Example script for SpatialDeltaGLMM for spatio-temporal analysis of catch-rate data

James Thorson

October 10, 2016

Contents

1 Overview

This tutorial will walk through a simple example of how to use `SpatialDeltaGLMM` for estimating abundance indices, distribution shifts, and range expansion.

2 Getting started

First, we install necessary packages. We also have to install TMB as appropriate for the operating system (see directions elsewhere).

```
devtools::install_github("nwfsc-assess/geostatistical_delta-GLMM")
devtools::install_github("james-thorson/utilities")
```

Next load libraries.

```
library(TMB)              # Can instead load library(TMBdebug)
library(SpatialDeltaGLMM)
```

3 Settings

First chose an example data set for this script, as archived with package

```
Data_Set = c("Chatham_rise_hake", "Iceland_cod", "WCGBTS_canary",
    "GSL_american_plaice", "BC_pacific_cod", "EBS_pollock",
    "GOA_Pcod", "GOA_pollock", "GB_spring_haddock",
    "GB_fall_haddock", "SAWC_jacopever", "Aleutian_islands_POP")[4]
```

Next use latest version for CPP code

```
Version = "geo_index_v4b"
```

3.1 Spatial settings

The following settings define the spatial resolution for the model, and whether to use a grid or mesh approximation

```
Method = c("Grid", "Mesh")[2]
grid_size_km = 25
n_x = c(100, 250, 500, 1000, 2000)[1] # Number of stations
Kmeans_Config = list( "randomseed"=1, "nstart"=100, "iter.max"=1e3 )
```

3.2 Model settings

The following settings define whether to include spatial and spatio-temporal variation, whether its autocorrelated, and whether there's overdispersion

```r
FieldConfig = c(Omega1 = 1, Epsilon1 = 1, Omega2 = 1,
    Epsilon2 = 1)
RhoConfig = c(Beta1 = 0, Beta2 = 0, Epsilon1 = 0, Epsilon2 = 0)
VesselConfig = c(Vessel = 0, VesselYear = 0)
ObsModel = 2
```

3.3 Stratification for results

We also define any potential stratification of results, and settings specific to any case-study data set

```r
# Default
if (Data_Set %in% c("GSL_american_plaice", "BC_pacific_cod",
    "EBS_pollock", "SAWC_jacopever", "Chatham_rise_hake",
    "Aleutian_islands_POP")) {
    strata.limits <- data.frame(STRATA = "All_areas")
}
# Specific (useful as examples)
if (Data_Set %in% c("WCGBTS_canary", "Sim")) {
    # In this case, it will calculate a coastwide
    # index, and also a separate index for each state
    # (although the state lines are approximate)
    strata.limits <- data.frame(STRATA = c("Coastwide",
        "CA", "OR", "WA"), north_border = c(49, 42,
        46, 49), south_border = c(32, 32, 42, 46),
        shallow_border = c(55, 55, 55, 55), deep_border = c(1280,
            1280, 1280, 1280))
    # Override default settings for vessels
    VesselConfig = c(Vessel = 0, VesselYear = 1)
}
if (Data_Set %in% c("GOA_Pcod", "GOA_pollock")) {
    # In this case, will calculating an unrestricted
    # index and a separate index restricted to west of
    # -140W
    strata.limits <- data.frame(STRATA = c("All_areas",
        "west_of_140W"), west_border = c(-Inf, -Inf),
        east_border = c(Inf, -140))
}
if (Data_Set %in% c("GB_spring_haddock", "GB_fall_haddock")) {
    # For NEFSC indices, strata must be specified as a
    # named list of area codes
    strata.limits = list(Georges_Bank = c(1130, 1140,
        1150, 1160, 1170, 1180, 1190, 1200, 1210, 1220,
        1230, 1240, 1250, 1290, 1300))
}
if (Data_Set %in% c("Iceland_cod")) {
    strata.limits = data.frame(STRATA = "All_areas")
    # Turn off all spatial, temporal, and
    # spatio-temporal variation in probability of
    # occurrence, because they occur almost everywhere
    FieldConfig = c(Omega1 = 0, Epsilon1 = 0, Omega2 = 1,
        Epsilon2 = 1)
    RhoConfig = c(Beta1 = 3, Beta2 = 0, Epsilon1 = 0,
```

```
        Epsilon2 = 0)
}
```

3.4 Derived objects

Depending on the case study, we define a `Region` used when extrapolating or plotting density estimates. If its a different data set, it will define `Region="Other"`, and this is a recognized level for all uses of `Region` (which attempts to define reasonable settings based on the location of sampling). For example `Data_Set="Iceland_cod"` has no associated meta-data for the region, so it uses `Region="Other"` by default.

```
Region = switch( Data_Set, "Chatham_rise_hake"="New_Zealand",
                 "WCGBTS_canary"="California_current",
                 "GSL_american_plaice"="Gulf_of_St_Lawrence",
                 "BC_pacific_cod"="British_Columbia",
                 "EBS_pollock"="Eastern_Bering_Sea",
                 "GOA_Pcod"="Gulf_of_Alaska",
                 "GOA_pollock"="Gulf_of_Alaska",
                 "GB_spring_haddock"="Northwest_Atlantic",
                 "GB_fall_haddock"="Northwest_Atlantic",
                 "SAWC_jacopever"="South_Africa",
                 "Aleutian_islands_POP"="Aleutian_Islands",
                 "Other")
```

3.5 Save settings

We then set the location for saving files.

```
DateFile = paste0(getwd(),'/SpatialDeltaGLMM_output/')
  dir.create(DateFile)
```

I also like to save all settings for later reference, although this is not necessary.

```
Record = ThorsonUtilities::bundlelist(c("Data_Set",
    "strata.limits", "Region", "Version", "Method",
    "grid_size_km", "n_x", "FieldConfig", "RhoConfig",
    "VesselConfig", "ObsModel", "Kmeans_Config"))
save(Record, file = file.path(DateFile, "Record.RData"))
capture.output(Record, file = paste0(DateFile, "Record.txt"))
```

4 Prepare the data

4.1 Data-frame for catch-rate data

Depending upon the `Data_Set` chosen, we load archived data sets that are distributed with the package. Each archived data set is then reformatted to create a data-frame `Data_Geostat` with a standardized set of columns. For a new data set, the user is responsible for formatting `Data_Geostat` appropriately to match this format. We show the first six rows of `Data_Geostat` given that Data_Set = `Data_Set`.

Year	Lat	Lon	Vessel	AreaSwept_km2	Catch_KG
1971	48	-64.8	missing	0.0324	129
1971	48	-65.1	missing	0.0393	88.6
1971	47.9	-65.2	missing	0.0509	38.2
1971	47.9	-65.5	missing	0.037	139
1971	48	-64.6	missing	0.0509	0.786
1971	48.2	-64.5	missing	0.044	0.637

4.2 Extrapolation grid

We also generate the extrapolation grid appropriate for a given region. For new regions, we use `Region="Other"`.

```
if (Region %in% c("California_current", "Eastern_Bering_Sea",
    "Gulf_of_Alaska", "Aleutian_Islands", "Northwest_Atlantic",
    "Gulf_of_St_Lawrence", "New_Zealand")) {
    Extrapolation_List = Prepare_Extrapolation_Data_Fn(Region = Region,
        strata.limits = strata.limits)
}
if (Region == "British_Columbia") {
    Extrapolation_List = Prepare_Extrapolation_Data_Fn(Region = Region,
        strata.limits = strata.limits, strata_to_use = c("HS",
            "QCS"))
}
if (Region == "South_Africa") {
    Extrapolation_List = Prepare_Extrapolation_Data_Fn(Region = Region,
        strata.limits = strata.limits, region = "west_coast")
}
if (Region == "Other") {
    Extrapolation_List = Prepare_Extrapolation_Data_Fn(Region = Region,
        strata.limits = strata.limits, observations_LL = Data_Geostat[,
            c("Lat", "Lon")], maximum_distance_from_sample = 15)
}
```

4.3 Derived objects for spatio-temporal estimation

And we finally generate the information used for conducting spatio-temporal parameter estimation, bundled in list `Spatial_List`

```
Spatial_List = Spatial_Information_Fn(grid_size_km = grid_size_km,
    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 = DateFile,
    Save_Results = FALSE)
# Add knots to Data_Geostat
Data_Geostat = cbind(Data_Geostat, Spatial_List$loc_UTM,
    knot_i = Spatial_List$knot_i)
```

5 Build and run model

5.1 Build model

To estimate parameters, we first build a list of data-inputs used for parameter estimation. `Data_Fn` has some simple checks for buggy inputs, but also please read the help file `?Data_Fn`.

```
TmbData = Data_Fn(Version = Version, FieldConfig = FieldConfig,
    RhoConfig = RhoConfig, ObsModel = ObsModel, 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 = c(SD_site_density = 0, SD_site_logdensity = 0,
        Calculate_Range = 1, Calculate_evenness = 0,
        Calculate_effective_area = 1))
```

We then build the TMB object.

```
TmbList = Build_TMB_Fn(TmbData = TmbData, RunDir = DateFile,
    Version = Version, RhoConfig = RhoConfig, VesselConfig = VesselConfig,
    loc_x = Spatial_List$loc_x)
Obj = TmbList[["Obj"]]
```

5.2 Estimate fixed effects and predict random effects

Next, we use a gradient-based nonlinear minimizer to identify maximum likelihood estimates for fixed-effects

```
Opt = TMBhelper::Optimize(obj = Obj, lower = TmbList[["Lower"]],
    upper = TmbList[["Upper"]], getsd = TRUE, savedir = DateFile,
    bias.correct = FALSE)
```

Finally, we bundle and save output

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

6 Diagnostic plots

We first apply a set of standard model diagnostics to confirm that the model is reasonable and deserves further attention. If any of these do not look reasonable, the model output should not be interpreted or used.

6.1 Plot data

It is always good practice to conduct exploratory analysis of data. Here, I visualize the spatial distribution of data. Spatio-temporal models involve the assumption that the probability of sampling a given location is statistically independent of the probability distribution for the response at that location. So if sampling “follows” changes in density, then the model is probably not appropriate!

```r
Plot_data_and_knots(Extrapolation_List = Extrapolation_List,
    Spatial_List = Spatial_List, Data_Geostat = Data_Geostat,
    PlotDir = DateFile)
```

```
## Warning: package 'maps' was built under R version
## 3.2.5
```

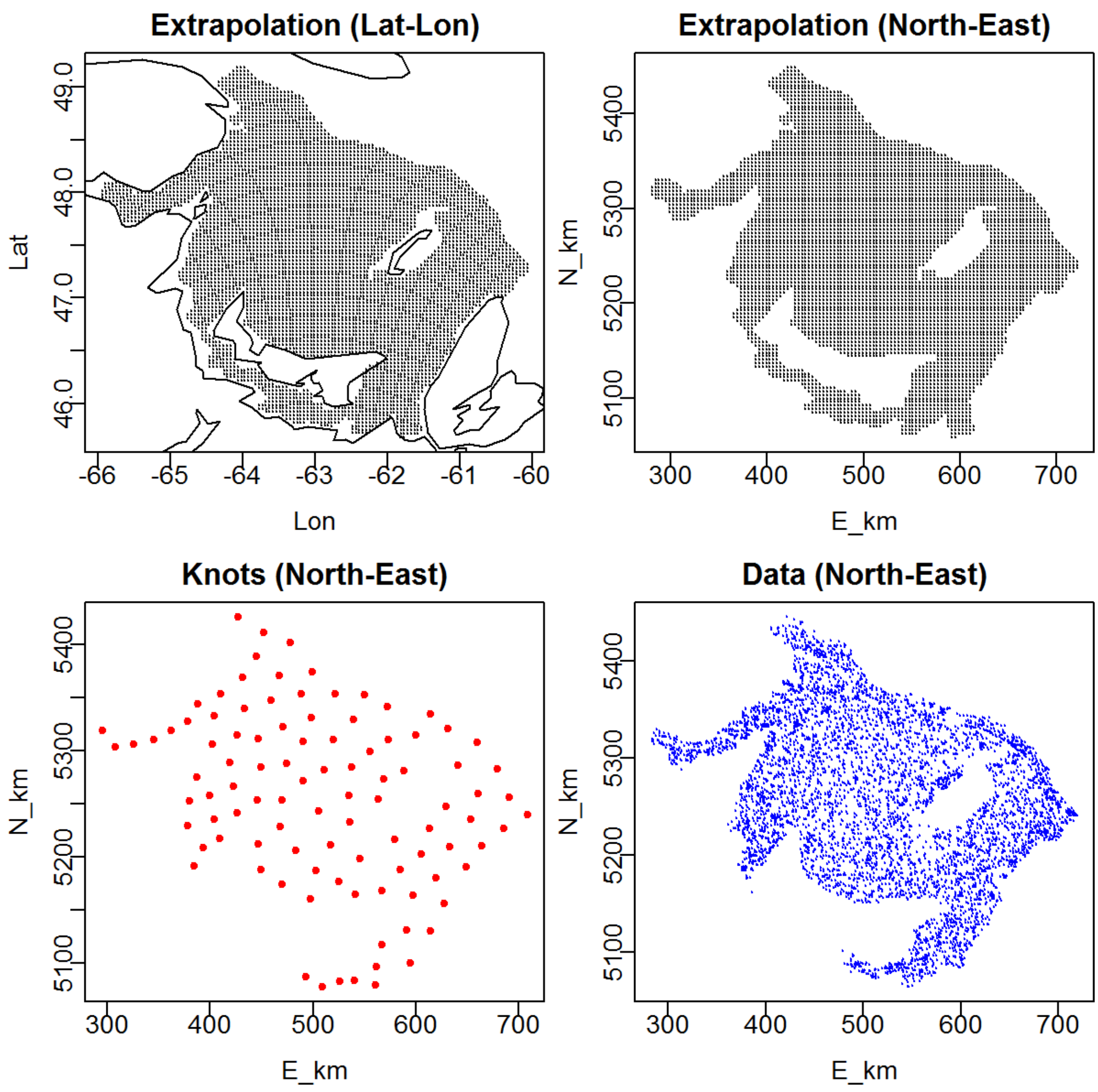

Figure 1: Spatial extent and location of knots

6.2 Convergence

Here I print the diagnostics generated during parameter estimation, and I confirm that (1) no parameter is hitting an upper or lower bound and (2) the final gradient for each fixed-effect is close to zero.

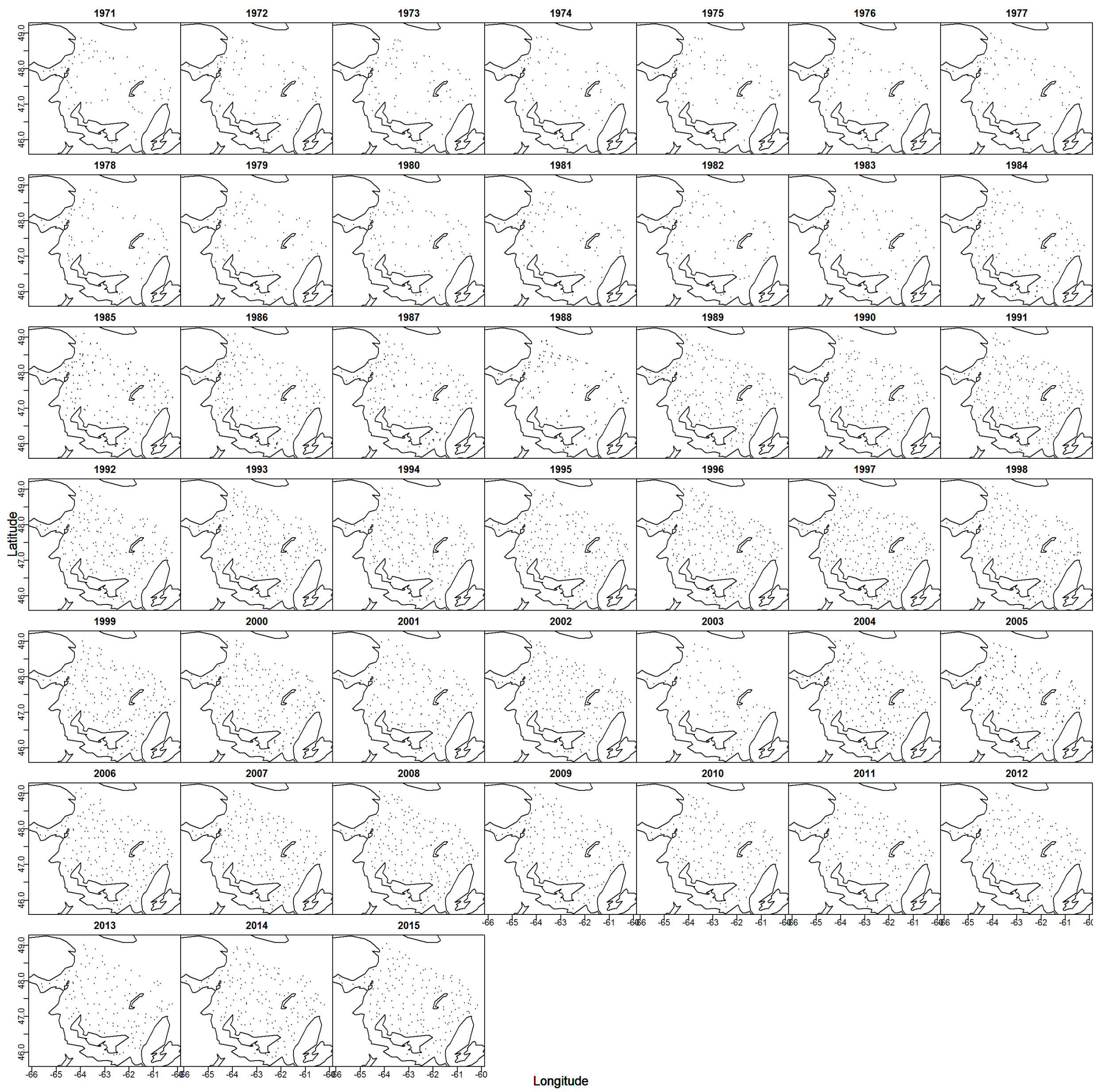

Figure 2: Spatial distribution of catch-rate data

```r
pander::pandoc.table( Opt$diagnostics[,c('Param','Lower','MLE','Upper','final_gradient')] )
```

Param	Lower	MLE	Upper	final_gradient
ln_H_input	-50	-0.1446	50	5.71e-06
ln_H_input	-50	-0.172	50	0.0002003
beta1_t	-50	2.715	50	-0.0001058
beta1_t	-50	1.624	50	1.571e-05
beta1_t	-50	1.956	50	1.548e-05
beta1_t	-50	3.501	50	5.817e-05
beta1_t	-50	2.901	50	-0.0001165
beta1_t	-50	2.283	50	3.038e-05
beta1_t	-50	3.217	50	-0.0001053
beta1_t	-50	2.996	50	9.826e-05
beta1_t	-50	2.904	50	-7.736e-05
beta1_t	-50	2.54	50	3.523e-05
beta1_t	-50	3.466	50	7.057e-05
beta1_t	-50	2.116	50	-2.019e-05
beta1_t	-50	3.081	50	0.0002701
beta1_t	-50	3.013	50	9.917e-05
beta1_t	-50	3.417	50	-1.721e-06
beta1_t	-50	4.295	50	-0.0001053
beta1_t	-50	3.316	50	9.827e-05
beta1_t	-50	3.038	50	-5.417e-05
beta1_t	-50	2.689	50	-5.502e-06
beta1_t	-50	2.804	50	1.494e-05
beta1_t	-50	2.789	50	-1.631e-05
beta1_t	-50	2.865	50	2.471e-05
beta1_t	-50	2.506	50	-9.835e-06
beta1_t	-50	3.391	50	2.341e-05
beta1_t	-50	2.899	50	-1.592e-05
beta1_t	-50	3.134	50	-1.67e-05
beta1_t	-50	2.422	50	-3.761e-06
beta1_t	-50	2.981	50	2.779e-06
beta1_t	-50	2.521	50	6.997e-06
beta1_t	-50	3.168	50	3.075e-06
beta1_t	-50	2.566	50	-2.369e-05
beta1_t	-50	2.465	50	2.634e-06
beta1_t	-50	3.812	50	5.203e-05
beta1_t	-50	2.794	50	2.832e-06
beta1_t	-50	3.103	50	-8.414e-06
beta1_t	-50	3.265	50	-4.36e-05
beta1_t	-50	3.438	50	-5.206e-05
beta1_t	-50	3.823	50	-9.353e-05
beta1_t	-50	3.053	50	1.551e-05
beta1_t	-50	3.444	50	1.429e-05
beta1_t	-50	2.753	50	-8.151e-05
beta1_t	-50	2.564	50	2.661e-05
beta1_t	-50	3.527	50	-0.0001185
beta1_t	-50	3.437	50	3.98e-05
beta1_t	-50	3.246	50	8.058e-05
logetaE1	-50	-0.2503	3.34	4.076e-05
logetaO1	-50	-2.093	3.34	8.2e-06
logkappa1	-5.005	-3.458	-1.595	-0.000169

Param	Lower	MLE	Upper	final_gradient
beta2_t	-50	5.718	50	4.451e-05
beta2_t	-50	6.045	50	2.506e-05
beta2_t	-50	6.101	50	-3.214e-05
beta2_t	-50	6.103	50	-1.887e-05
beta2_t	-50	6.322	50	-1.738e-05
beta2_t	-50	6.915	50	-3.289e-05
beta2_t	-50	6.773	50	-2.268e-05
beta2_t	-50	6.058	50	3.025e-05
beta2_t	-50	6.69	50	-2.96e-06
beta2_t	-50	6.386	50	-2.546e-05
beta2_t	-50	5.795	50	3.317e-06
beta2_t	-50	6.151	50	-6.023e-05
beta2_t	-50	6.014	50	-1.589e-05
beta2_t	-50	5.727	50	2.093e-05
beta2_t	-50	5.554	50	-5.419e-05
beta2_t	-50	5.654	50	2.092e-05
beta2_t	-50	5.665	50	1.778e-06
beta2_t	-50	5.825	50	2.463e-05
beta2_t	-50	5.475	50	1.027e-05
beta2_t	-50	5.966	50	1.316e-05
beta2_t	-50	5.578	50	2.02e-05
beta2_t	-50	5.676	50	-1.209e-05
beta2_t	-50	5.52	50	6.527e-06
beta2_t	-50	5.571	50	-4.917e-06
beta2_t	-50	5.25	50	5.972e-06
beta2_t	-50	5.162	50	-2.082e-05
beta2_t	-50	4.827	50	4.264e-05
beta2_t	-50	5.037	50	1.205e-05
beta2_t	-50	5.065	50	4.958e-06
beta2_t	-50	4.663	50	4.009e-06
beta2_t	-50	4.508	50	-5.327e-05
beta2_t	-50	4.774	50	-3.009e-06
beta2_t	-50	5.004	50	3.012e-05
beta2_t	-50	4.716	50	5.841e-05
beta2_t	-50	4.564	50	1.166e-05
beta2_t	-50	4.666	50	1.986e-06
beta2_t	-50	4.8	50	2.678e-05
beta2_t	-50	4.761	50	2.049e-05
beta2_t	-50	4.458	50	6.923e-06
beta2_t	-50	4.657	50	2.078e-05
beta2_t	-50	4.657	50	-6.121e-05
beta2_t	-50	4.45	50	-4.082e-05
beta2_t	-50	4.812	50	6.406e-06
beta2_t	-50	4.435	50	5.179e-05
beta2_t	-50	4.56	50	-1.045e-05
logetaE2	-50	-0.8921	3.34	-4.222e-06
logetaO2	-50	-1.578	3.34	-0.0002329
logkappa2	-5.005	-3.7	-1.595	0.0005085
logSigmaM	-50	-0.03413	10	0.002595

6.3 Diagnostics for encounter-probability component

Next, we check whether observed encounter frequencies for either low or high probability samples are within the 95% predictive interval for predicted encounter probability

```
Enc_prob = Check_encounter_prob(Report = Report, Data_Geostat = Data_Geostat,
    DirName = DateFile)
```

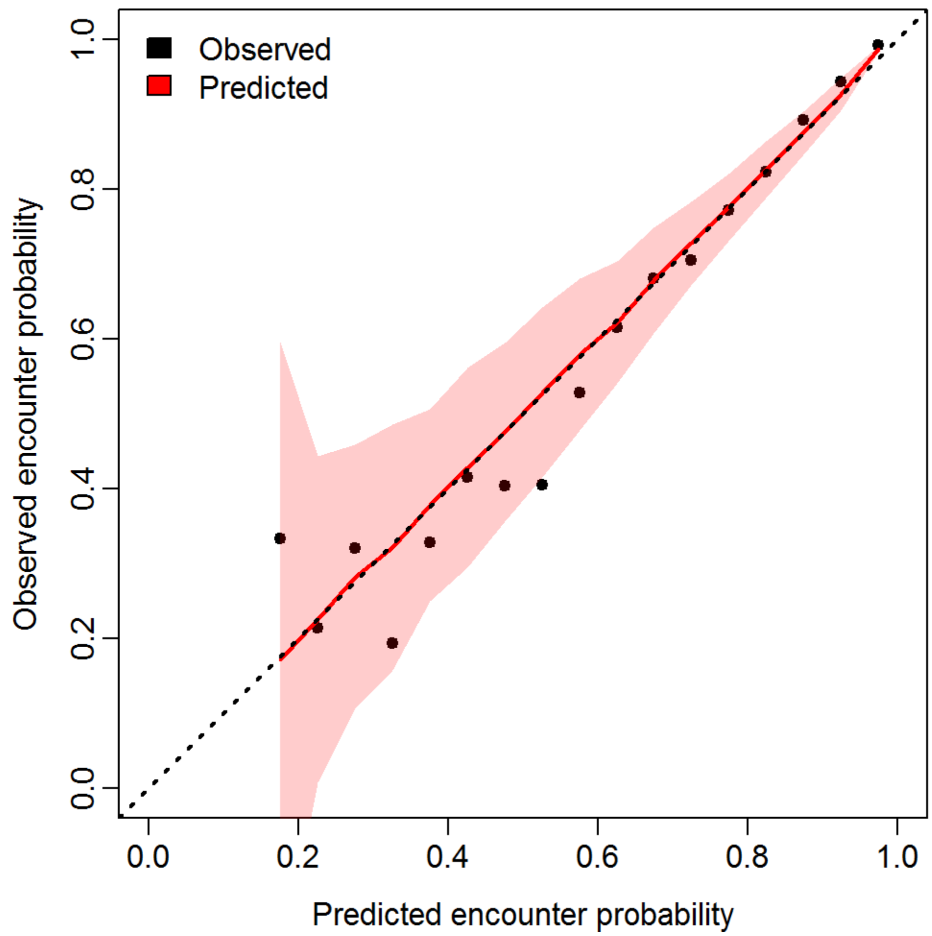

Figure 3: Expectated probability and observed frequency of encounter for "encounter probability" component

6.4 Diagnostics for positive-catch-rate component

We can visualize fit to residuals of catch-rates given encounters using a Q-Q plot. A good Q-Q plot will have residuals along the one-to-one line.

```
Q = QQ_Fn(TmbData = TmbData, Report = Report, FileName_PP = paste0(DateFile,
    "Posterior_Predictive.jpg"), FileName_Phist = paste0(DateFile,
    "Posterior_Predictive-Histogram.jpg"), FileName_QQ = paste0(DateFile,
    "Q-Q_plot.jpg"), FileName_Qhist = paste0(DateFile,
    "Q-Q_hist.jpg"))
```

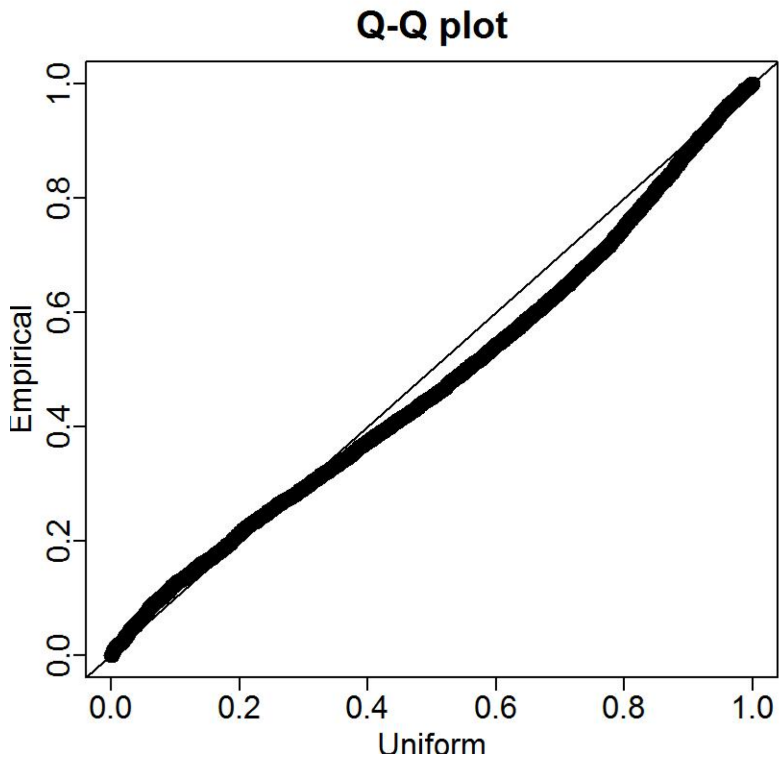

Figure 4: Quantile-quantile plot indicating residuals for "positive catch rate" component

7 Model output

Last but not least, we generate useful plots by first determining which years to plot (`Years2Include`), and labels for each plotted year (`Year_Set`)

```
Year_Set = seq(min(Data_Geostat[,'Year']),max(Data_Geostat[,'Year']))
Years2Include = which( Year_Set %in% sort(unique(Data_Geostat[,'Year'])))
```

We then run a set of pre-defined plots for visualizing results

7.1 Direction of "geometric anisotropy"

We can visualize which direction has faster or slower decorrelation (termed "geometric anisotropy")

```
PlotAniso_Fn( FileName=paste0(DateFile,"Aniso.png"), Report=Report, TmbData=TmbData )
```

7.2 Density surface for each year

We can visualize many types of output from the model. Here I only show predicted density, but other options are obtained via other integers passed to `plot_set` as described in `?PlotResultsOnMap_Fn`

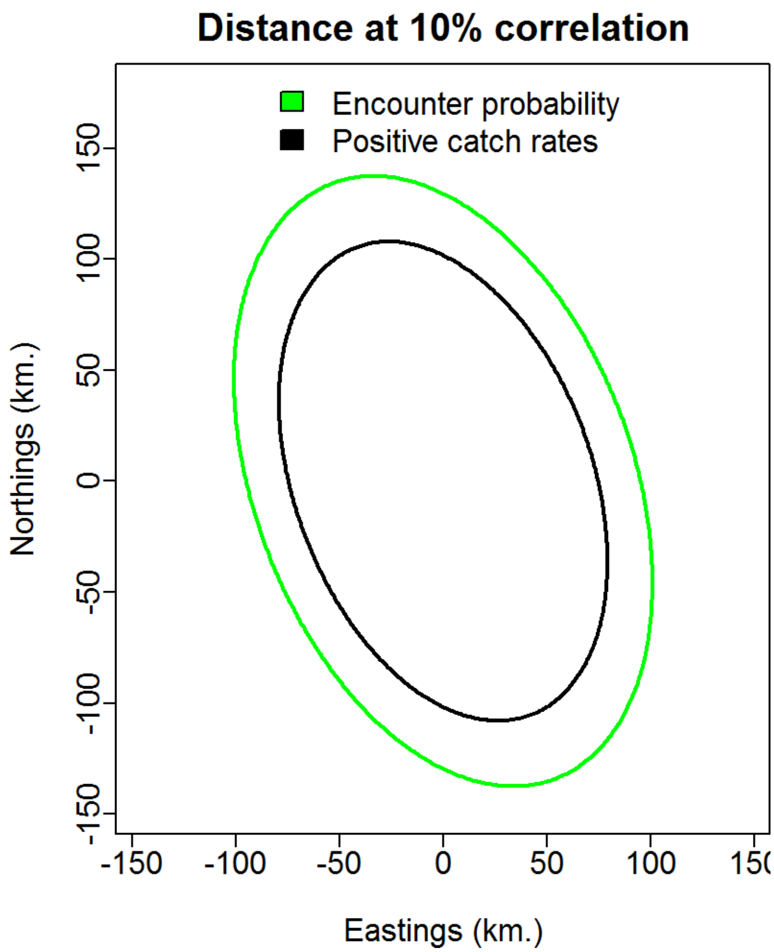

Figure 5: Decorrelation distance for different directions

```r
# Get region-specific settings for plots
MapDetails_List = MapDetails_Fn(Region = Region, NN_Extrap = Spatial_List$PolygonList$NN_Extrap,
    Extrapolation_List = Extrapolation_List)
# Plot maps representing density or other variables
PlotResultsOnMap_Fn(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 = DateFile, 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)
```

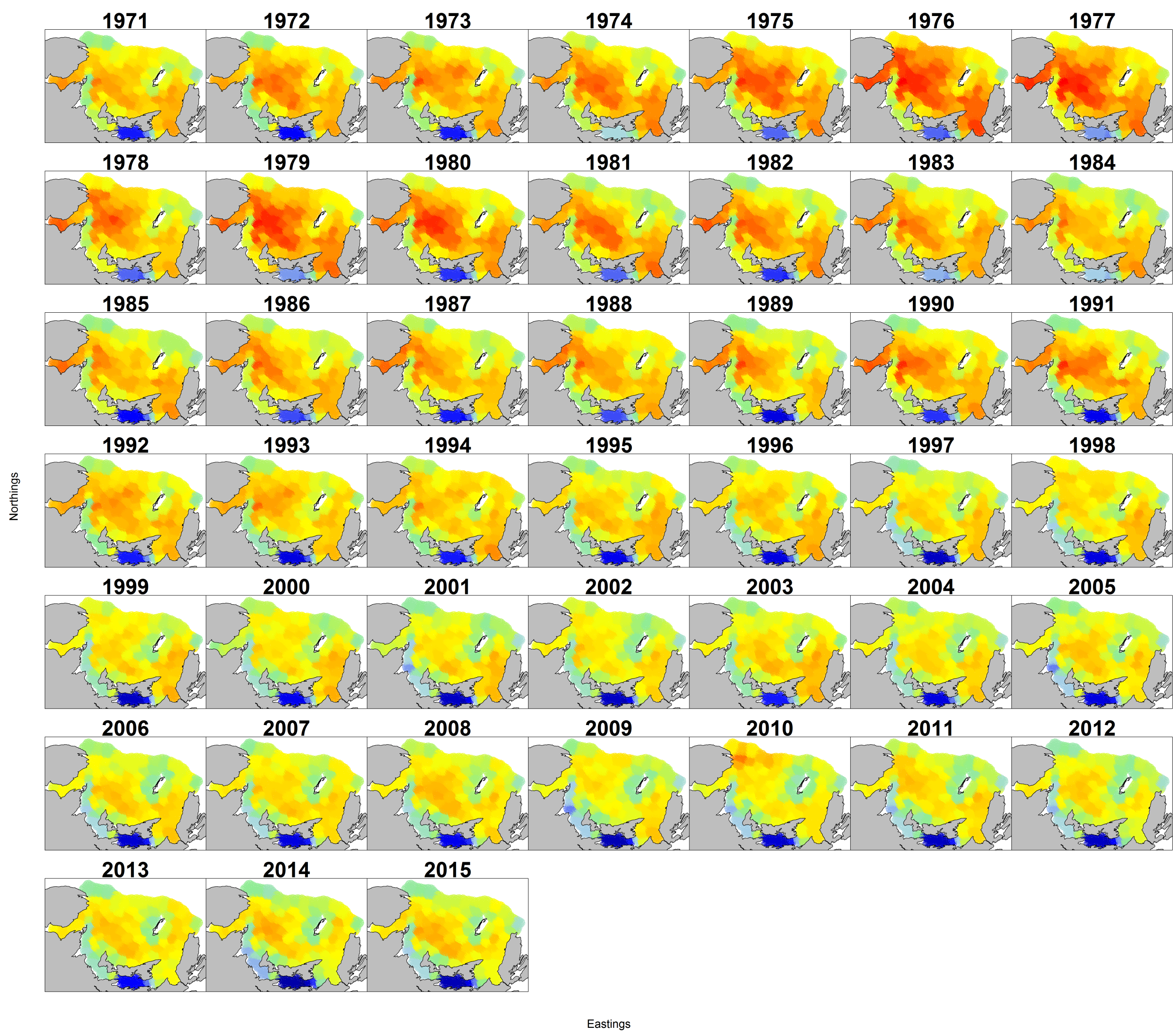

Figure 6: Density maps for each year

7.3 Index of abundance

The index of abundance is generally most useful for stock assessment models.

```
Index = PlotIndex_Fn(DirName = DateFile, TmbData = TmbData,
    Sdreport = Opt[["SD"]], Year_Set = Year_Set, Years2Include = Years2Include,
    strata_names = strata.limits[, 1], use_biascorr = TRUE)
pander::pandoc.table(Index$Table[, c("Year", "Fleet",
    "Estimate_metric_tons", "SD_log", "SD_mt")])
```

Year	Fleet	Estimate_metric_tons	SD_log	SD_mt
1971	All_areas	47684	0.1871	8922
1972	All_areas	70144	0.1995	13993
1973	All_areas	70018	0.1913	13396
1974	All_areas	88608	0.1852	16409
1975	All_areas	112463	0.1916	21548
1976	All_areas	180954	0.1775	32122
1977	All_areas	163790	0.1799	29458
1978	All_areas	75803	0.1843	13973
1979	All_areas	145901	0.1762	25713
1980	All_areas	116147	0.1803	20944
1981	All_areas	80664	0.182	14678
1982	All_areas	80053	0.193	15452
1983	All_areas	65770	0.1672	10997
1984	All_areas	48756	0.1382	6738
1985	All_areas	51918	0.1003	5207
1986	All_areas	60117	0.1099	6609
1987	All_areas	52884	0.1077	5694
1988	All_areas	58157	0.1306	7595
1989	All_areas	51245	0.1173	6011
1990	All_areas	74321	0.1124	8351
1991	All_areas	70618	0.09892	6985
1992	All_areas	51355	0.09628	4945
1993	All_areas	43180	0.1007	4346
1994	All_areas	40792	0.09314	3799
1995	All_areas	34726	0.09441	3279
1996	All_areas	34678	0.09126	3165
1997	All_areas	23678	0.08731	2067
1998	All_areas	26879	0.0891	2395
1999	All_areas	24455	0.09172	2243
2000	All_areas	22089	0.09605	2122
2001	All_areas	20399	0.1107	2259
2002	All_areas	18367	0.09544	1753
2003	All_areas	28638	0.1356	3882
2004	All_areas	19135	0.09377	1794
2005	All_areas	21246	0.08883	1887
2006	All_areas	21739	0.09886	2149
2007	All_areas	22179	0.09481	2103
2008	All_areas	25721	0.09409	2420
2009	All_areas	15483	0.1011	1566
2010	All_areas	21244	0.1065	2263
2011	All_areas	20422	0.1137	2321
2012	All_areas	16126	0.1081	1743

Year	Fleet	Estimate_metric_tons	SD_log	SD_mt
2013	All_areas	21813	0.1133	2472
2014	All_areas	21919	0.1035	2269
2015	All_areas	24343	0.1035	2520

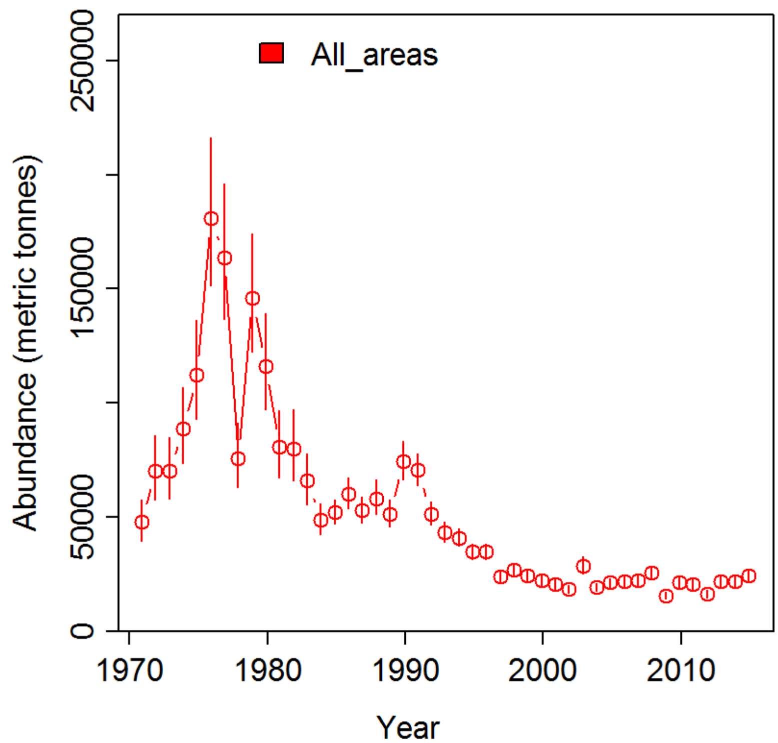

Figure 7: Index of abundance plus/minus 1 standard error

7.4 Center of gravity and range expansion/contraction

We can detect shifts in distribution or range expansion/contraction.

```
Plot_range_shifts(Report = Report, TmbData = TmbData,
    Sdreport = Opt[["SD"]], Znames = colnames(TmbData$Z_xm),
    PlotDir = DateFile)
```

7.5 Vessel effects if included

Most example data-sets don't have vessel effects, so this plot is generally skipped

```
Return = Vessel_Fn(TmbData = TmbData, Sdreport = Opt[["SD"]],
    FileName_VYplot = paste0(DateFile, "VY-effect.jpg"))
```

```
## Not plotting vessel effects because none are present
```

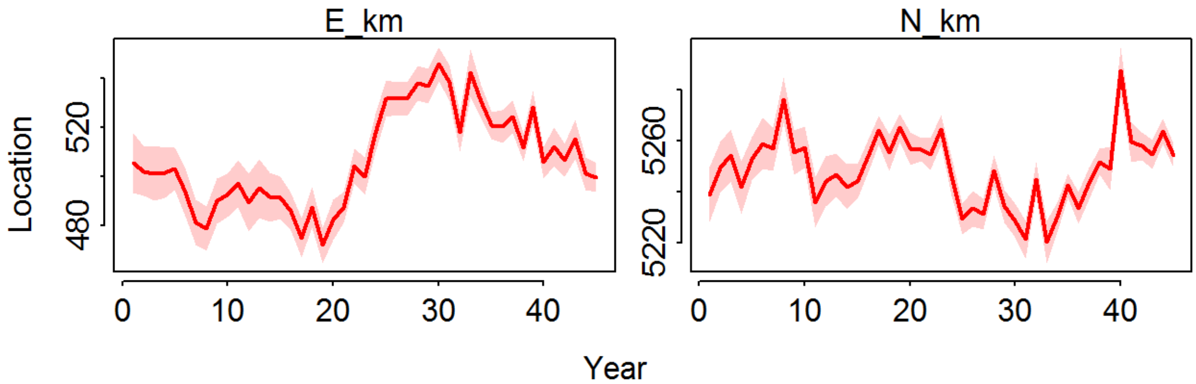

Figure 8: Center of gravity (COG) indicating shifts in distribution plus/minus 1 standard error

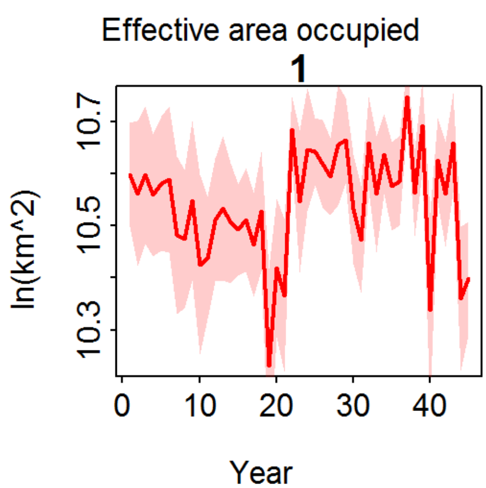

Figure 9: Effective area occupied indicating range expansion/contraction plus/minus 1 standard error