Corrigendum to "Guidance for decisions using the Vector Autoregressive Spatio-Temporal (VAST) package in stock, ecosystem, habitat and climate assessments" [Fish. Res. 210 (February) 2019, 143–161]

James T. Thorson

PW8-RD03

Received: 7/19/2021

Fisheries Research 215 (2019) 143–150

Contents lists available at ScienceDirect

Fisheries Research

journal homepage: www.elsevier.com/locate/fishres

Corrigendum

Corrigendum to "Guidance for decisions using the Vector Autoregressive Spatio-Temporal (VAST) package in stock, ecosystem, habitat and climate assessments" [Fish. Res. 210 (February) 2019, 143–161]

James T. Thorson

Habitat and Ecosystem Process Research program, Alaska Fisheries Science Center, National Marine Fisheries Service, NOAA, Seattle, WA, USA

The author regrets that they did not notice and correct a mistake made by the journal during copy-editing, which resulted in the following omissions in the originally published version of the manuscript. Specifically, the originally published version was missing R code associated with footnotes 3, 18, 19, and 20. The complete footnotes should read:

Footnote 3.

Footnote 3.
For example, users can read an overview of the package using:
install.packages("devtools")
devtools::install_github("james-thorson/VAST")
library(VAST)
?VAST

Footnote 18.

Footnote 18.
These decisions are implemented using the following inputs (with # indicating comments in R):
Region = "Eastern_Bering_Sea" # Spatial domain for extrapolation
n_x = 1000 # Number of knots (defines spatial resolution
FieldConfig = c("Omega1"=1, "Epsilon1"=1, "Omega2"=1, "Epsilon2"=1) # Number of spatio-temporal factors
RhoConfig = c("Beta1"=0, "Beta2"=0, "Epsilon1"=0, "Epsilon2"=0) # Temporal structure
ObsModel = c(2,1) # Distribution for data, and link-function for linear predictors
Options = c("Calculate_Range"=TRUE, "Calculate_effective_area"=TRUE) # Which derived quantities
OverdispersionConfig = c("Eta1"=0, "Eta2"=0) # Number of factors for overdispersion

DOI of original article: https://doi.org/10.1016/j.fishres.2018.10.013
E-mail address: James.Thorson@noaa.gov.

https://doi.org/10.1016/j.fishres.2019.02.016

Available online 26 March 2019
0165-7836/ Published by Elsevier B.V.

Footnote 19.

```
Footnote 19.
These decisions are implemented using the following inputs (with # indicating comments in
R):
Region = "Gulf_of_Alaska"  # Spatial domain for extrapolation
n_x = 500  # Number of knots (defines spatial resolution
FieldConfig = c("Omega1"=2, "Epsilon1"=2, "Omega2"=0, "Epsilon2"=0)  # Number of
spatio-temporal factors
RhoConfig = c("Beta1"=3, "Beta2"=3, "Epsilon1"=4, "Epsilon2"=0)  # Temporal structure
ObsModel = c(2,1)  # Distribution for data, and link-function for linear predictors
Options =  c("Calculate_Range"=FALSE, "Calculate_effective_area"=FALSE)  # Which
derived quantities
OverdispersionConfig = c("Eta1"=0, "Eta2"=0)  # Number of factors for overdispersion
```

Footnote 20.

```
Footnote 20.
These decisions are implemented using the following inputs (with # indicating comments in
R):
Region = "California_current"  # Spatial domain for extrapolation
n_x = 100   # Number of knots (defines spatial resolution
FieldConfig = c("Omega1"=5, "Epsilon1"=5, "Omega2"=5, "Epsilon2"=5)  # Number of
spatio-temporal factors
RhoConfig = c("Beta1"=2, "Beta2"=2, "Epsilon1"=4, "Epsilon2"=4)  # Temporal structure
ObsModel = c(2,1)  # Distribution for data, and link-function for linear predictors
Options =  c("Calculate_Range"=FALSE, "Calculate_effective_area"=FALSE)  # Which
derived quantities
OverdispersionConfig = c("Eta1"=5, "Eta2"=5)  # Number of factors for overdispersion
```

In addition, the version printed by Fisheries Research was also missing the Supporting Information, which contains code to replicate the three case studies. This R code is now included below:

```
devtools::install_github("james-thorson/VAST", ref="2.0.1")
devtools::install_github("james-thorson/FishData")

# Change directory to local destination
RootDir = "C:/Users/James.Thorson/Desktop/UW Hideaway/Collaborations/2018 -- VAST
decision tree/"

library(TMB)
library(VAST)

# Global memory
Version = get_latest_version(package="VAST")  # Checks for latest version
Date = Sys.Date()

# Choose example
Example = c("Index", "Ordination", "Allocation")[2]

# Settings for each example
if( Example == "Index" ){
  Method = "Mesh"
  n_x = 1000   # Specify number of stations (a.k.a. "knots")
  Kmeans_Config = list( "randomseed"=1, "nstart"=100, "iter.max"=1e3 )
  FieldConfig = c("Omega1"=1, "Epsilon1"=1, "Omega2"=1, "Epsilon2"=1)
  RhoConfig = c("Beta1"=0, "Beta2"=0, "Epsilon1"=0, "Epsilon2"=0)
  OverdispersionConfig = c("Eta1"=0, "Eta2"=0)
  ObsModel = c(2,0)
  Options =  c("Calculate_Range"=TRUE, "Calculate_effective_area"=TRUE)
  strata.limits <- data.frame('STRATA'="All_areas")
  Survey = Region = "Eastern_Bering_Sea"
  Species_set = "Atheresthes stomias"
  Vars2Correct = c("Index_cyl", "effective_area_cyl", "mean_Z_cym")
}
```

```
if( Example == "Ordination" ){
  Method = "Mesh"
  n_x = 500   # Specify number of stations (a.k.a. "knots")
  Kmeans_Config = list( "randomseed"=1, "nstart"=100, "iter.max"=1e3 )
  FieldConfig = c("Omega1"=2, "Epsilon1"=2, "Omega2"=0, "Epsilon2"=0)
  RhoConfig = c("Beta1"=3, "Beta2"=3, "Epsilon1"=4, "Epsilon2"=0)
  OverdispersionConfig = c("Eta1"=0, "Eta2"=0)
  ObsModel = c(2,1)
  Options =  c("Calculate_Range"=FALSE, "Calculate_effective_area"=FALSE)
  strata.limits <- data.frame('STRATA'="All_areas")
  Survey = "GOABTS"
  Region = "Gulf_of_Alaska"
  Species_set = 8
  Vars2Correct = c()
}


if( Example == "Allocation" ){
  Method = "Mesh"
  n_x = 100   # Specify number of stations (a.k.a. "knots")
  Kmeans_Config = list( "randomseed"=1, "nstart"=100, "iter.max"=1e3 )
  FieldConfig = c("Omega1"=5, "Epsilon1"=5, "Omega2"=5, "Epsilon2"=5)
  RhoConfig = c("Beta1"=2, "Beta2"=2, "Epsilon1"=4, "Epsilon2"=4)
  OverdispersionConfig = c("Eta1"=5, "Eta2"=5)
  ObsModel = c(2,1)
  Options =  c("Calculate_Range"=FALSE, "Calculate_effective_area"=FALSE)
  strata.limits <- data.frame(
    'STRATA' = c("Coastwide","CA","OR","WA"),
    'north_border' = c(49.0, 42.0, 46.0, 49.0),
    'south_border' = c(32.0, 32.0, 42.0, 46.0),
    'shallow_border' = c(55, 55, 55, 55),
    'deep_border' = c(1280, 1280, 1280, 1280)
  )
  Survey = "WCGBTS"
  Region = "California_current"
  Species_set = c("Microstomus pacificus","Anoplopoma fimbria","Eopsetta
jordani","Sebastolobus alascanus","Ophiodon elongatus")
  Vars2Correct = c()
}


# Create directory for results
DateDir = paste0(RootDir,Date,'/')
RunDir = paste0(DateDir,Example,ifelse(Example=="Index",paste0("_",Species_set),""),'/')
dir.create(RunDir,recursive=TRUE)

# Save a record of settings
Record =
list("Version"=Version,"Method"=Method,"grid_size_km"=50,"n_x"=n_x,"FieldConfig"=Fie
ldConfig,"RhoConfig"=RhoConfig,"OverdispersionConfig"=OverdispersionConfig,"ObsMod
el"=ObsModel,"Kmeans_Config"=Kmeans_Config,"Region"=Region,"Survey"=Survey,"Spe
cies_set"=Species_set,"strata.limits"=strata.limits)
save( Record, file=file.path(RunDir,"Record.RData"))
capture.output( Record, file=paste0(RunDir,"Record.txt"))

# Download public data
DF = FishData::download_catch_rates(survey=Survey, species_set=Species_set,
localdir=RunDir)
Data_Geostat = data.frame( "spp"=DF[,"Sci"], "Year"=DF[,"Year"],
"Catch_KG"=DF[,"Wt"], "AreaSwept_km2"=0.01, "Vessel"=0, "Lat"=DF[,"Lat"],
"Lon"=DF[,"Long"] )
PredTF_i = rep(0, nrow(Data_Geostat))

# Formatting changes to Data_Geostat
if( Example == "Allocation" ){
  Data_Geostat[,'Vessel'] = factor( paste(DF[,'Year'],DF[,'Vessel'],sep="_") )
  Data_Geostat = rbind( Data_Geostat[1,], Data_Geostat[1,], Data_Geostat )
  Data_Geostat[1:2,'Year'] = max(Data_Geostat[,'Year']) + 1:2
  PredTF_i = c( 1,1, PredTF_i )
}
```

```
# Create data frame of spatial domain
Extrapolation_List = make_extrapolation_info( Region=Region, strata.limits=strata.limits )

# Determine location of knots
if( Example == "Index" ){
  Spatial_List = make_spatial_info( grid_size_km=50, n_x=n_x, Method=Method,
Lon=Data_Geostat[,'Lon'], Lat=Data_Geostat[,'Lat'],
LON_intensity=Extrapolation_List$Data_Extrap[,'Lon'],
LAT_intensity=Extrapolation_List$Data_Extrap[,'Lat'],
Extrapolation_List=Extrapolation_List, randomseed=Kmeans_Config[["randomseed"]],
nstart=Kmeans_Config[["nstart"]], iter.max=Kmeans_Config[["iter.max"]], DirPath=RunDir,
Save_Results=TRUE )
}else{
  Spatial_List = make_spatial_info( grid_size_km=50, n_x=n_x, Method=Method,
Lon=Data_Geostat[,'Lon'], Lat=Data_Geostat[,'Lat'], Extrapolation_List=Extrapolation_List,
randomseed=Kmeans_Config[["randomseed"]], nstart=Kmeans_Config[["nstart"]],
iter.max=Kmeans_Config[["iter.max"]], DirPath=RunDir, Save_Results=TRUE )
}


 # Add knots to Data_Geostat
 Data_Geostat = cbind( Data_Geostat, "knot_i"=Spatial_List$knot_i )

 # Assemble data
 TmbData = Data_Fn("Version"=Version, "FieldConfig"=FieldConfig,
 "OverdispersionConfig"=OverdispersionConfig, "RhoConfig"=RhoConfig,
 "ObsModel"=ObsModel, "c_i"=as.numeric(Data_Geostat[,'spp'])-1,
 "b_i"=Data_Geostat[,'Catch_KG'], "a_i"=Data_Geostat[,'AreaSwept_km2'],
 "v_i"=as.numeric(Data_Geostat[,'Vessel'])-1, "s_i"=Data_Geostat[,'knot_i']-1,
 "t_i"=Data_Geostat[,'Year'], "a_xl"=Spatial_List$a_xl, "MeshList"=Spatial_List$MeshList,
 "GridList"=Spatial_List$GridList, "Method"=Spatial_List$Method, "Options"=Options,
 "PredTF_i"=PredTF_i )

 # Build model
 TmbList = Build_TMB_Fn("TmbData"=TmbData, "RunDir"=RunDir, "Version"=Version,
 "RhoConfig"=RhoConfig, "loc_x"=Spatial_List$loc_x, "Method"=Method )
 Obj = TmbList[["Obj"]]

 # Estimate parameters
 Opt = TMBhelper::Optimize( obj=Obj, startpar=Obj$par+0.01*runif(length(Obj$par)),
 lower=TmbList[["Lower"]], upper=TmbList[["Upper"]], getsd=TRUE, newtonsteps=1,
 savedir=RunDir, bias.correct=ifelse(length(Vars2Correct)>0,TRUE,FALSE),
 bias.correct.control=list(sd=FALSE, split=NULL, nsplit=1, vars_to_correct=Vars2Correct) )

 # Save results
 Report = Obj$report()

 Save = list("Opt"=Opt, "Report"=Report, "ParHat"=Obj$env$parList(Opt$par),
 "TmbData"=TmbData)
 save(Save, file=paste0(RunDir,"Save.RData"))

 # Get region-specific settings for plots
 MapDetails_List = make_map_info( "Region"=Region,
 "NN_Extrap"=Spatial_List$PolygonList$NN_Extrap,
 "Extrapolation_List"=Extrapolation_List )
 # Decide which years to plot
 Year_Set = seq(min(Data_Geostat[,'Year']),max(Data_Geostat[,'Year']))
 Years2Include = which( Year_Set %in% sort(unique(Data_Geostat[,'Year'])))
```

```
if( Example == "Index" ){
  Index = plot_biomass_index( DirName=RunDir, TmbData=TmbData,
Sdreport=Opt[["SD"]], Year_Set=Year_Set, Years2Include=Years2Include,
strata_names=strata.limits[,1], use_biascorr=TRUE,
category_names=levels(Data_Geostat[,'spp']) )
  COG = plot_range_index(Report=Report, TmbData=TmbData, Sdreport=Opt[["SD"]],
Znames=colnames(TmbData$Z_xm), PlotDir=RunDir,
category_names=levels(Data_Geostat[,'spp']), Year_Set=Year_Set)
  plot_maps(plot_set=c(3), MappingDetails=MapDetails_List[["MappingDetails"]],
Report=Report, Sdreport=Opt$SD, PlotDF=MapDetails_List[["PlotDF"]],
MapSizeRatio=MapDetails_List[["MapSizeRatio"]], Xlim=MapDetails_List[["Xlim"]],
Ylim=MapDetails_List[["Ylim"]], FileName=RunDir, Year_Set=Year_Set,
Years2Include=Years2Include, Rotate=MapDetails_List[["Rotate"]],
Cex=MapDetails_List[["Cex"]], Legend=MapDetails_List[["Legend"]],
zone=MapDetails_List[["Zone"]], mar=c(0,0,2,0), oma=c(3.5,3.5,0,0), cex=1.8,
category_names=levels(Data_Geostat[,'spp']))

  ThorsonUtilities::save_fig( paste0(DateDir,"Index_summary"), width=3, height=6 )
    par( mfrow=c(3,1), mar=c(0.5,4,0.5,0.5), oma=c(3,0,0,0), mgp=c(1.75,0.25,0), tck=-0.02 )
    # Index
    Ybounds =
(Index$Table[,'Estimate_metric_tons']%o%c(1,1)+Index$Table[,'SD_mt']%o%c(-1,1))
    plot_lines( x=Index$Table[,'Year'], y=Index$Table[,'Estimate_metric_tons']/1000,
xaxt="n", fn=plot, ybounds=Ybounds/1000, ylim=c(0,max(Ybounds)/1000), main="",
xlab="", ylab="Biomass\n(1000 metric tons)" )
    legend( "topleft", legend="(A)", bty="n")
    # Area occupied
    Ybounds =
(COG$EffectiveArea_Table[,'EffectiveArea']%o%c(1,1)+COG$EffectiveArea_Table[,'SE']%
o%c(-1,1))
    plot_lines( x=Index$Table[,'Year'],
y=exp(COG$EffectiveArea_Table[,'EffectiveArea'])/1000, xaxt="n", fn=plot,
ybounds=exp(Ybounds)/1000, ylim=range(exp(Ybounds))/1000, main="", xlab="",
ylab="Effective area occupied\n(1000 square-kilometers)" )
    legend( "topleft", legend="(B)", bty="n")
    # Northward COG
    COG_Table = COG$COG_Table[ which(COG$COG_Table[,'m']==2 ), ]
    Ybounds = (COG_Table[,'COG_hat']%o%c(1,1)+COG_Table[,'SE']%o%c(-1,1))
    plot_lines( x=Index$Table[,'Year'], y=COG_Table[,'COG_hat'], xaxt="n", fn=plot,
ybounds=Ybounds, ylim=range(Ybounds), main="", xlab="", ylab="Northward center-of-
gravity\n(kilometers north of equator)" )
    legend( "topleft", legend="(C)", bty="n")
    axis(1)
  mtext( side=1, text="Year", outer=TRUE, line=1.5 )
 dev.off()
}
```

```
if( Example == "Ordination" ){
  Index = plot_biomass_index( DirName=RunDir, TmbData=TmbData,
Sdreport=Opt[["SD"]], Year_Set=Year_Set, Years2Include=Years2Include,
strata_names=strata.limits[,1], use_biascorr=TRUE,
category_names=levels(Data_Geostat[,'spp']) )
  Factor_list = Plot_factors( Report=Report, ParHat=Obj$env$parList(), Data=TmbData,
RotationMethod="Varimax", SD=Opt$SD, mapdetails_list=MapDetails_List,
Year_Set=Year_Set, category_names=levels(DF[,'Sci']), plotdir=RunDir )
  plot_anisotropy( FileName=paste0(RunDir,"Aniso.png"), Report=Report,
TmbData=TmbData )

  # Names
  ThorsonUtilities::save_fig( paste0(DateDir,"Ordination_summary"), width=3, height=6 )
    par( mfrow=c(2,1), mar=c(0.5,1.5,2,0.5), oma=c(2,1,0,0), mgp=c(1.75,0.25,0), tck=-0.02 )
    Range = max(abs( c(as.vector(Factor_list$Rotated_loadings$Omega1),
as.vector(Factor_list$Rotated_loadings$Epsilon1)) ))
    # Spatial
    plot( 1, type="n", xlim=c(-1,1)*Range, ylim=c(-1,1)*Range, xaxt="n", main="Spatial",
xlab="", ylab="")
    text( x=Factor_list$Rotated_loadings$Omega1[,1],
y=Factor_list$Rotated_loadings$Omega1[,2],
labels=1:nrow(Factor_list$Rotated_loadings$Omega1) )
    abline( h=0, lty="dotted" )
    abline( v=0, lty="dotted" )
    # Spatio-temporal
    plot( 1, type="n", xlim=c(-1,1)*Range, ylim=c(-1,1)*Range, xaxt="n", main="Spatio-
temporal", xlab="", ylab="")
    text( x=Factor_list$Rotated_loadings$Epsilon1[,1],
y=Factor_list$Rotated_loadings$Epsilon1[,2],
labels=1:nrow(Factor_list$Rotated_loadings$Epsilon1) )
    abline( h=0, lty="dotted" )
    abline( v=0, lty="dotted" )
    axis(1)
    mtext( side=1:2, text=c("Factor 1", "Factor 2"), outer=TRUE, line=c(1,0) )
  dev.off()
  cbind( 1:nrow(Factor_list$Rotated_loadings$Epsilon1),
rownames(Factor_list$Rotated_loadings$Epsilon1) )


  # Omega
  Psi_rot = Factor_list$Rotated_factors[["Omega1"]]
  Psi_rot = ifelse( abs(Psi_rot)>4, sign(Psi_rot)*4, Psi_rot )
  zlim = c(-1,1) * max(abs(Psi_rot[1:TmbData$n_x,,]))
  Col =
colorRampPalette(colors=c("darkblue","blue","lightblue","lightgreen","yellow","orange","red
"),alpha=0.2)
  ThorsonUtilities::save_fig( paste0(DateDir,"Ordination_Omega"),
width=MapDetails_List$MapSizeRatio['Width(in)']*2+0.5,
height=MapDetails_List$MapSizeRatio['Height(in)']*1+1 )
    par( mfcol=c(1,2), mar=c(0.5,0.5,0.5,0.5), oma=c(2.5,4.5,2,0), mgp=c(1.75,0.25,0), tck=-
0.02 )
    for( colI in 1:2 ){
      PlotMap_Fn( MappingDetails=list("worldHires",NULL), zlim=zlim,
Mat=Psi_rot[,,1][,colI,drop=FALSE], PlotDF=MapDetails_List$PlotDF,
MapSizeRatio=MapDetails_List$MapSizeRatio, Xlim=MapDetails_List$Xlim,
Ylim=MapDetails_List$Ylim, FileName="", Year_Set="", Rescale=FALSE,
Rotate=MapDetails_List$Rotate, Format="", Res=MapDetails_List$Res,
zone=Extrapolation_List$zone, Cex=0.15, textmargin="", add=TRUE, pch=15,
Legend=list("use"=FALSE), plot_legend_fig=FALSE )
      mtext( side=3, text=paste0("Factor ",colI), line=0.5, font=2 )
      if( colI==1 ) axis(2)
      axis(1)
      if( colI==2 ){
        FishStatsUtils:::smallPlot( FishStatsUtils:::Heatmap_Legend(colvec=Col(50),
heatrange=zlim, dopar=FALSE), x=MapDetails_List$Legend$x,
y=MapDetails_List$Legend$y, mar=c(0,0,0,0), mgp=c(2,0.5,0), tck=-0.2, font=2 )  #
      }
    }
    mtext( side=1:2, text=c("Longitude","Latitude"), outer=TRUE, line=c(1,3) )
  dev.off()
```

```
  # Epsilon
  Psi_rot = Factor_list$Rotated_factors[["Epsilon1"]]
  Psi_rot = ifelse( abs(Psi_rot)>4, sign(Psi_rot)*4, Psi_rot )
  zlim = c(-1,1) * max(abs(Psi_rot[1:TmbData$n_x,]))
  Col =
colorRampPalette(colors=c("darkblue","blue","lightblue","lightgreen","yellow","orange","red
"),alpha=0.2)
  ThorsonUtilities::save_fig( paste0(DateDir,"Ordination_Epsilon"),
width=MapDetails_List$MapSizeRatio['Width(in)']*2+0.5,
height=MapDetails_List$MapSizeRatio['Height(in)']*5+1 )
    par( mfrow=c(5,2), mar=c(0.5,0.5,0.5,0.5), oma=c(2.5,3,2,1.5), mgp=c(1.75,0.25,0), tck=-
0.02 )
    for( rowI in 1:5 ){
    for( colI in 1:2 ){
      tI = c(1,10,18,26,34)[rowI]
      PlotMap_Fn( MappingDetails=list("worldHires",NULL), zlim=zlim,
Mat=Psi_rot[,,tI][,colI,drop=FALSE], PlotDF=MapDetails_List$PlotDF,
MapSizeRatio=MapDetails_List$MapSizeRatio, Xlim=MapDetails_List$Xlim,
Ylim=MapDetails_List$Ylim, FileName="", Year_Set="", Rescale=FALSE,
Rotate=MapDetails_List$Rotate, Format="", Res=MapDetails_List$Res,
zone=Extrapolation_List$zone, Cex=0.2, textmargin="", add=TRUE, pch=15,
Legend=list("use"=FALSE), plot_legend_fig=FALSE )
      if( colI==2 ) mtext( side=4, text=Year_Set[tI], line=0.5 )
      if( colI==1 ) axis(2)
      if( rowI==5 ) axis(1)
      if( rowI==1 ) mtext( side=3, text=paste0("Factor ",colI), line=0.5, font=2 )
      if( colI==2 & rowI==5 ){
        FishStatsUtils:::smallPlot( FishStatsUtils:::Heatmap_Legend(colvec=Col(50),
heatrange=zlim, dopar=FALSE), x=MapDetails_List$Legend$x,
y=MapDetails_List$Legend$y, mar=c(0,0,0,0), mgp=c(2,0.5,0), tck=-0.2, font=2 )  #
      }
    }}
    mtext( side=1:2, text=c("Longitude","Latitude"), outer=TRUE, line=c(1,1) )
  dev.off()
}
```

```
if( Example == "Allocation" ){
  Index = plot_biomass_index( DirName=RunDir, TmbData=TmbData,
Sdreport=Opt[["SD"]], Year_Set=Year_Set, Years2Include=Years2Include,
strata_names=strata.limits[,1], use_biascorr=TRUE,
category_names=levels(Data_Geostat[,'spp']) )

  # Plot
  Index_ctl = Index$Index_ctl[,,2:4,]
  log_Index_ctl = Index$log_Index_ctl[,,2:4,]
  Index_ctl = Index_ctl / aperm( outer(apply(Index_ctl, MARGIN=c(1,2,4), FUN=sum),
rep(1,3)), c(1,2,4,3))
  interval_width = 1
  category_names = sapply( levels(Data_Geostat[,'spp']),
FUN=function(char){paste(strsplit(char,"_")[[1]],collapse=" ")} )
  ThorsonUtilities::save_fig( file=paste0(DateDir,"Allocation_summary"), width=2*2,
height=3*2 )
    par( mar=c(2,0,1,0.5), mgp=c(2,0.5,0), tck=-0.02, yaxs="i", oma=c(1,4,0,0), mfrow=c(3,2),
xaxs="i" )
    for( cI in 1:TmbData$n_c ){
      # Calculate y-axis limits
      Ylim = c(0,1)
      # Plot stuff
      plot(1, type="n", xlim=range(Year_Set), ylim=Ylim, xlab="", ylab="",
main=ifelse(TmbData$n_c>1,category_names[cI],""), yaxt="n" )
      polygon( x=c(2015.5,2018,2018,2015.5), y=c(-1,-1,2,2), col=rgb(0,0,0,0.1), border=NA )
      for(l in 1:(TmbData$n_l-1)){                                                    #
+seq(-0.1,0.1,length=TmbData$n_l)[l]
        plot_lines( y=Index_ctl[cI,Years2Include,l,'Estimate'], x=Year_Set[Years2Include],
ybounds=(Index_ctl[cI,Years2Include,l,'Estimate']%o%c(1,1))*exp(log_Index_ctl[cI,Years2I
nclude,l,'Std. Error']%o%c(-interval_width,interval_width)), type="l",
col=rainbow(TmbData[['n_l']]-1)[l], col_bounds=rainbow(TmbData[['n_l']]-1,alpha=0.2)[l],
ylim=Ylim, bounds_type="shading")
      }
      if( cI==TmbData$n_c ) legend( "top", bty="n", fill=rainbow(TmbData[['n_l']]-1),
legend=as.character(strata.limits[2:4,1]), ncol=3 )
      #abline( v=2015.5, lty="dotted" )
      if( cI %in% c(1,3,5) ) axis(2)
    }


    mtext( side=1:2, text=c("Year","Poportion of coastwide abundance by state"),
outer=TRUE, line=c(0,2) )
  dev.off()

  plot_anisotropy( FileName=paste0(RunDir,"Aniso.png"), Report=Report,
TmbData=TmbData )
  plot_lines(plot_set=c(3), MappingDetails=MapDetails_List[["MappingDetails"]],
Report=Report, Sdreport=Opt$SD, PlotDF=MapDetails_List[["PlotDF"]],
MapSizeRatio=MapDetails_List[["MapSizeRatio"]], Xlim=MapDetails_List[["Xlim"]],
Ylim=MapDetails_List[["Ylim"]], FileName=RunDir, Year_Set=Year_Set,
Years2Include=Years2Include, Rotate=MapDetails_List[["Rotate"]],
Cex=MapDetails_List[["Cex"]], Legend=MapDetails_List[["Legend"]],
zone=MapDetails_List[["Zone"]], mar=c(0,0,2,0), oma=c(3.5,3.5,0,0), cex=1.8,
category_names=levels(Data_Geostat[,'spp']))
  Cov_List = Summarize_Covariance( Report=Report, ParHat=Obj$env$parList(),
Data=TmbData, SD=Opt$SD, plot_cor=FALSE,
category_names=levels(Data_Geostat[,'spp']), plotdir=RunDir, plotTF=FieldConfig,
mgp=c(2,0.5,0), tck=-0.02, oma=c(0,5,2,2) )
  Plot_Overdispersion( filename1=paste0(RunDir,"Overdispersion"),
filename2=paste0(RunDir,"Overdispersion--panel"), Data=TmbData, ParHat=ParHat,
Report=Report, ControlList1=list("Width"=5, "Height"=10, "Res"=200, "Units"='in'),
ControlList2=list("Width"=TmbData$n_c, "Height"=TmbData$n_c, "Res"=200, "Units"='in')
)
  Plot_factors( Report=Report, ParHat=Obj$env$parList(), Data=TmbData, SD=Opt$SD,
mapdetails_list=MapDetails_List, Year_Set=Year_Set, category_names=levels(DF[,'Sci']),
plotdir=RunDir )
}
```

The author would like to apologise for any inconvenience caused.