##      Xr.01      Xr.02      Xr.03
## StdDev: 0.000995 0.000995 0.000995
##
## Formula: ~Xr.1 - 1 | g.1 %in% g.0 %in% g
## Structure: pdIdnot
##      Xr.11 Xr.12 Xr.13 Residual
## StdDev: 6.46 6.46 6.46 12.4
##
## Correlation Structure: AR(1)
## Formula: ~1 | g/g.0/g.1
## Parameter estimate(s):
## Phi
## 0.848
## Fixed effects: y ~ X - 1
##              Value Std.Error DF t-value p-value
## X(Intercept) 27.97      4.55 75    6.14 0.0000
## Xs(AMO_DJFMA_17)Fx1 -2.36      0.99 75   -2.37 0.0204
## Xs(NAO_DJF_PC_13)Fx1 1.72      0.68 75    2.54 0.0131
## Xs(NAO_DJF_st_14)Fx1 2.19      1.38 75    1.58 0.1185
## Correlation:
##              X(Int) X(AMO_ X(NAO_DJF_P
## Xs(AMO_DJFMA_17)Fx1 0.015
## Xs(NAO_DJF_PC_13)Fx1 0.010 0.004
## Xs(NAO_DJF_st_14)Fx1 0.003 0.008 0.244
##
## Standardized Within-Group Residuals:
##      Min      Q1      Med      Q3      Max
## -1.5449 -0.5554 -0.0331 0.7020 2.6664
##
## Number of Observations: 79
## Number of Groups:
##              g              g.0 %in% g g.1 %in% g.0 %in% g
##              1              1              1
```

Normality of residuals:

```
shapiro.test(residuals(gamfit$lme, type = "normalized"))

##
## Shapiro-Wilk normality test
##
## data:  residuals(gamfit$lme, type = "normalized")
## W = 1, p-value = 0.7

shapiro.test(residuals(gamfit$lme, type = "response"))

##
## Shapiro-Wilk normality test
##
## data:  residuals(gamfit$lme, type = "response")
## W = 1, p-value = 0.06
```

Run more checks:

```
gamfit = gamfit$gam
concurvity(gamfit)

##              para s(AMO_DJFMA_17) s(NAO_DJF_PC_13) s(NAO_DJF_st_14)
## worst      2.39e-28      0.258      0.205      0.226
## observed 2.39e-28      0.144      0.145      0.167
## estimate 2.39e-28      0.154      0.138      0.158
```



```

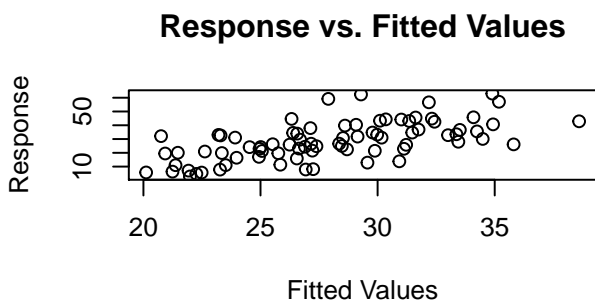
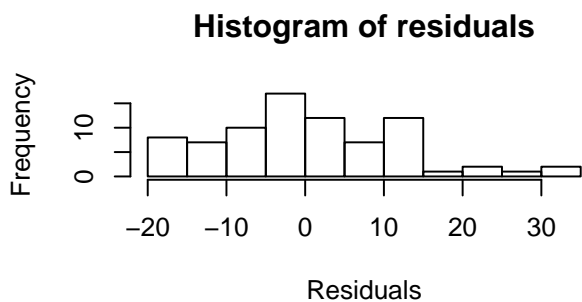
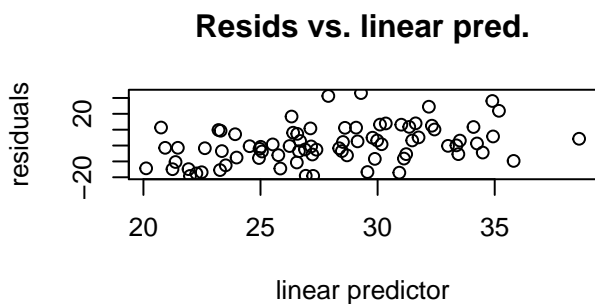
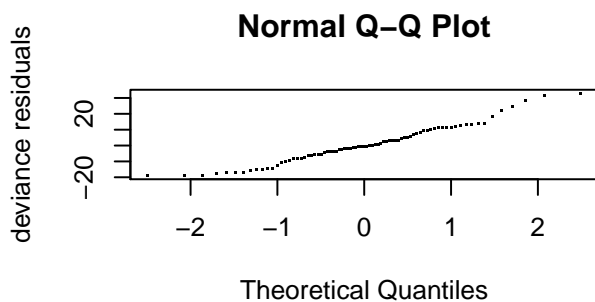
concurvity(gamfit, full = FALSE)$estimate

##               para s(AMO_DJFMA_17) s(NAO_DJF_PC_13) s(NAO_DJF_st_14)
## para           1.00e+00      2.09e-31      6.83e-31      4.45e-32
## s(AMO_DJFMA_17)  1.34e-28      1.00e+00      7.50e-02      9.96e-02
## s(NAO_DJF_PC_13) 5.08e-29      7.09e-02      1.00e+00      7.30e-02
## s(NAO_DJF_st_14) 2.08e-29      9.54e-02      6.89e-02      1.00e+00

gam.check(gamfit)

##
## 'gamm' based fit - care required with interpretation.
## Checks based on working residuals may be misleading.
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##               k'   edf k-index p-value
## s(AMO_DJFMA_17) 4.00 1.00   0.73  0.01 **
## s(NAO_DJF_PC_13) 4.00 1.00   0.99  0.41
## s(NAO_DJF_st_14) 4.00 1.98   1.12  0.85
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```



```
ggpairs(DATAnoNA[,c(v, RESPONSE)])
```

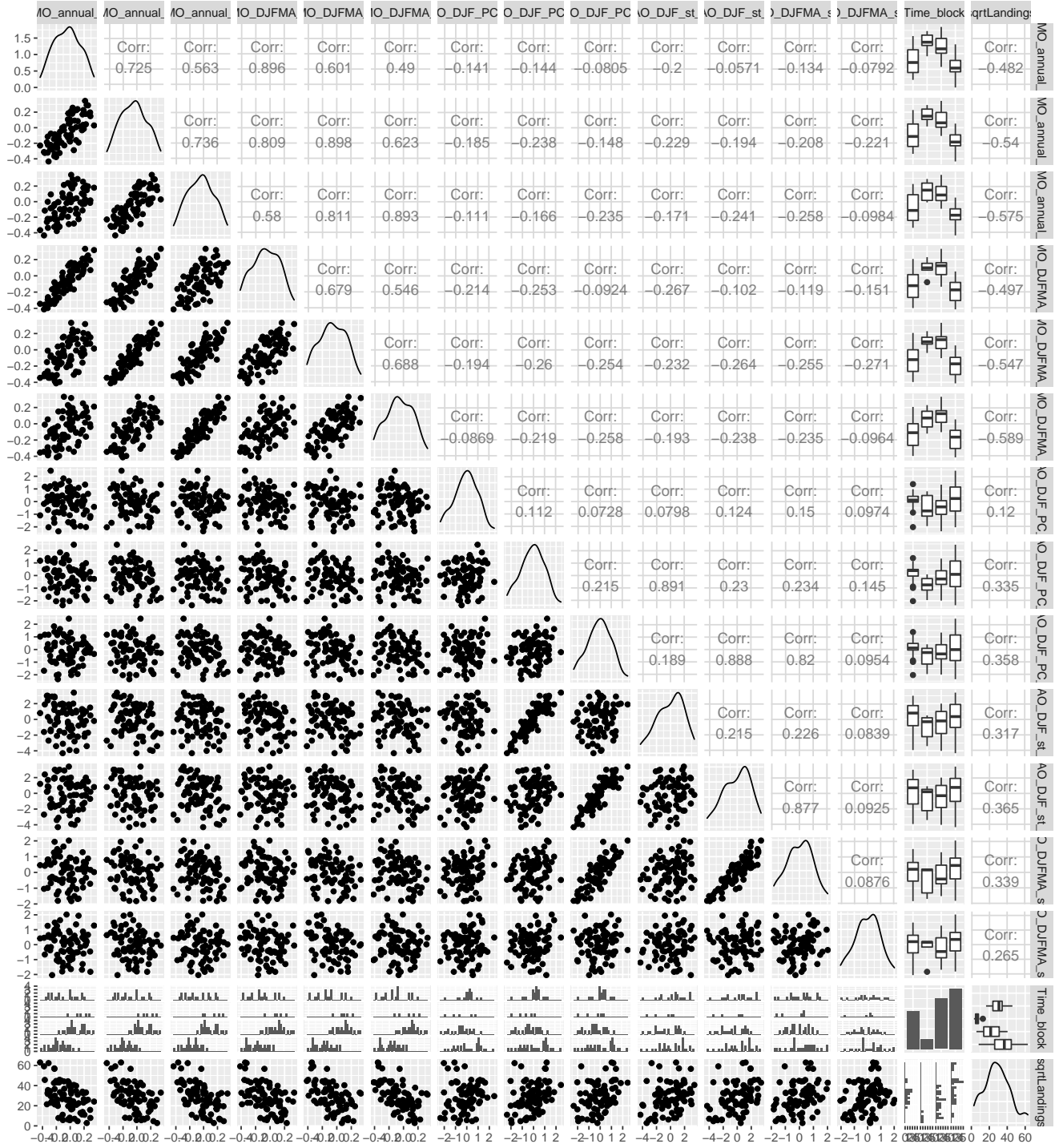


Figure 6: Scatterplot matrix of sqrt(Landings) and variables selected in the random forest.

```
par(mfrow = c(1, 2))
plot.ts(residuals(gamfit$lme, type = "normalized"), las = 1)
abline(h = 0, col = COL["green"], lty = 2)
acf(residuals(gamfit$lme, type = "normalized"), las = 1, main = "Standardized residual ACF")
```

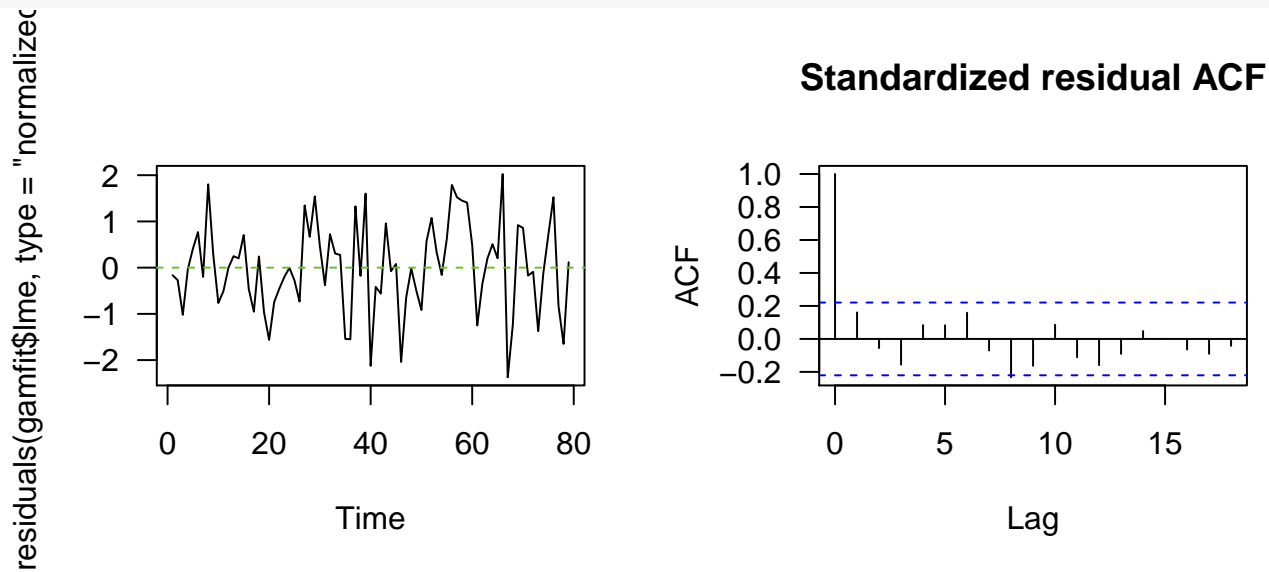


Figure 7: Time series plot and autocorrelation function of residuals of the reduced GAMM.

```
par(mfrow = c(1, 3))
plot(gamfit, las = 1)
```

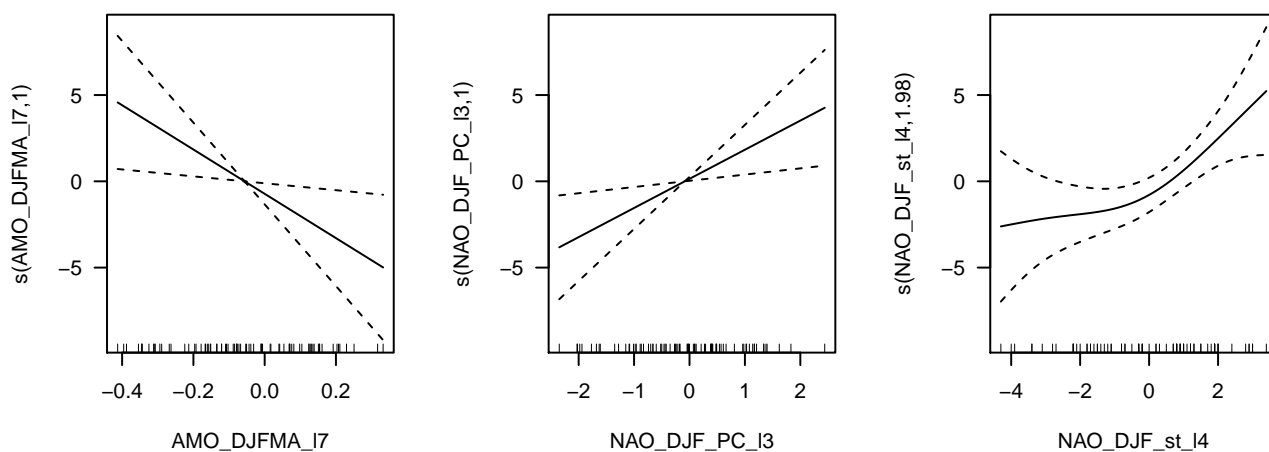


Figure 8: Smoothed terms of the final GAMM for the $\sqrt{\text{Landings}}$.

4.1 Cross-validation

The scheme is the same as the one applied to random forests (see p. 12).

```
set.seed(222222)
CVperiods = c("PostWWII", "Longline")
PRED = as.list(rep(NA, length(CVperiods)))
yrs = as.numeric(rownames(DATAnoNA))
for (cv in seq_along(CVperiods)){ # cv=1
  indtrain = yrs[1]:TBcenter[CVperiods[cv]]
  indtest = (TBcenter[CVperiods[cv]] + 1):TBend[CVperiods[cv]]
  gam_cv = mgcv::gamm(sqrtLandings ~
    + s(AMO_DJFMA_17, k = K)
    + s(NAO_DJF_PC_13, k = K)
    + s(NAO_DJF_st_14, k = K)
    , select = TRUE
    , bs = "cr" #cr/cs/cc
    , method = "REML"
    , correlation = corAR1()
    , control = list(maxIter = 1000)
    , data = DATAnoNA[as.character(indtrain),])
  obs = DATAnoNA[as.character(indtest), RESPONSE] #observed in the testing set
  # Predictions:
  ## make a prediction, with random effects zero
  pred_gam = predict(gam_cv$gam, newdata = DATAnoNA[as.character(indtest),])
  phi = gam_cv$lme$modelStruct$corStruct
  phi = coef(phi, unconstrained = FALSE) #AR(1) coefficient
  ## extract autocorrelated errors in the training period (we'll need the last observed error)
  fit_gam = fitted(gam_cv$lme)
  e_train = DATAnoNA[as.character(indtrain), RESPONSE] - fit_gam
  #acf(e_train) #ar(e_train, aic = FALSE, order.max = 1) #check that is similar to phi
  e = obs - pred_gam #autocorrelated error in the testing set
  pred_gamm = pred_gam + phi * c(e_train[length(e_train)], e[-length(e)])
  PRED[[cv]] = list(obs = obs, pred = pred_gamm)
}
```

Prediction mean absolute error (PMAE) and prediction root mean square error (PRMSE).

```
err = lapply(PRED, function(x) x$obs - x$pred)
err = unlist(err)
```

PMAE

```
mean(abs(err))
```

```
## [1] 6.64
```

PRMSE

```
sqrt(mean(err^2))
```

```
## [1] 8.43
```

```
save.image(paste0("./dataderived/tmp_", Sys.Date(), ".RData"))
```

References

- Breiman L (2001) Random forests. *Machine Learning* 45(1):5–32, DOI [10.1023/A:1010933404324](https://doi.org/10.1023/A:1010933404324)
- Fisher JAD, Frank KT, Petrie B, Leggett WC (2014) Life on the edge: environmental determinants of tilefish (*Lopholatilus chamaeleonticeps*) abundance since its virtual extinction in 1882. *ICES Journal of Marine Science* 71(9):2371–2378, DOI [10.1093/icesjms/fsu053](https://doi.org/10.1093/icesjms/fsu053)
- Hastie TJ, Tibshirani RJ, Friedman JH (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd edn. Springer, New York, DOI [10.1007/978-0-387-84858-7](https://doi.org/10.1007/978-0-387-84858-7)
- Kursa MB, Rudnicki WR (2010) Feature selection with the Boruta package. *Journal of Statistical Software* 36(11):1–13
- Lyubchich V, Gel YR, El-Shaarawi A (2013) On detecting non-monotonic trends in environmental time series: a fusion of local regression and bootstrap. *Environmetrics* 24(4):209–226, DOI [10.1002/env.2212](https://doi.org/10.1002/env.2212)
- Wood SN (2006) *Generalized Additive Models: An Introduction with R*. Chapman and Hall/CRC, New York
- Wright MN, Ziegler A (2017) ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software* 77(1):1–17, DOI [10.18637/jss.v077.i01](https://doi.org/10.18637/jss.v077.i01)
- Zuur A, Ieno EN, Walker NJ, Saveliev AA, Smith GM (2009) *Mixed Effects Models and Extensions in Ecology with R*. Springer, New York, DOI [10.1007/978-0-387-87458-6](https://doi.org/10.1007/978-0-387-87458-6)