Electronic supplementary material to '*Spatial conservation prioritization of marine bird distribution models to improve guidance for siting of offshore developments*'

Kristopher J. Winiarski[a,], David L. Miller[a], Peter W. C. Paton[a], Scott R. McWilliams[a]*

[a]Department of Natural Resources Science, 1 Greenhouse Road, University of Rhode Island, Kingston, RI 02881, United States

[*]Corresponding author e-mail: Withakri@gmail.com

Preamble

This electronic supplementary material documents the analyses for the accompanying manuscript. Also included are the R package `osampuri` containing the data necessary to run the analyses, along with a set of "`.Rmd`" files which can be used to generate the reports below.

Common loons

This document is a record of modelling for the common loon data from the URI aerial line transect survey of the OSAMP area off the coast of Rhode Island. It is provided as a `knitr` (Xie 2013) file, that includes all the necessary code to re-create the analysis.

Preamble

Load the data and `dsm` package.

```
suppressPackageStartupMessages(library(dsm))
suppressPackageStartupMessages(library(osampuri))
data("uri-lt-data")
```

Select only loons

```
obs.loons <- obs[obs$Group == "Loon", ]
rm(obs)
```

Then only those loons with recorded bins on the water in the Winter

```
obs.loons <- obs.loons[obs.loons$Bin != "Not recorded", ]
obs.loons <- obs.loons[obs.loons$Location == "On Water", ]
obs.loons <- obs.loons[obs.loons$Season == "Winter", ]
effort <- effort[effort$Season == "Winter", ]
```

Then allocate the effort

```
new.segobs <- allocate.effort(obs.loons, seg)
obs.loons <- new.segobs$obs
seg <- new.segobs$seg
```

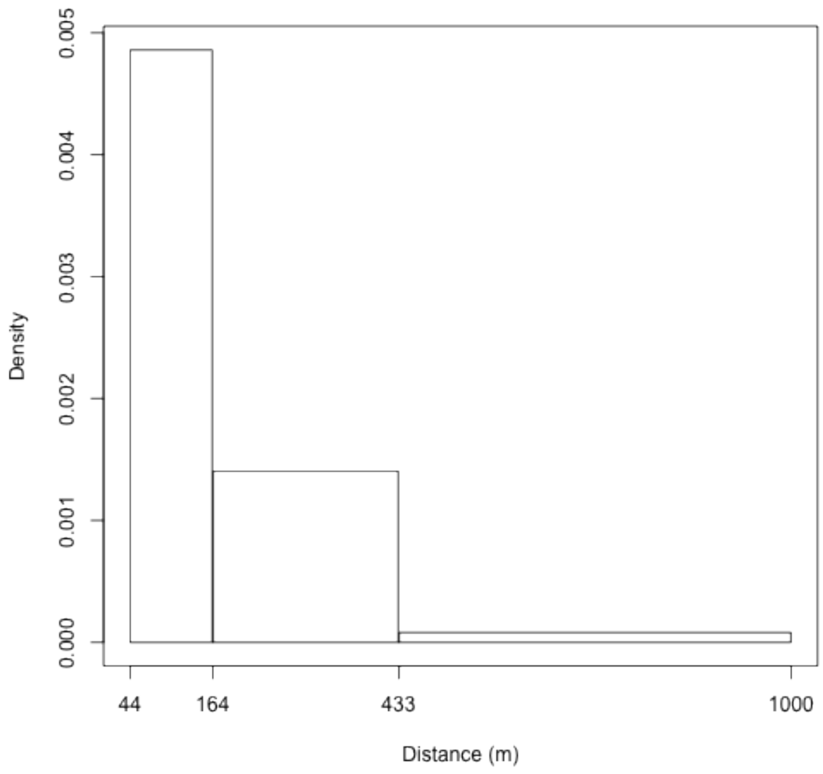
Save some plotting options

```
pred$width <- rep(2, nrow(pred))
pred$height <- rep(2, nrow(pred))

p.opts.geo <- theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
    panel.background = element_blank(), strip.background = element_blank(),
    legend.key = element_blank(), aspect.ratio = 1)
xlims <- c(-45, 40)
ylims <- c(-27, 34)
```

We begin by plotting the raw data: the observed distances, raw observations both unaggregated and split according to survey season.

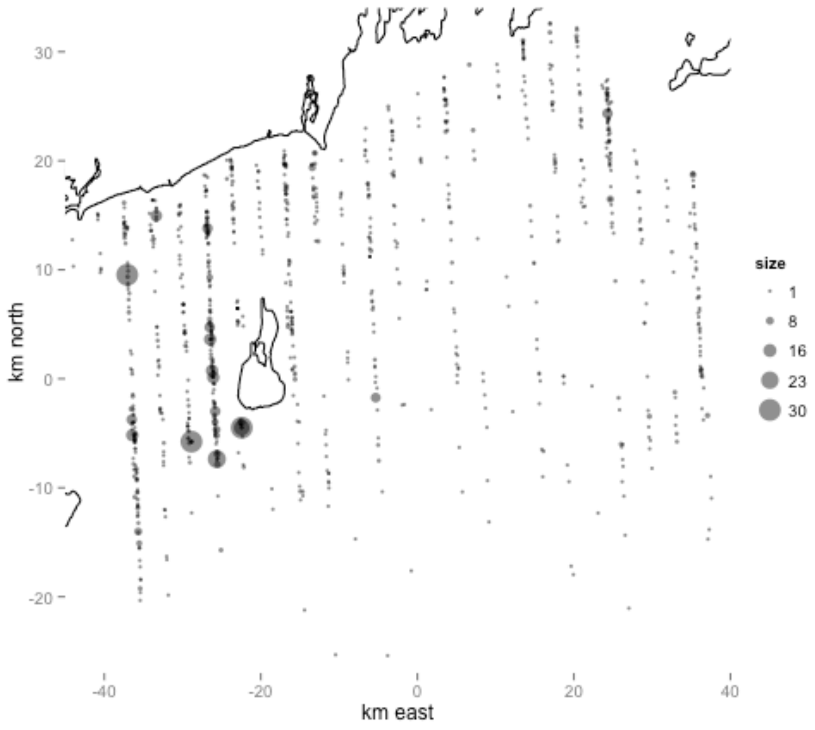
First plotting the histogram of observed distances:

```
hist(obs.loons$distance, breaks = sort(unique(c(obs.loons$distbegin, obs.loons$distend))),
    main = "", xlab = "Distance (m)", axes = FALSE)
axis(2)
axis(1, at = c(44, 164, 433, 1000))
box()
```



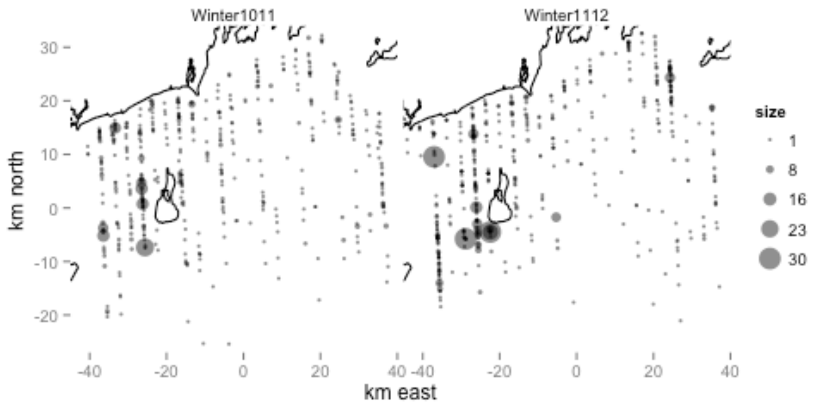
Histogram of observed distances to flocks of common loons

```
p <- ggplot(obs.loons)
p <- p + geom_point(aes(x = x, y = y, size = size), alpha = 0.5)
p <- p + geom_path(aes(x = x, y = y, group = group), data = coast)
p <- p + p.opts.geo
p <- p + coord_equal(xlim = xlims, ylim = ylims)
leg.breaks <- unique(quantile(obs.loons$size))
leg.breaks <- round(seq(leg.breaks[1], leg.breaks[2], len = 5), 0)
leg.breaks <- round(leg.breaks, 0)
p <- p + scale_size(breaks = leg.breaks)
p <- p + labs(x = "km east", y = "km north")
print(p)
```



Raw observations of common loons. The size of the circle relates to the size of the observed flock.

```
p <- p + facet_wrap(~SeasonYear)
print(p)
```



Raw observations of common loons aggregated to the survey season. The size of the circle relates to the size of the observed flock.

These raw plots clearly show higher observed abundances in the area between Block Island and Long Island Sound.

Detection function analysis

Stage one of the density surface modelling approach is to adjust the counts to account for detectability. We begin by fitting a detection function to the distance data using the `R` package `Distance`.

Fit a detection function with half-normal and hazard rate and uniform key functions with no covariates

```
hn.df <- ds(obs.loons, truncation = list(right = 1000, left = 44), adjustment = NULL)
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting half-normal key function AIC= 1603.281 No survey area information
## supplied, only estimating detection function.
```

```
hr.df <- ds(obs.loons, truncation = list(right = 1000, left = 44), key = "hr",
    adjustment = NULL)
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function AIC= 1580.559 No survey area information
## supplied, only estimating detection function.
```

We can also see if covariates have an effect, looking at flock size:

```
hn.df.size <- ds(obs.loons, formula = ~size, adjustment = NULL, truncation = list(right = 1000,
    left = 44))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting half-normal key function AIC= 1602.062 No survey area information
## supplied, only estimating detection function.
```

```
hr.df.size <- ds(obs.loons, formula = ~size, adjustment = NULL, truncation = list(right = 1000,
    left = 44), key = "hr")
```

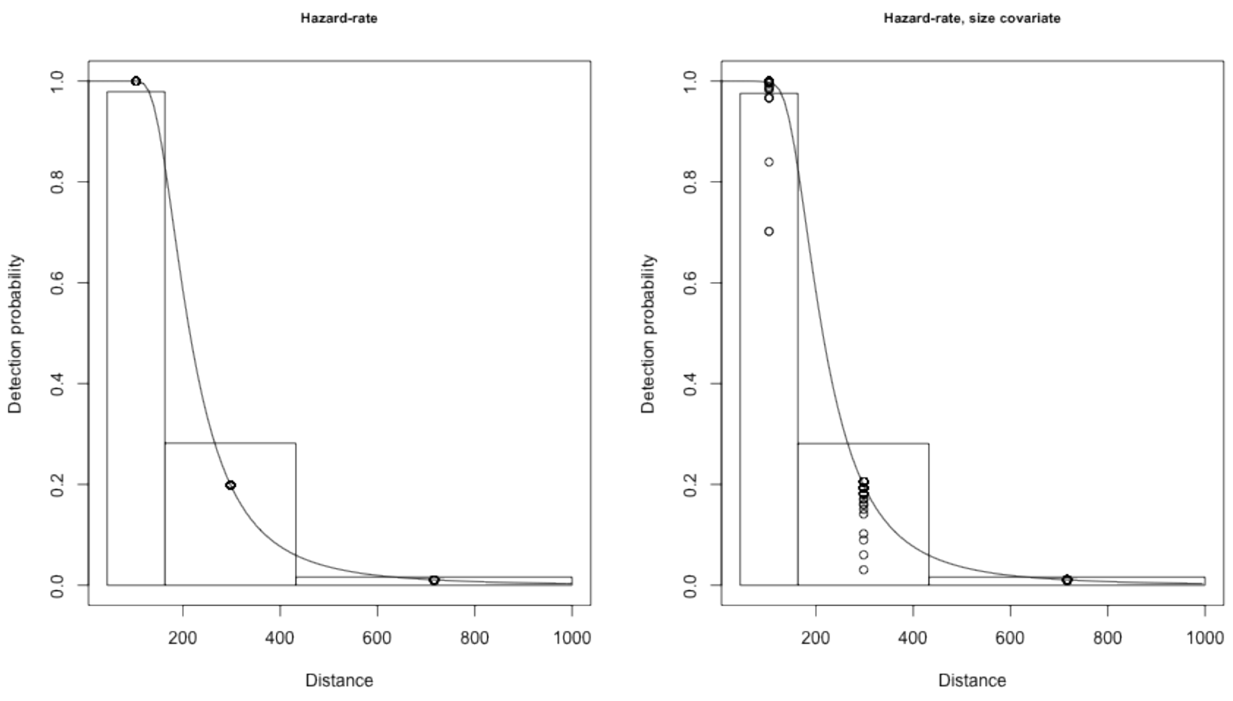
```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function AIC= 1580.124 No survey area information
## supplied, only estimating detection function.
```

Below is a table of the results ordered by AIC.

	Detection function	Adjustments	Covariates	AIC	Δ AIC	# pars	p	CV(p)	C-vM p	KS p
1	hr		size	1580.124	0	3	0.201	0.037	0.03	0
2	hr			1580.559	0.435	2	0.201	0.036	0.034	0
3	hn		size	1602.062	21.938	2	0.193	0.027	0.025	0
4	hn			1603.281	23.157	1	0.194	0.027	0.03	0

```
par(mfrow = c(1, 2))
plot(hr.df, pl.den = 0, main = "Hazard-rate")
plot(hr.df.size, pl.den = 0, main = "Hazard-rate, size covariate")
```



Fitted detection functions.

Given the very small difference in AIC, we opt for the hazard-rate model without covariates.

Spatial modelling

First setting the basis sizes for unidimensional and bivariate smooth terms:

```
k1 <- 10
k2 <- 18
```

We then proceed with model fitting.

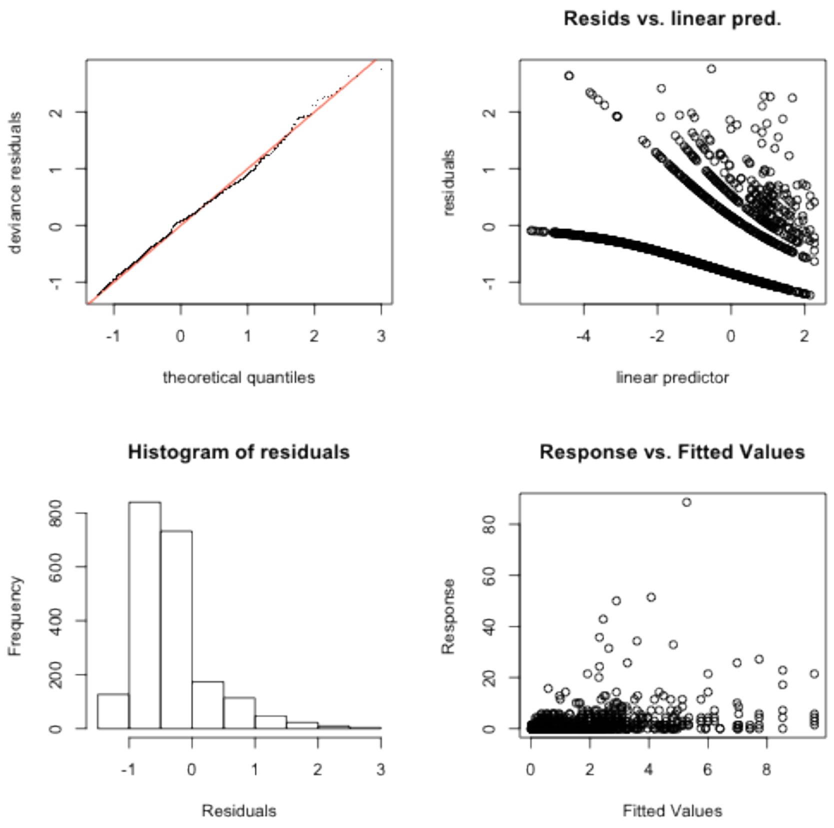
```
loon.model.hr <- dsm(N~s(gchl_long,k=k1)+
#                     s(gchl_winter,k=k1)+
#                     s(fcpi,k=k1)+
#                     s(roughness,k=k1)+
#                     s(phimedian,k=k1)+
#                     s(distancelandkm,k=k1)+
                     s(depthm,k=k1)+
#                     s(x,k=k1)+
                     s(y,k=k1),#+
#                     s(x,y,k=k2),
                   hr.df, seg, obs.loons,
                   family=negbin(theta=c(0.1,0.2)), availability=0.7,
                   select=TRUE, method="REML")
```

```
summary(loon.model.hr)
```

```
##
## Family: Negative Binomial(0.199)
## Link function: log
##
## Formula:
## N ~ s(gchl_long, k = k1) + s(depthm, k = k1) + s(y, k = k1) +
##     offset(off.set)
## <environment: 0x113b75098>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -14.6640     0.0837    -175   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##               edf Ref.df Chi.sq p-value
## s(gchl_long) 4.71      9   97.2 < 2e-16 ***
## s(depthm)    2.37      9   27.7 1.7e-08 ***
## s(y)         3.48      9   40.1 2.2e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.129   Deviance explained = 37.7%
## REML score = 2066.9  Scale est. = 1         n = 2067
```

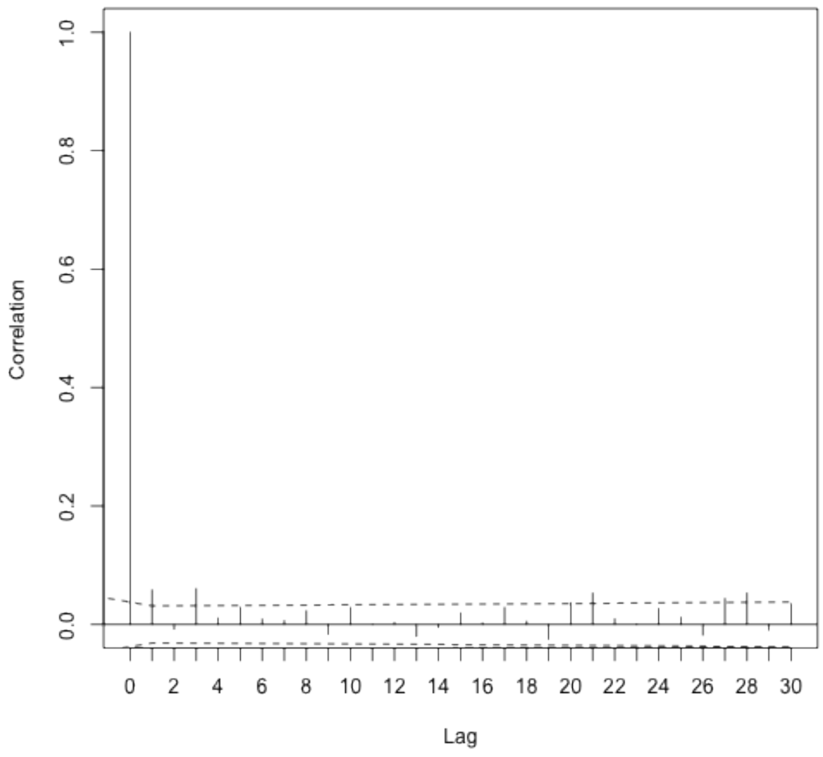
The Q-Q plot for this model looks rather good!

```
gam.check(loon.model.hr)
```



```
##
## Method: REML   Optimizer: outer newton
## full convergence after 1 iteration.
## Gradient range [-8.916e-05,7.864e-06]
## (score 2067 & scale 1).
## Hessian positive definite, eigenvalue range [2.877e-05,1.321].
##
## 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(gchl_long) 9.000 4.711   0.712    0.00
## s(depthm)    9.000 2.374   0.719    0.01
## s(y)         9.000 3.485   0.718    0.01
```

```
dsm.cor(loon.model.hr, max.lag = 30)
```



Predicting the abundance over the OSAMP area and the corresponding confidence interval using the method of Williams et al (2011).

```
summary(dsm.var.prop(loon.model.hr, pred, pred$cellaream))
```

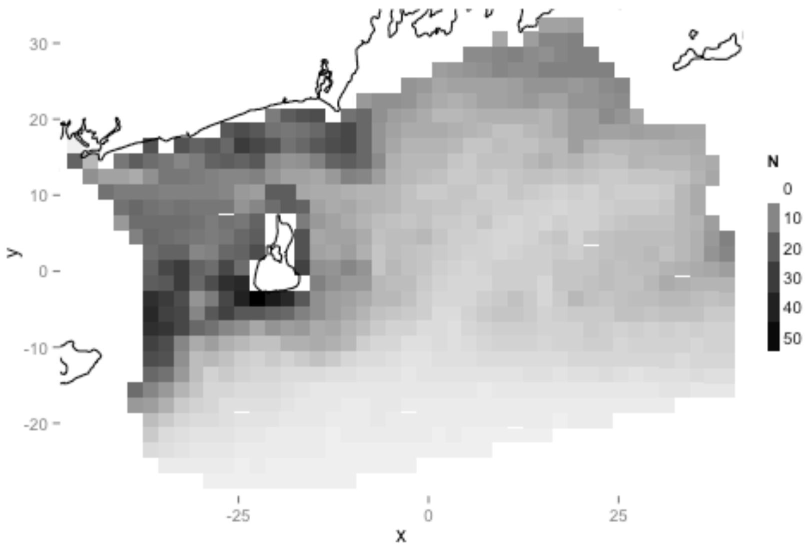
```
## Summary of uncertainty in a density surface model calculated
##  by variance propagation.
##
## Quantiles of differences between fitted model and variance model
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -5.77e-04 -5.10e-06 -3.00e-07 -1.60e-06  3.60e-06  1.41e-03
##
## Approximate asymptotic confidence interval:
##   5% Mean  95%
## 3993 5047 6379
## (Using delta method)
##
## Point estimate                 : 5047
## Standard error                 : 605.4
## Coefficient of variation       : 0.12
```

Predictions and uncertainty

Predictions over the OSAMP area are plotted below:

```
plot.preds(loon.model.hr)
```

```
## Abundance = 5047
```



Plot of model predictions for common loons

Predicted overall abundance and associated confidence interval:

```
loon.var.grid <- dsm.var.gam(loon.model.hr, pred.grid, pred$cellarea)
```

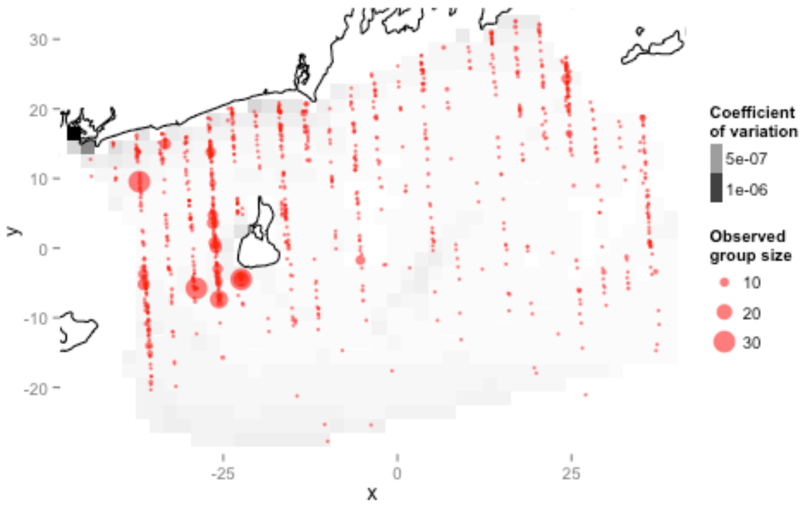
```
## Error: object 'pred.grid' not found
```

```
summary(loon.var.grid)
```

```
## Error: object 'loon.var.grid' not found
```

Finally, a plot of the uncertainty:

```
plot.uncertainty(loon.model.hr, obs = obs.loons)
```



Plot of uncertainty for the common loon DSM

Save everything

```
pred.grid <- split(pred, 1:nrow(pred))
loon.var.grid <- dsm.var.gam(loon.model.hr, pred.grid, pred$cellarea)

loon.preds <- predict(loon.model.hr, pred, pred$cellaream)



loon.pred.data <- data.frame(cellid = 1:920, pred = loon.preds, var = diag(loon.var.grid$pred.var),
    cv = sqrt(diag(loon.var.grid$pred.var))/loon.preds, season = rep("Winter",
        920))
save(loon.pred.data, file = "loon-preds.RData")
```

References

Xie Y (2013) knitr: A general-purpose package for dynamic report generation in R. R package version 1.0. http://CRAN.R-project.org/package=knitr

Storm-petrel analysis

This document is a record of modelling for the storm-petrel data from the URI aerial line transect survey of the OSAMP area off the coast of Rhode Island. It is provided as a `knitr` (Xie 2013) file, that includes all the necessary code to re-create the analysis.

```
suppressPackageStartupMessages(library(osampuri))
data("uri-lt-data")
```

Select the storm-petrels only:

```
obs.petrel <- obs[obs$Group == "Petrel", ]
```

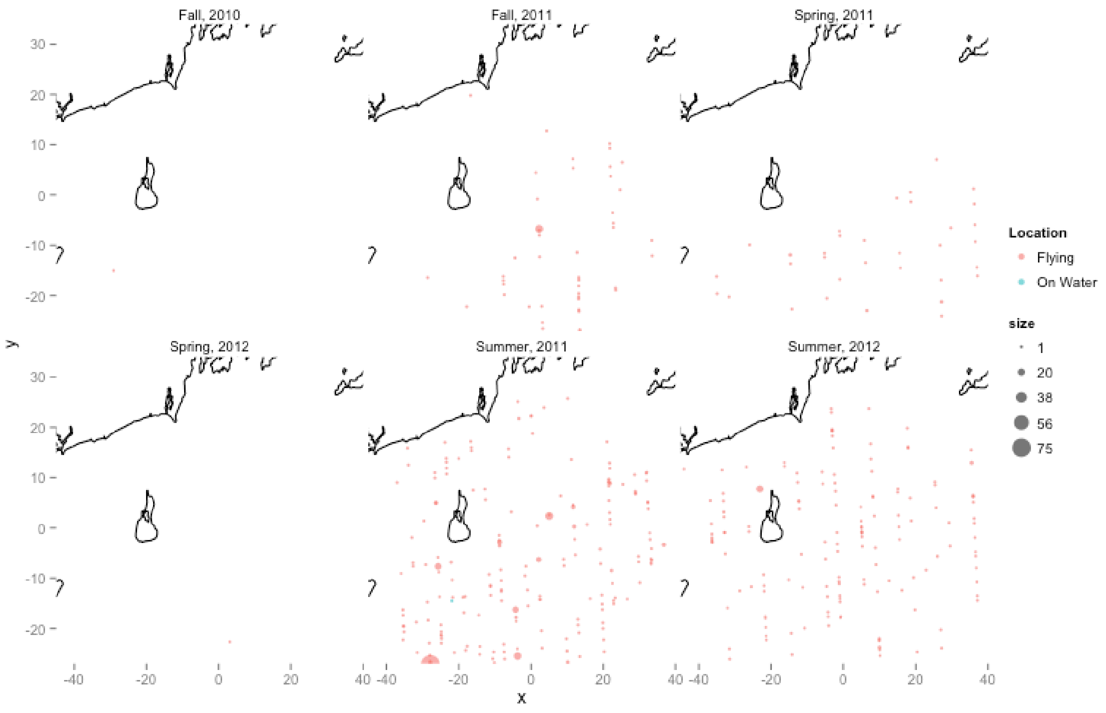
How many are on the water versus flying...

```
table(obs.petrel$Location)
```

```
##
##   Flying On Water
##      455        1
```

Note that almost all of the observations are flying, but since storm-petrels fly very close to the surface of the water, we assume that the distances are recorded correctly.

```
plot.uri.data(obs.petrel, facet = eval(parse(text = "Season~Year")))
```



Storm-petrel observations, plotted according to season and year.

what about the tabulation:

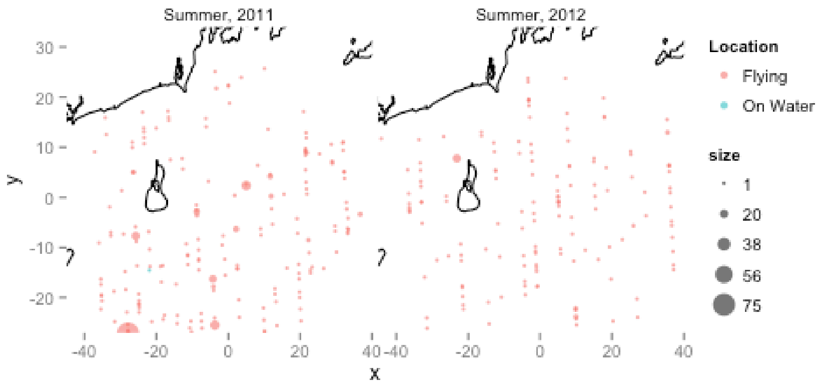
```
table(obs.petrel$SeasonYear)
```

```
##
##   Fall2010   Fall2011 Spring2011 Spring2012 Summer2011 Summer2012
##          1         52         35          5        209        154
## Winter1011 Winter1112
##          0          0
```

Summer has the most observations, so let's model that...

```
obs.petrel <- obs.petrel[obs.petrel$Season == "Summer", ]
effort <- effort[effort$Season == "Summer", ]
```

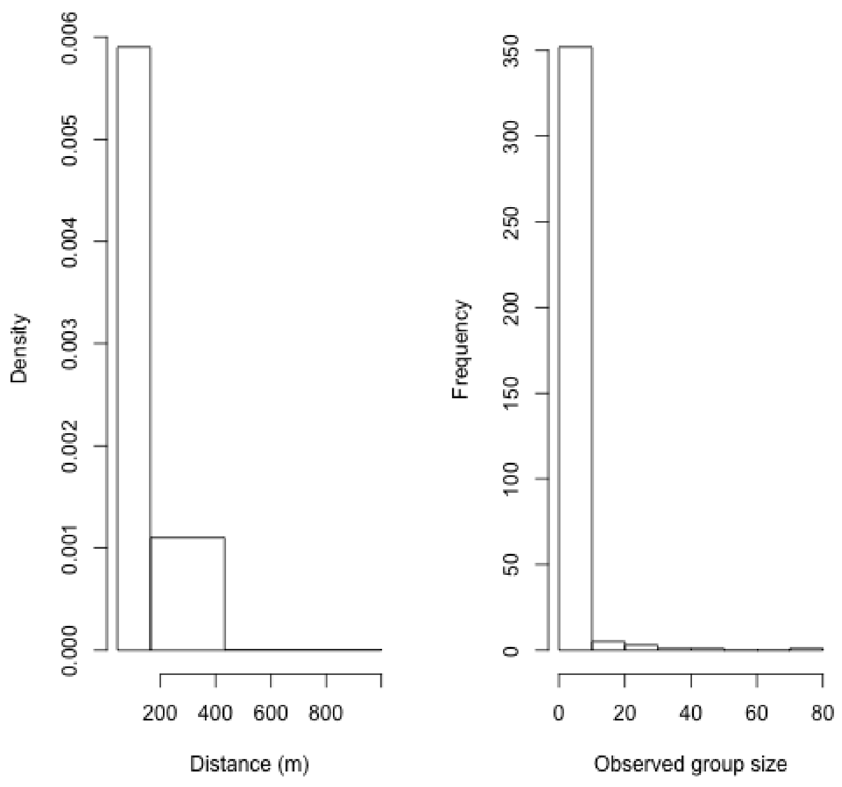
```
plot.uri.data(obs.petrel, facet = eval(parse(text = "Season~Year")))
```



Summer storm-petrel observations, plotted according to season and year.

Plotting a histogram of distances and group sizes:

```
par(mfrow = c(1, 2))
hist(obs.petrel$distance, breaks = sort(unique(c(obs.petrel$distbegin, obs.petrel$distend))),
    main = "", xlab = "Distance (m)")
hist(obs.petrel$size, main = "", xlab = "Observed group size")
```



Histogram of storm-petrel observed distances and group sizes.

Detection function

Fitting some detection functions to the distance data:

```
hn.df <- ds(obs.petrel, adjustment = NULL, truncation = list(left = 44, right = 1000))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting half-normal key function AIC= 458.324 No survey area information
## supplied, only estimating detection function.
```

```
hr.df <- ds(obs.petrel, adjustment = NULL, truncation = list(left = 44, right = 1000),
    key = "hr")
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function AIC= 461.229 No survey area information
## supplied, only estimating detection function.
```

```
obs.petrel$ssize <- obs.petrel$size/sd(obs.petrel$size)
hr.df.size <- ds(obs.petrel, adjustment = NULL, truncation = list(left = 44,
    right = 1000), key = "hr", formula = ~ssize)
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function AIC= 463.195 No survey area information
## supplied, only estimating detection function.
```

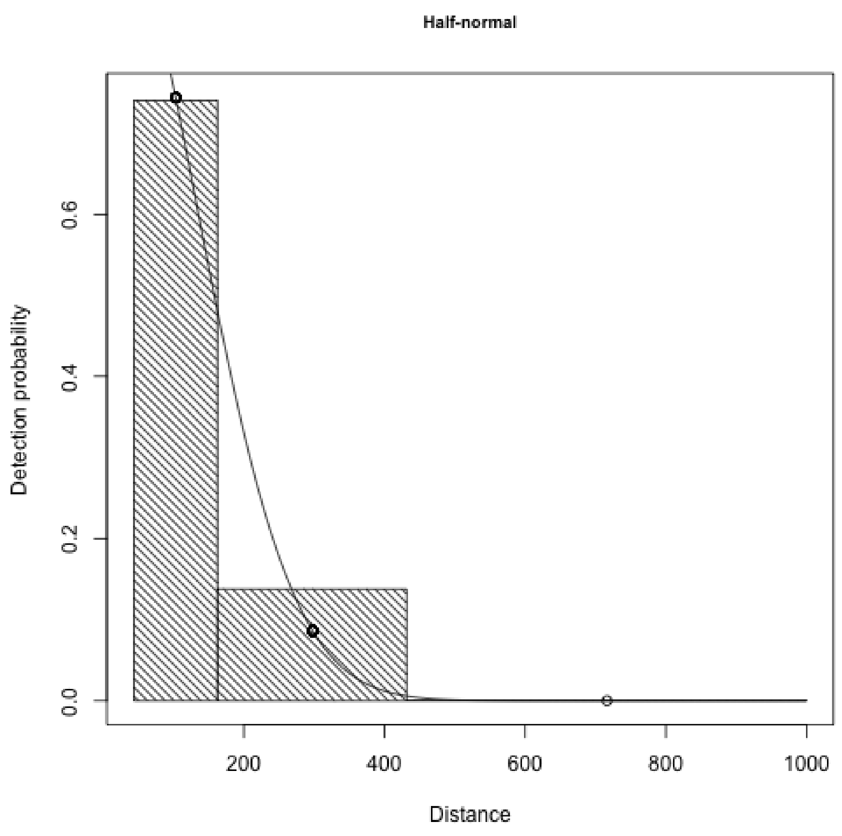
```
hn.df.size <- ds(obs.petrel, adjustment = NULL, truncation = list(left = 44,
    right = 1000), formula = ~ssize)
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting half-normal key function AIC= 458.577 No survey area information
## supplied, only estimating detection function.
```

	Detection function	Adjustments	Covariates	AIC	Δ AIC	# pars	p	CV(p)	C-vM p	KS p
1	hn			458.324	0	1	0.125	0.056	0.008	0
2	hn		ssize	458.577	0.253	2	0.124	0.062	0.007	0
3	hr			461.229	2.905	2	0.169	0.039	0.008	0
4	hr		ssize	463.195	4.871	3	0.169	0.039	0.007	0

```
plot(hn.df, main = "Half-normal")
```



Plot of half-normal detection function fitted to the storm-petrel data.

From the above plots and table, we see that the half-normal detection function is the best model in AIC terms.

Spatial modelling

First, allocating the effort correctly...

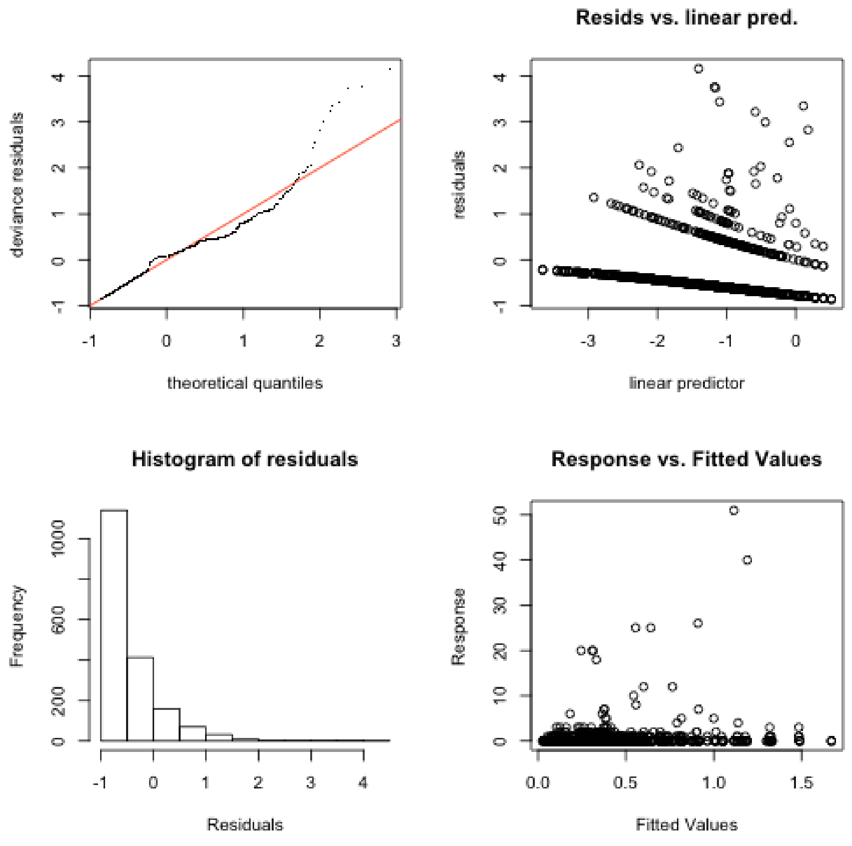
```
new.segobs <- allocate.effort(obs.petrel, seg)
obs.petrel <- new.segobs$obs
seg <- new.segobs$seg
```

And then fitting the DSM...

```
k1 <- 7
k2 <- 20
petrel.model <- dsm(N~ #s(roughness,k=k1)+
#                    s(gchl_summer,k=k1)+
#                    s(gchl_spring,k=k1)+
#                   s(phimedian,k=k1)+
#                   s(distancelandkm,k=k1)+
#                   s(depthm,k=k1)+
                  depthm+
#                   s(x,k=k1)+
                  s(y,k=k1),
                hn.df, seg, obs.petrel,
                family=negbin(theta=c(0.1,0.15)),
        select=TRUE,method="REML")
summary(petrel.model)
```

```
##
## Family: Negative Binomial(0.146)
## Link function: log
##
## Formula:
## N ~ depthm + s(y, k = k1) + offset(off.set)
## <environment: 0x11000e318>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -16.1306     0.5115  -31.53   <2e-16 ***
## depthm       -0.0451     0.0134   -3.35    8e-04 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##       edf Ref.df Chi.sq p-value
## s(y) 4.04      6   25.9 8.6e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0146   Deviance explained = 11.2%
## REML score = 1170.4  Scale est. = 1         n = 1836
```

```
gam.check(petrel.model)
```



GAM check plot for the storm-petrel DSM.

```
##
## Method: REML   Optimizer: outer newton
## full convergence after 1 iteration.
## Gradient range [1.343e-05,0.000341]
## (score 1170 & scale 1).
## Hessian positive definite, eigenvalue range [0.2856,0.9991].
##
## 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(y) 6.000 4.042   0.696    0.85
```

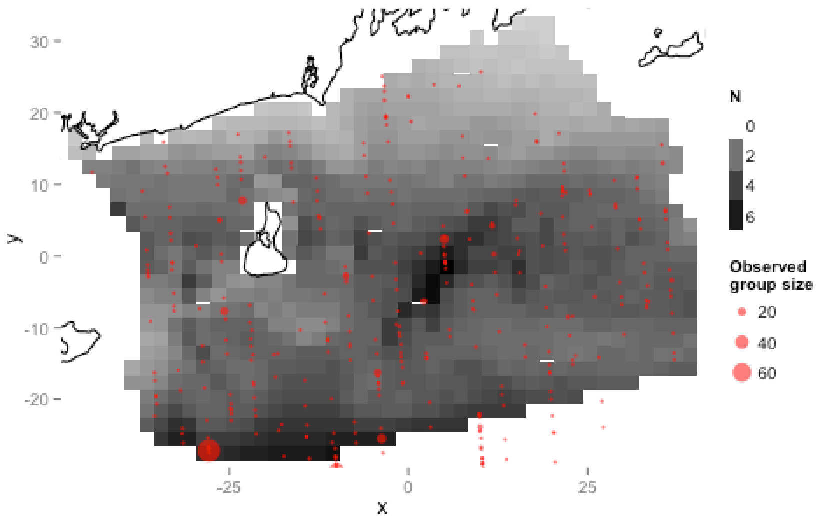
Uncertainty in the predicted abundance:

```
dsm.var.prop(petrel.model, pred, pred$cellaream)
```

```
## Summary of uncertainty in a density surface model calculated
##  by variance propagation.
##
## Quantiles of differences between fitted model and variance model
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -3.51e-05 -6.02e-06  3.70e-07  1.71e-06  3.66e-06  2.14e-04
##
## Approximate asymptotic confidence interval:
##    5% Mean  95%
## 1697 2115 2636
## (Using delta method)
##
## Point estimate                 : 2115
## Standard error                 : 238.5
## Coefficient of variation       : 0.1128
```

```
plot.preds(petrel.model, obs.petrel)
```

```
## Abundance = 2115
```



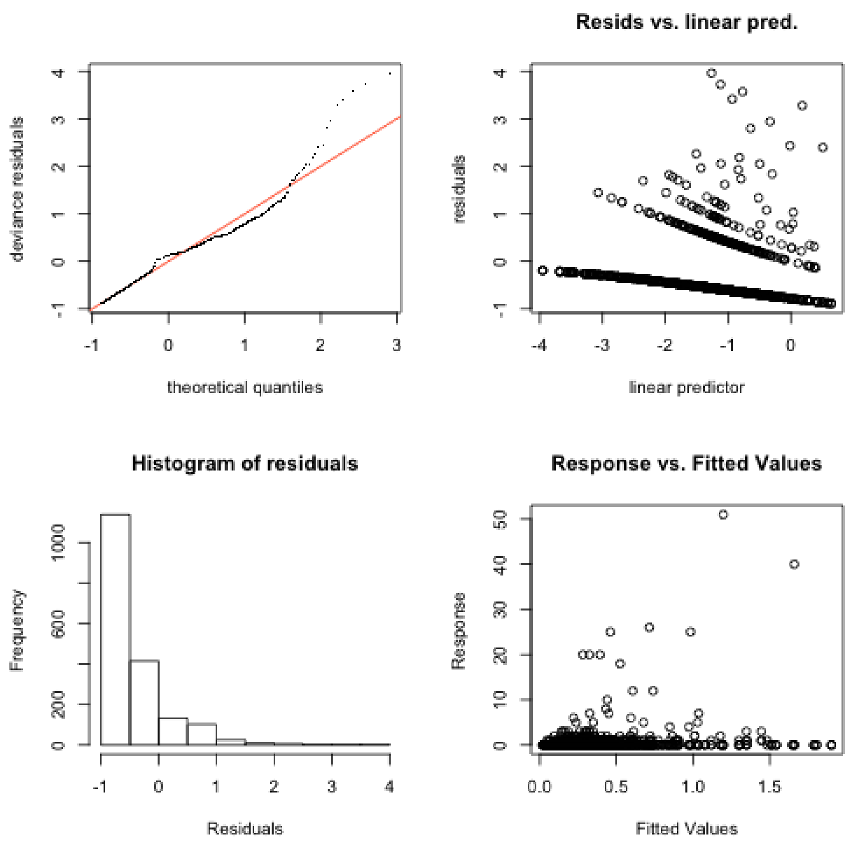
Predicted abundance surface for the storm-petrel model.

What about using the bivariate smooth of `x` and `y`

```
petrel.model.xy <- dsm(N~ #s(roughness,k=k1)+
#                    s(gchl_summer,k=k1)+
#                    s(gchl_spring,k=k1)+
#                   s(phimedian,k=k1)+
#                   s(distancelandkm,k=k1)+
#                   s(depthm,k=k1)+
                  depthm+
                  s(x,y,k=k2),
                hn.df, seg, obs.petrel,
                family=negbin(theta=c(0.14,16)),
        select=TRUE,method="REML")
summary(petrel.model.xy)
```

```
##
## Family: Negative Binomial(0.151)
## Link function: log
##
## Formula:
## N ~ depthm + s(x, y, k = k2) + offset(off.set)
## <environment: 0x10f2a5600>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -16.2632     0.5376  -30.25  < 2e-16 ***
## depthm       -0.0474     0.0141   -3.36  0.00079 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##         edf Ref.df Chi.sq p-value
## s(x,y) 12.3     19   40.5 8.4e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0166   Deviance explained = 13.8%
## REML score = 1171.7  Scale est. = 1         n = 1836
```

```
gam.check(petrel.model.xy)
```



GAM check plot for storm-petrel DSM with bivarirate smooth of location.

```
##
## Method: REML   Optimizer: outer newton
## step failed after 2 iterations.
## Gradient range [-0.005016,0.0003915]
## (score 1172 & scale 1).
## Hessian positive definite, eigenvalue range [0.5349,2.774].
##
## 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(x,y) 19.000 12.301   0.707     0.9
```

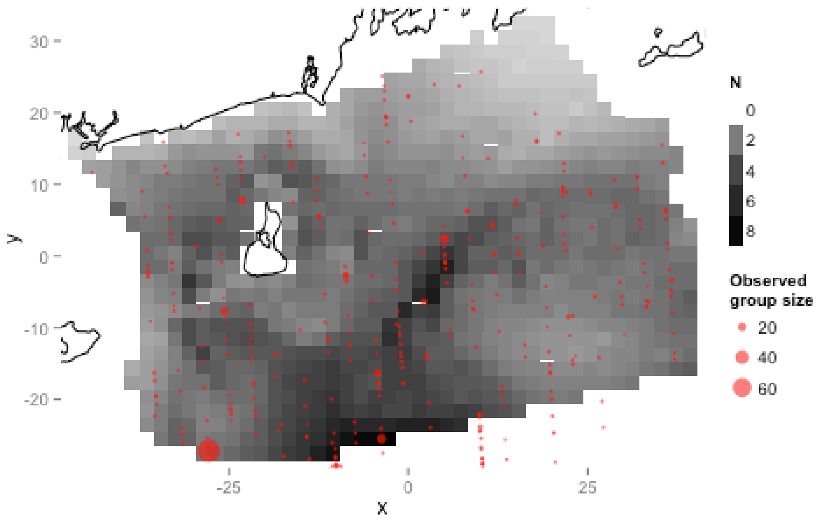
Uncertainty in the model, along with the predicted abundance:

```
dsm.var.prop(petrel.model.xy, pred, pred$cellaream)
```

```
## Summary of uncertainty in a density surface model calculated
##  by variance propagation.
##
## Quantiles of differences between fitted model and variance model
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -1.93e-06 -2.97e-07 -7.60e-08  6.00e-08  2.87e-07  9.62e-06
##
## Approximate asymptotic confidence interval:
##   5% Mean  95%
## 1666 2076 2588
## (Using delta method)
##
## Point estimate                 : 2076
## Standard error                 : 234.1
## Coefficient of variation       : 0.1127
```

```
plot.preds(petrel.model.xy, obs.petrel)
```

```
## Abundance = 2076
```



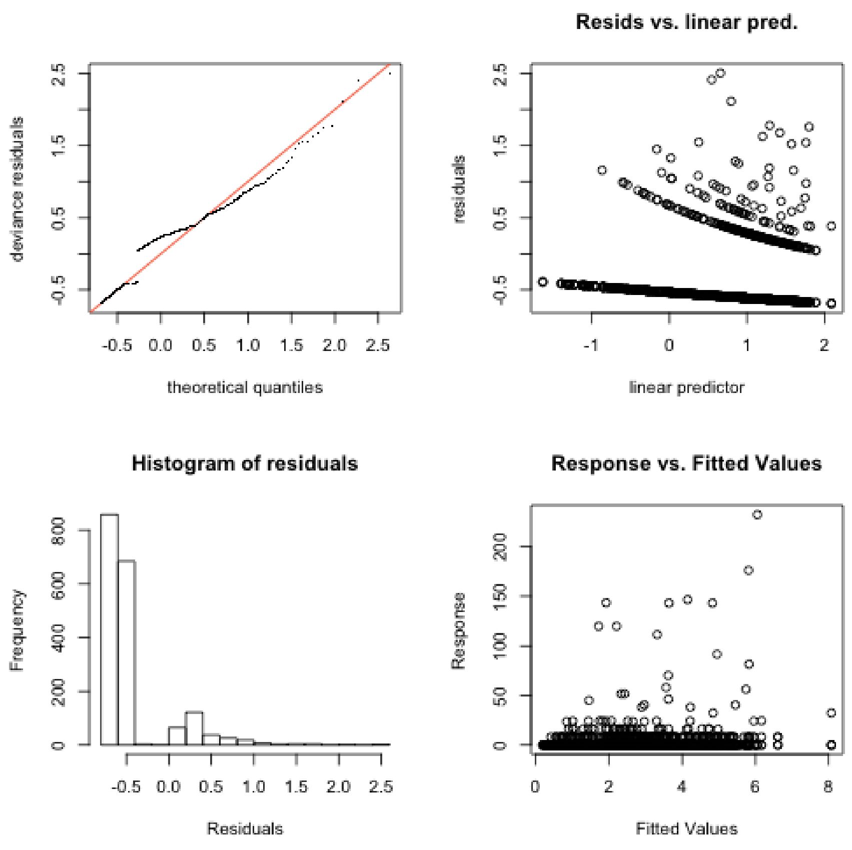
Predicted abundance surface for the storm-petrel model with bivariate smooth of location.

The Q-Q plot here does not look great, perhaps due to size bias. The detection function with size as a covariate was very close in AIC terms. So, refitting the model with observed group size as a covariate in the detection function...

```
petrel.size.model <- dsm(Nhat~ #s(roughness,k=k1)+
#                   s(gchl_summer,k=k1)+
#                   s(gchl_spring,k=k1)+
#                  s(phimedian,k=k1)+
#                  s(distancelandkm,k=k1)+
#                  s(depthm,k=k1)+
                  depthm+
#                  s(x,k=k1)+
                  s(y,k=k1),
                hn.df.size, seg, obs.petrel,
                family=negbin(theta=0.046),
#                family=negbin(theta=c(0.01,1)),
        select=TRUE,method="REML")
summary(petrel.size.model)
```

```
##
## Family: Negative Binomial(0.046)
## Link function: log
##
## Formula:
## Nhat ~ depthm + s(y, k = k1) + offset(off.set)
## <environment: 0x10f7366d8>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -16.4155     0.6272  -26.17   <2e-16 ***
## depthm       -0.0511     0.0167   -3.07   0.0021 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##       edf Ref.df Chi.sq p-value
## s(y) 2.34      6   11.6  0.0021 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0134   Deviance explained = 4.96%
## REML score = 1979.1  Scale est. = 1         n = 1836
```

```
gam.check(petrel.size.model)
```



GAM check plot for the storm-petrel DSM with size as a covariate in the detection function.

```
##
## Method: REML   Optimizer: outer newton
## full convergence after 4 iterations.
## Gradient range [-0.0004679,-0.0002997]
## (score 1979 & scale 1).
## Hessian positive definite, eigenvalue range [0.1001,0.5651].
##
## 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(y) 6.000 2.338   0.508       1
```

This looks a little better for the larger residuals. How different is the predicted abundance and uncertainty?

```
dsm.var.gam(petrel.size.model, pred, pred$cellaream)
```

```
## Summary of uncertainty in a density surface model calculated
##  analytically for GAM, with delta method
##
## Approximate asymptotic confidence interval:
##    5% Mean  95%
## 1580 2057 2679
## (Using delta method)
##
## Point estimate                 : 2057
## Standard error                 : 248.1
## CV of detection function       : 0.06155
## CV from GAM                    : 0.1206
## Total coefficient of variation : 0.1354
```

Slightly lower, but the fit is much better. Trying with a bivariate smooth of location, as above:

```
petrel.size.model.xy <- dsm(Nhat~ #s(roughness,k=k1)+
                   #s(gchl_summer,k=k1)+
                   #s(gchl_spring,k=k1)+
                  #s(phimedian,k=k1)+
                  #s(distancelandkm,k=k1)+
                  #s(depthm,k=k1)+
                  depth,#+
                  #s(x,y,k=k2),
                hn.df.size, seg, obs.petrel,
#                family=negbin(theta=0.046),
                family=negbin(theta=c(0.01,1)),
        select=TRUE,method="REML")
summary(petrel.size.model.xy)
```

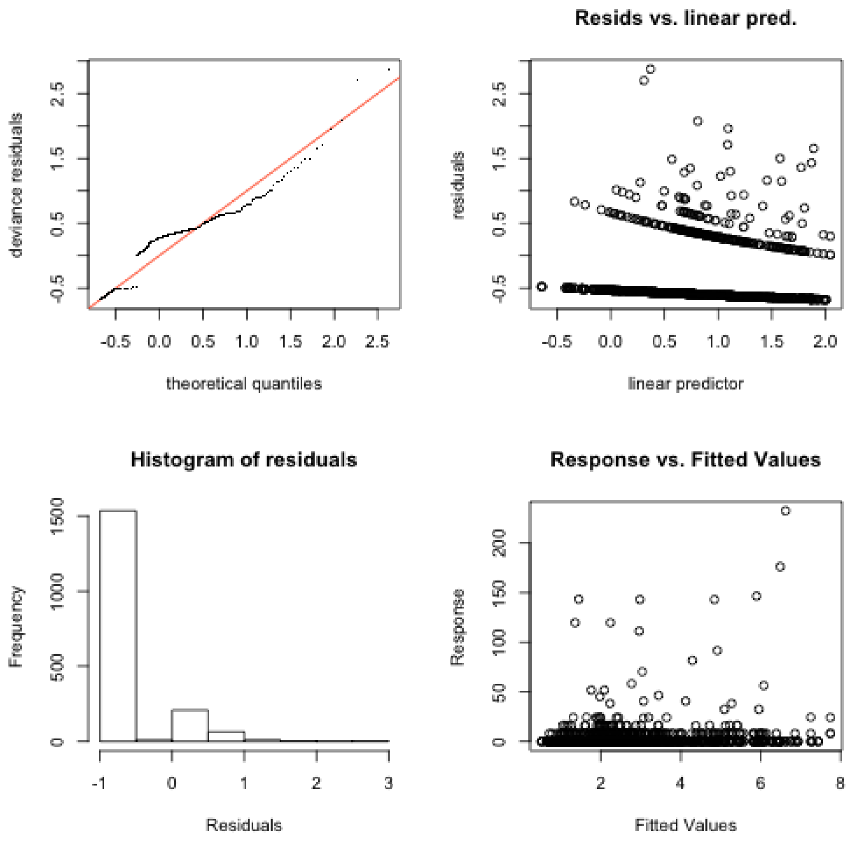
```
##
## Family: Negative Binomial(0.045)
## Link function: log
##
## Formula:
## Nhat ~ depth + offset(off.set)
## <environment: 0x10f084ad0>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -16.32299    0.42701  -38.23  < 2e-16 ***
## depth        -0.01539    0.00338   -4.56  5.2e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## R-sq.(adj) =  0.0141   Deviance explained = 3.13%
## REML score = 1982.6  Scale est. = 1         n = 1836
```

```
gam.check(petrel.size.model.xy)
```

```
##
## Method: REML   Optimizer: outer newton
##  after 0 iterations.
## Warning: no non-missing arguments to min; returning Inf Warning: no
## non-missing arguments to max; returning -Inf
```

```
##
## Gradient range [Inf,-Inf]
## (score 1983 & scale 1).
```

```
## Error: 'data' must be of a vector type, was 'NULL'
```

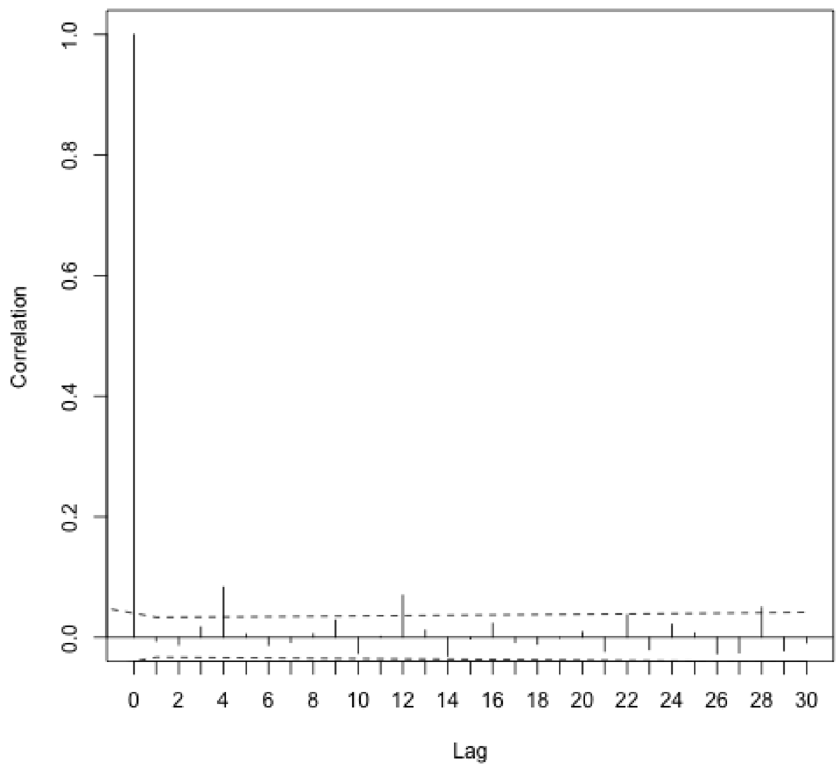


GAM check plot for storm-petrel DSM with bivarirate smooth of location and size as a covariate in the detection function.

So, interestingly both models with bivariate smoothers ended up with the same covariate (only linear depth effect) and without the bivariate smooth both had a linear depth term and a smooth of `y`. This leads us to believe that the models without the bivariate smooth should be preferred. We also see that the model with size as a covariate in the deteciton function has a much better Q-Q plot. We therefore choose this model.

Plotting the autocorrelogram for our chosen model:

```
dsm.cor(petrel.size.model, max.lag = 30)
```



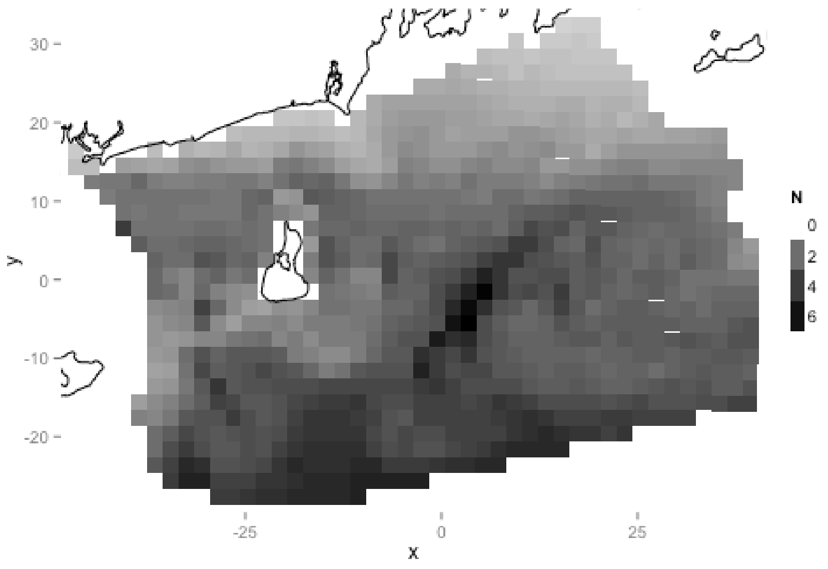
Autocorrelogram for our final model

we see there is very little residual autocorrelation.

Finally, plotting the predicted abundance surface and associate uncertainty:

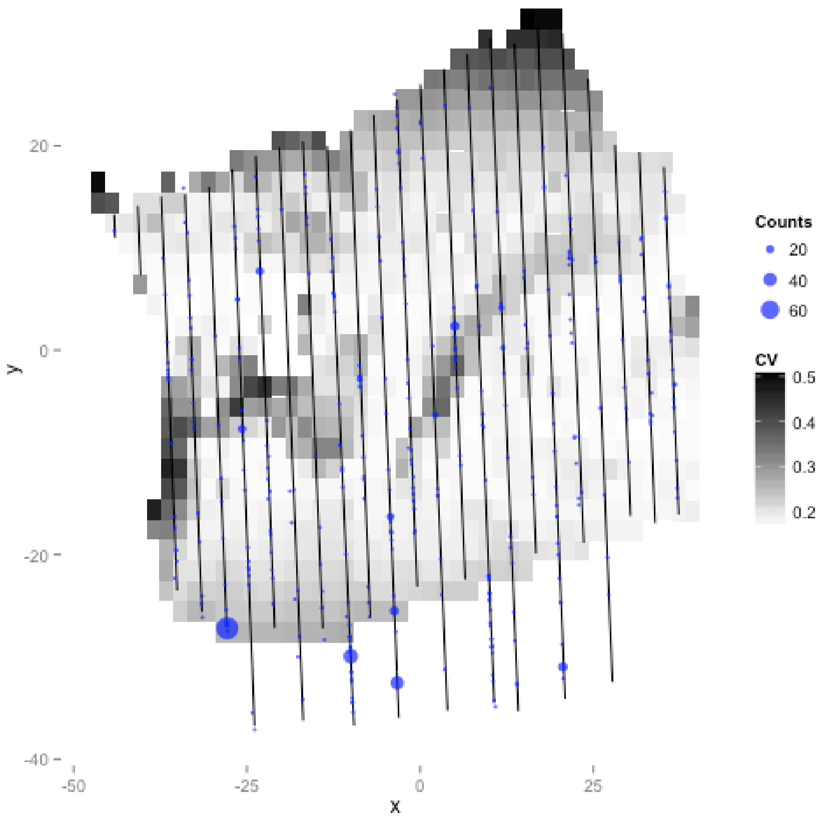
```
plot.preds(petrel.size.model)
```

```
## Abundance = 2057
```



Plot of the predicted abundance for storm-petrels

```
pred$height <- pred$width <- 2
pred.grid <- split(pred, 1:nrow(pred))
petrel.var.grid <- dsm.var.gam(petrel.size.model, pred.grid, pred$cellarea)
plot(petrel.var.grid)
```



Plot of the predicted abundance for storm-petrels

Finish up

Save the analyses

```
save.image("petrel-models.RData")
```

Save predictions for Zonation:

```
petrel.preds <- predict(petrel.size.model, pred, pred$cellaream)


petrel.pred.data <- data.frame(cellid = 1:920, pred = petrel.preds, var = diag(petrel.var.grid$pred.var),
    cv = sqrt(diag(petrel.var.grid$pred.var))/petrel.preds, season = rep("Summer",
        920))
save(petrel.pred.data, file = "petrel-preds.RData")
```

Alcids

This document is a record of modelling for the alcid data from the URI aerial line transect survey of the OSAMP area off the coast of Rhode Island. It is provided as a `knitr` (Xie 2013) file, that includes all the necessary code to re-create the analysis.

Preamble

Load some data and packages

```
suppressPackageStartupMessages(library(osampuri))
data("uri-lt-data")
```

Subset the data to include only alcids:

```
obs.alcids <- obs[obs$Group == "Alcid", ]
rm(obs)
```

Remove the 3 observations with no recorded distances

```
obs.alcids <- obs.alcids[obs.alcids$Bin != "Not recorded", ]
```

... and the 143 birds which were not on the water

```
obs.alcids <- obs.alcids[obs.alcids$Location == "On Water", ]
```

Finally, only include the 290 observations in winter (there are 1025spring observations):

```
obs.alcids <- obs.alcids[obs.alcids$Season == "Winter", ]
effort <- effort[effort$Season == "Winter", ]
```

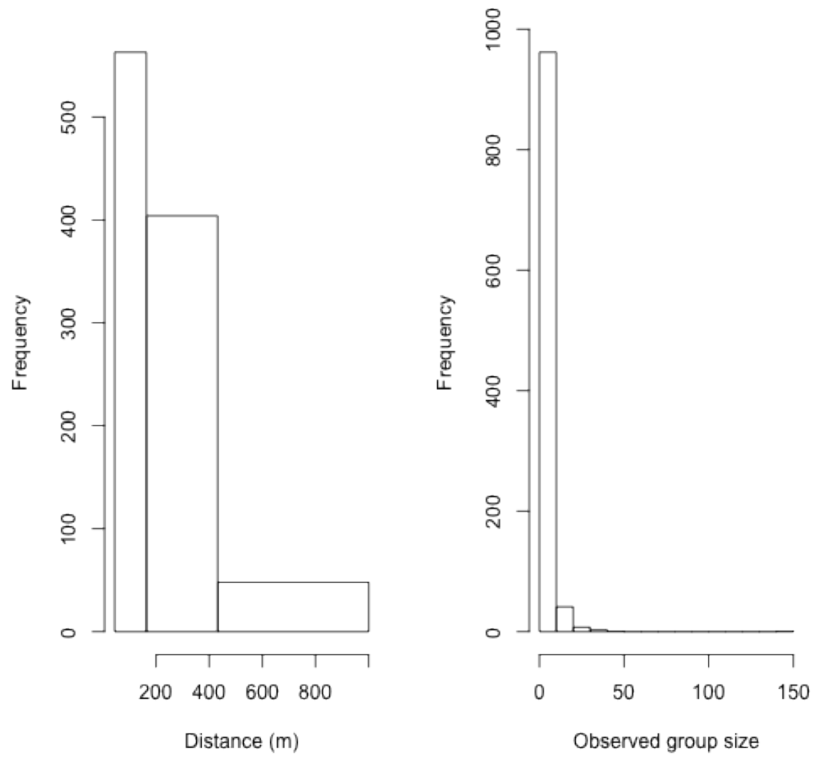
Detection function analysis

Before fitting a detection function, we first plot a histogram of distances and group sizes:

```
par(mfrow = c(1, 2))
hist(obs.alcids$distance, breaks = sort(unique(c(obs.alcids$distbegin, obs.alcids$distend))),
    freq = TRUE, xlab = "Distance (m)", main = "")
```

```
## Warning: the AREAS in the plot are wrong -- rather use 'freq = FALSE'
```

```
hist(obs.alcids$size, freq = TRUE, xlab = "Observed group size", main = "")
```



Histogram of distances (left) and observed group sizes (right).

Fit a detection function with half-normal and hazard rate key functions with observed group size as a covariate:

```
hn.df <- ds(obs.alcids, adjustment = NULL, truncation = list(right = 1000, left = 44))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting half-normal key function AIC= 1728.749 No survey area information
## supplied, only estimating detection function.
```

```
hr.df <- ds(obs.alcids, adjustment = NULL, truncation = list(right = 1000, left = 44),
    key = "hr")
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function AIC= 1714.031 No survey area information
## supplied, only estimating detection function.
```

```


obs.alcids$ssize <- obs.alcids$size/sd(obs.alcids$size)


hn.df.size <- ds(obs.alcids, adjustment = NULL, truncation = list(right = 1000,
    left = 44), formula = ~ssize)
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
## Fitting half-normal key function AIC= 1711.558 No survey area information
## supplied, only estimating detection function.
```

```
hr.df.size <- ds(obs.alcids, adjustment = NULL, truncation = list(right = 1000,
    left = 44), key = "hr", formula = ~ssize)
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function AIC= 1698.642 No survey area information
## supplied, only estimating detection function.
```

```
plot(hr.df, pl.den = 0, xlab = "Distance (m)")
```

```
## Error: formal argument "xlab" matched by multiple actual arguments
```

How many observations in each species?

```
table(obs.alcids$Species)[table(obs.alcids$Species) > 0]
```

```
##
##      Alcid       COMU       DOVE Murre spp.       RAZO
##        581        262         64         23         85
```

Dovekies

Dovekies are significantly smaller than the other species, so their probability of detection might be rather different:

```
hn.df.do <- ds(obs.alcids[obs.alcids$Species == "DOVE", ], adjustment = NULL,
    truncation = list(right = 1000, left = 44))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting half-normal key function AIC= 76.677 No survey area information
## supplied, only estimating detection function.
```

```
hr.df.do <- ds(obs.alcids[obs.alcids$Species == "DOVE", ], adjustment = NULL,
    truncation = list(right = 1000, left = 44), key = "hr")
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function AIC= 78.884 No survey area information
## supplied, only estimating detection function.
```

Indeed the models give the following average probabilities of detection:

```
fitted(hn.df.do$ddf)[1]
```

```
##      1
## 0.1135
```

```
fitted(hr.df.do$ddf)[1]
```

```
##      1
## 0.1633
```

which are quite different from those given above. This indicates it might be useful to include species as a covariate in the detection function.

Species as a covariate?

So, fitting both species and size and species as covariates:

```
hn.df.sp <- ds(obs.alcids, adjustment = NULL, truncation = list(right = 1000,
    left = 44), formula = ~as.factor(Species))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting half-normal key function AIC= 1650.549 No survey area information
## supplied, only estimating detection function.
```

```
hr.df.sp <- ds(obs.alcids, adjustment = NULL, truncation = list(right = 1000,
    left = 44), key = "hr", formula = ~as.factor(Species))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function AIC= 1659.802 No survey area information
## supplied, only estimating detection function.
```

```
hn.df.size.sp <- ds(obs.alcids, adjustment = NULL, truncation = list(right = 1000,
    left = 44), formula = ~ssize + as.factor(Species))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting half-normal key function AIC= 1640.892 No survey area information
## supplied, only estimating detection function.
```

```
hr.df.size.sp <- ds(obs.alcids, adjustment = NULL, truncation = list(right = 1000,
    left = 44), key = "hr", formula = ~ssize + as.factor(Species))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function AIC= 1649.347 No survey area information
## supplied, only estimating detection function.
```

Table of results

	Detection function	Adjustments	Covariates	AIC	Δ AIC	# pars	p	CV(p)	C-vM p	KS p
1	hn		ssize+Species	1640.892	0	6	0.179	0.035	0.002	0
2	hr		ssize+Species	1649.347	8.455	7	0.211	0.037	0.004	0
3	hn		Species	1650.549	9.657	5	0.181	0.034	0.003	0
4	hr		Species	1659.802	18.91	6	0.211	0.038	0.006	0
5	hr		ssize	1698.642	57.75	3	0.212	0.034	0.027	0
6	hn		ssize	1711.558	70.666	2	0.198	0.027	0.021	0
7	hr			1714.031	73.139	2	0.212	0.034	0.037	0
8	hn			1728.749	87.857	1	0.202	0.027	0.031	0

```
par(mfrow = c(2, 3))
plot(hr.df.size.sp, main = "all", xlab = "Distance(m)")
```

```
## Error: formal argument "xlab" matched by multiple actual arguments
```

```
plot(hr.df.size.sp, subset = Species == "COMU", main = "COMU", xlab = "Distance(m)")
```

```
## Error: formal argument "xlab" matched by multiple actual arguments
```

```
plot(hr.df.size.sp, subset = Species == "DOVE", main = "DOVE", xlab = "Distance(m)")
```

```
## Error: formal argument "xlab" matched by multiple actual arguments
```

```
plot(hr.df.size.sp, subset = Species == "RAZO", main = "RAZO", xlab = "Distance(m)")
```

```
## Error: formal argument "xlab" matched by multiple actual arguments
```

```
plot(hr.df.size.sp, subset = Species == "Murre spp.", main = "Murre spp.", xlab = "Distance(m)")
```

```
## Error: formal argument "xlab" matched by multiple actual arguments
```

```
plot(hr.df.size.sp, subset = Species == "Alcid", main = "Alcid", xlab = "Distance(m)")
```

```
## Error: formal argument "xlab" matched by multiple actual arguments
```

Allocating effort

```
new.segobs <- allocate.effort(obs.alcids, seg)
obs.alcids <- new.segobs$obs
seg <- new.segobs$seg
```

To make plotting look nicer, here are some options...

```
## (rough) dummy width and height
pred$width <- rep(2, nrow(pred))
pred$height <- rep(2, nrow(pred))

p.opts.geo <- theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
    panel.background = element_blank(), strip.background = element_blank(),
    legend.key = element_blank(), aspect.ratio = 1)
```

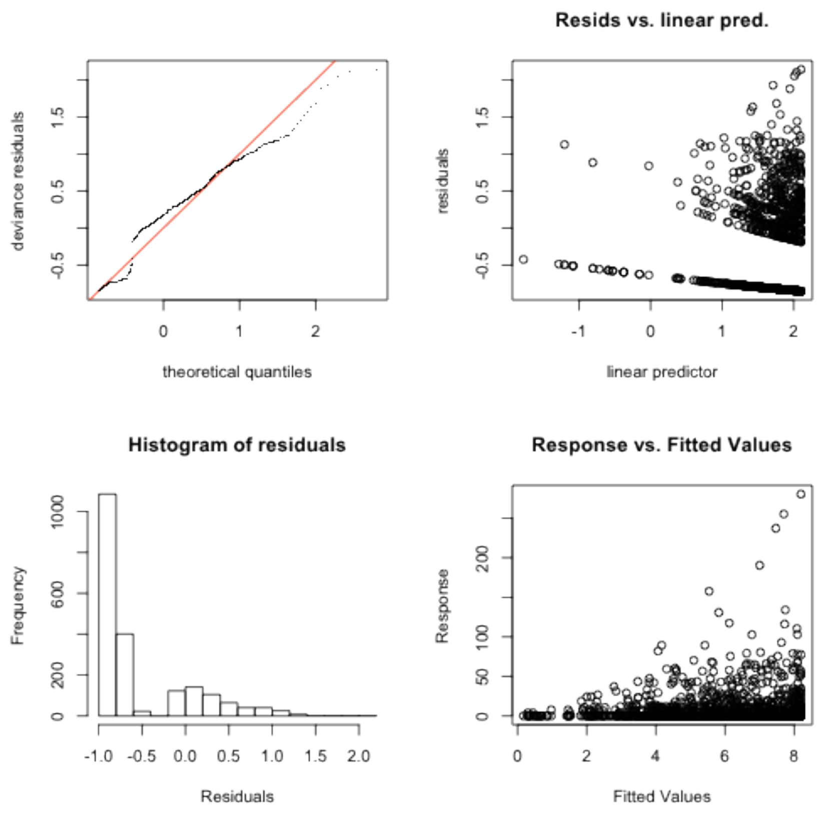
Full data density surface model

We now fit a DSMs using the best fitting detection function for all data (that inlcuded covariates for both species and group size).

```
k1 <- 7
k2 <- 15
alcid.all.covars.nb <- dsm(Nhat~ #s(roughness,k=k1)+
                                 #s(gchl_winter,k=k1)+
                                 #s(phimedian,k=k1)+
                                 #s(distancelandkm,k=k1)+
                                 s(gchl_fall,k=k1),#+
                                 #s(depthm,k=k1)+
                                 #s(x,k=k1)+
                                 #s(y,k=k1),
                         hr.df.size.sp, seg, obs.alcids,
                         family=negbin(theta=c(0.01,0.1)),
                         select=TRUE,method="REML")
summary(alcid.all.covars.nb)
```

```
##
## Family: Negative Binomial(0.078)
## Link function: log
##
## Formula:
## Nhat ~ s(gchl_fall, k = k1) + offset(off.set)
## <environment: 0x12b1fd000>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -13.5243     0.0796    -170   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##               edf Ref.df Chi.sq p-value
## s(gchl_fall) 2.76      6   24.3 3.6e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.00908   Deviance explained = 1.84%
## REML score = 3590.1  Scale est. = 1         n = 2067
```

```
gam.check(alcid.all.covars.nb)
```



plot of chunk alcid-gamcheck

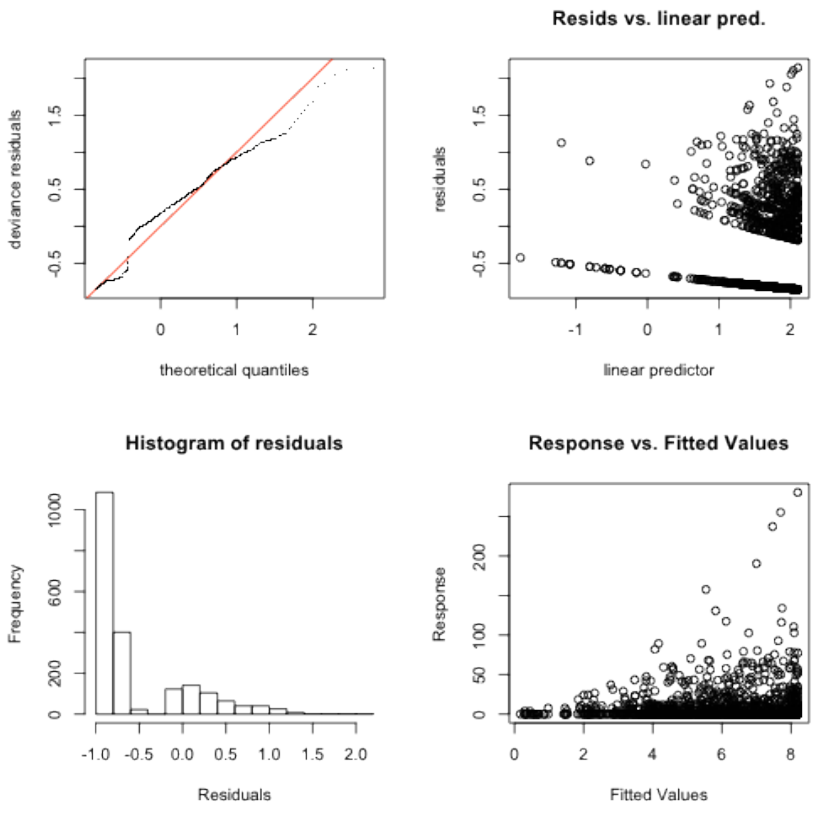
```
##
## Method: REML   Optimizer: outer newton
## full convergence after 1 iteration.
## Gradient range [-0.0025,-4.477e-05]
## (score 3590 & scale 1).
## Hessian positive definite, eigenvalue range [0.002481,0.8364].
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                  k'   edf k-index p-value
## s(gchl_fall) 6.000 2.756   0.563    0.82
```

Including a bivariate smooth of location:

```
k1 <- 7
k2 <- 15
alcid.all.covars.nb.xy <- dsm(Nhat~ #s(roughness,k=k1)+
                                    #s(gchl_winter,k=k1)+
                                    #s(phimedian,k=k1)+
                                    #s(distancelandkm,k=k1)+
                                    s(gchl_fall,k=k1),#+
                                    #s(depthm,k=k1)+
                                    #s(x,y,k=k2),
                         hr.df.size.sp, seg, obs.alcids,
                         family=negbin(theta=c(0.01,0.1)),
                         select=TRUE,method="REML")
summary(alcid.all.covars.nb.xy)
```

```
##
## Family: Negative Binomial(0.078)
## Link function: log
##
## Formula:
## Nhat ~ s(gchl_fall, k = k1) + offset(off.set)
## <environment: 0x10c6b7310>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -13.5243     0.0796    -170   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##               edf Ref.df Chi.sq p-value
## s(gchl_fall) 2.76      6   24.3 3.6e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.00908   Deviance explained = 1.84%
## REML score = 3590.1  Scale est. = 1         n = 2067
```

```
gam.check(alcid.all.covars.nb.xy)
```



plot of chunk alcid-gamcheckxy

```
##
## Method: REML   Optimizer: outer newton
## full convergence after 1 iteration.
## Gradient range [-0.0025,-4.477e-05]
## (score 3590 & scale 1).
## Hessian positive definite, eigenvalue range [0.002481,0.8364].
##
## 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(gchl_fall) 6.000 2.756   0.563    0.84
```

Both options result in the same model, both with relatively disappointing diagnostic plots. Trying a Tweedie response instead...

```
par(mfrow = c(3, 3))
for (tw.p in seq(1.1, 1.9, by = 0.1)) {
    alcid.all.covars.tw <- dsm(Nhat ~ s(roughness, k = k1) + s(gchl_winter,
        k = k1) + s(phimedian, k = k1) + s(distancelandkm, k = k1) + s(gchl_fall,
        k = k1) + s(depthm, k = k1) + s(x, k = k1) + s(y, k = k1), hr.df.size.sp,
        seg, obs.alcids, family = Tweedie(p = tw.p), select = TRUE, method = "REML")
    cat("p=", tw.p, " AIC=", AIC(alcid.all.covars.tw), "\n")
    qq.gam(alcid.all.covars.tw, main = paste0("p=", tw.p))
}
```

```
## p= 1.1  AIC= 8435
```

```
## p= 1.2  AIC= 7299
```

```
## p= 1.3  AIC= 7058
```

```
## p= 1.4  AIC= 7033
```

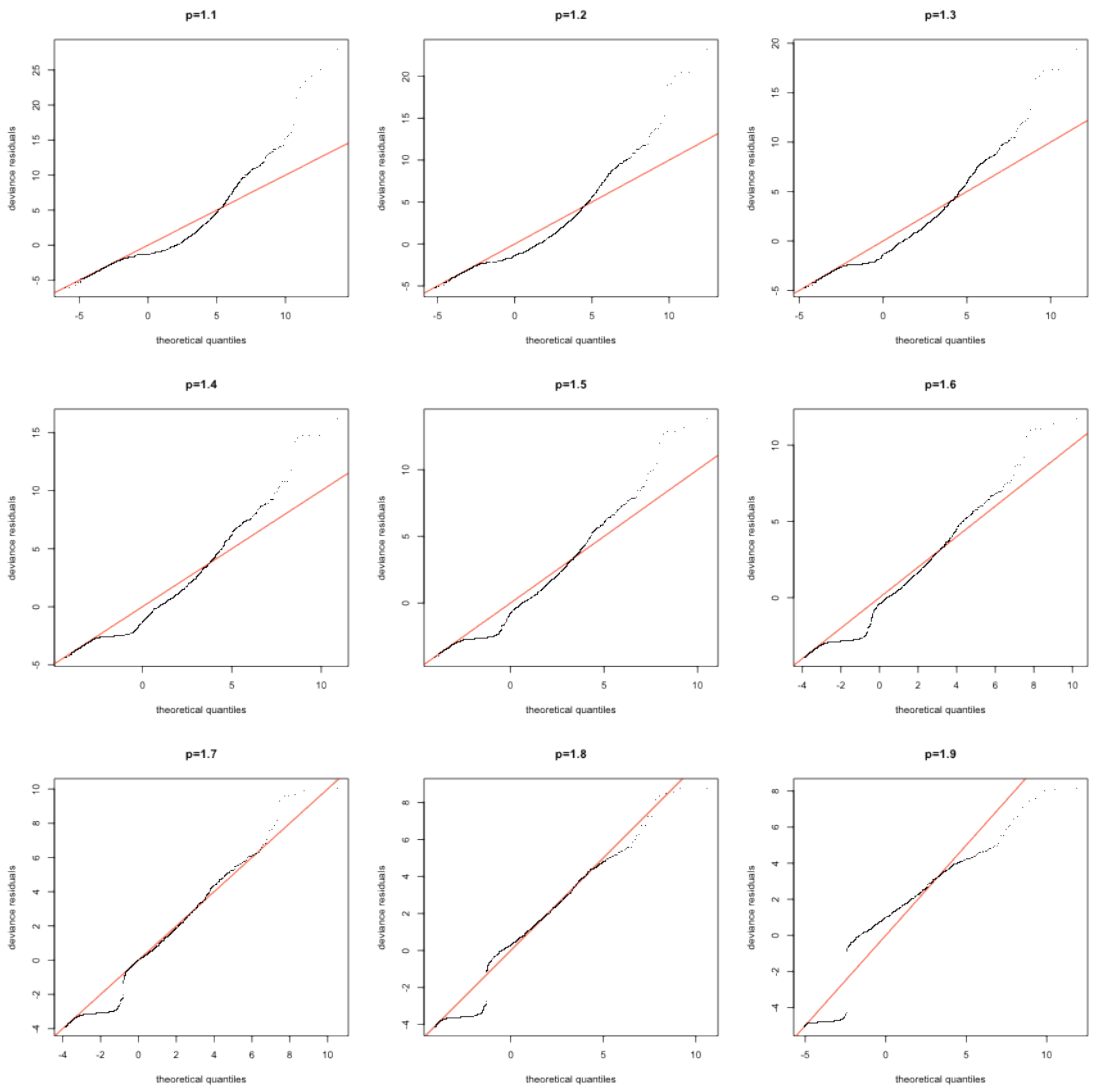
```
## p= 1.5  AIC= 7100
```

```
## p= 1.6  AIC= 7229
```

```
## p= 1.7  AIC= 7431
```

```
## p= 1.8  AIC= 7746
```

```
## p= 1.9  AIC= 8336
```



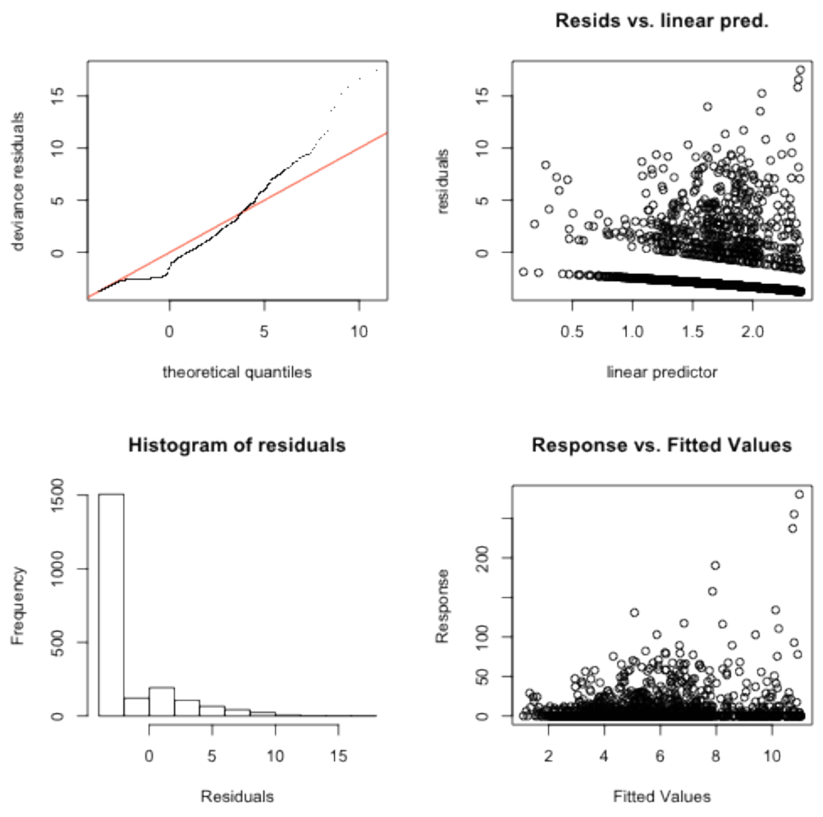
Q-Q plots for Tweedie models

Looks like setting `p=1.4` gives the best Q-Q plot and that this looks much better than that for the negative binomial. So, fitting that model with separate smooths for `x` and `y` ...

```
alcid.all.covars.tw <- dsm(Nhat~ #s(roughness,k=k1)+
                             #s(gchl_winter,k=k1)+
                             #s(phimedian,k=k1)+
                             #s(distancelandkm,k=k1)+
                             #s(gchl_fall,k=k1)+
                             #s(depthm,k=k1)+
                             #s(x,y,k=k2)+
                             s(x,k=k1)+
                             s(y,k=k1),
                          hr.df.size.sp, seg, obs.alcids,
                          family=Tweedie(p=1.4),
                          select=TRUE,method="REML")
summary(alcid.all.covars.tw)
```

```
##
## Family: Tweedie(1.4)
## Link function: log
##
## Formula:
## Nhat ~ s(x, k = k1) + s(y, k = k1) + offset(off.set)
## <environment: 0x108ed03c8>
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.5306     0.0444    -305   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##        edf Ref.df     F p-value
## s(x) 3.22      6 5.08 2.5e-07 ***
## s(y) 2.59      6 4.37 9.8e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0136   Deviance explained = 3.43%
## REML score = 3482.1  Scale est. = 11.172    n = 2067
```

```
gam.check(alcid.all.covars.tw)
```



GAM check plot for the DSM of all data, using a detection function with observed group size and species, Tweedie response and no bivaraite smooth of location

```
##
## Method: REML   Optimizer: outer newton
## full convergence after 6 iterations.
## Gradient range [-0.000123,6.323e-06]
## (score 3482 & scale 11.17).
## Hessian positive definite, eigenvalue range [4.438e-05,1156].
##
## 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(x) 6.000 3.221   0.807    0.92
## s(y) 6.000 2.594   0.784    0.55
```

```
par(mfrow = c(3, 3))
for (tw.p in seq(1.1, 1.9, by = 0.1)) {
    alcid.all.covars.tw.xy <- dsm(Nhat ~ s(roughness, k = k1) + s(gchl_winter,
        k = k1) + s(phimedian, k = k1) + s(distancelandkm, k = k1) + s(gchl_fall,
        k = k1) + s(depthm, k = k1) + s(x, k = k1) + s(y, k = k1), hr.df.size.sp,
        seg, obs.alcids, family = Tweedie(p = tw.p), select = TRUE, method = "REML")
    cat("p=", tw.p, " AIC=", AIC(alcid.all.covars.tw.xy), "\n")
    qq.gam(alcid.all.covars.tw.xy, main = paste0("p=", tw.p))
}
```

```
## p= 1.1  AIC= 8435
```

```
## p= 1.2  AIC= 7299
```

```
## p= 1.3  AIC= 7058
```

```
## p= 1.4  AIC= 7033
```

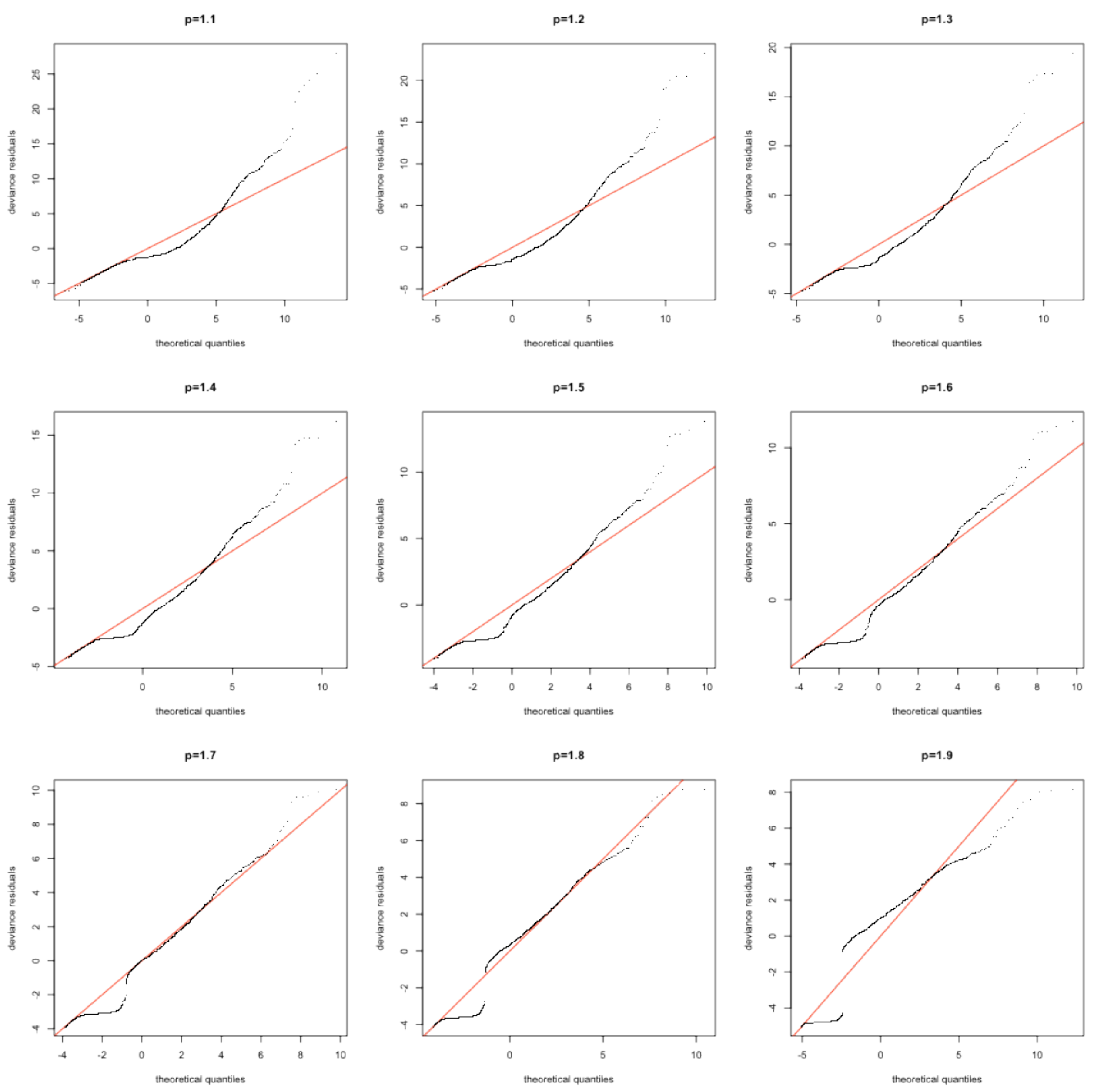
```
## p= 1.5  AIC= 7100
```

```
## p= 1.6  AIC= 7229
```

```
## p= 1.7  AIC= 7431
```

```
## p= 1.8  AIC= 7746
```

```
## p= 1.9  AIC= 8336
```

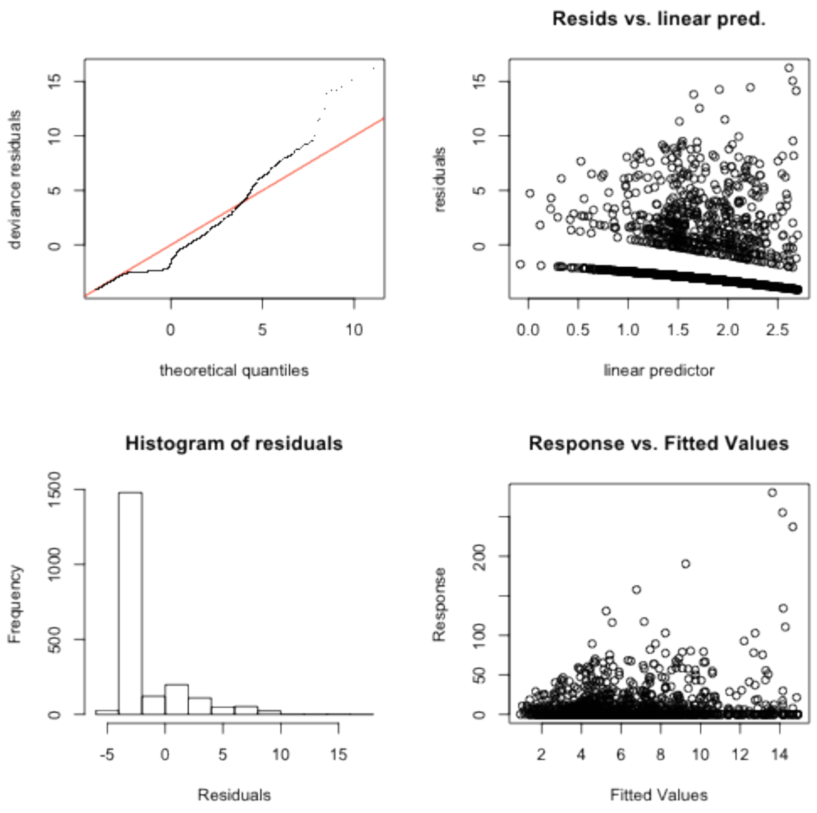


Q-Q plots for Tweedie models (bivariate smooth of location)

```
alcid.all.covars.tw.xy <- dsm(Nhat~ #s(roughness,k=k1)+
                                    #s(gchl_winter,k=k1)+
                                    #s(phimedian,k=k1)+
                                    #s(distancelandkm,k=k1)+
                                    #s(gchl_fall,k=k1)+
                                    #s(depthm,k=k1)+
                                    s(x,y,k=k2),
                         hr.df.size.sp, seg, obs.alcids,
                         family=Tweedie(p=1.4),
                         select=TRUE,method="REML")
summary(alcid.all.covars.tw.xy)
```

```
##
## Family: Tweedie(1.4)
## Link function: log
##
## Formula:
## Nhat ~ s(x, y, k = k2) + offset(off.set)
## <environment: 0x10c7b9340>
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.5759     0.0446    -304   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##         edf Ref.df     F p-value
## s(x,y) 9.52     14 8.27  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0271   Deviance explained = 6.04%
## REML score = 3466.9  Scale est. = 10.89     n = 2067
```

```
gam.check(alcid.all.covars.tw.xy)
```

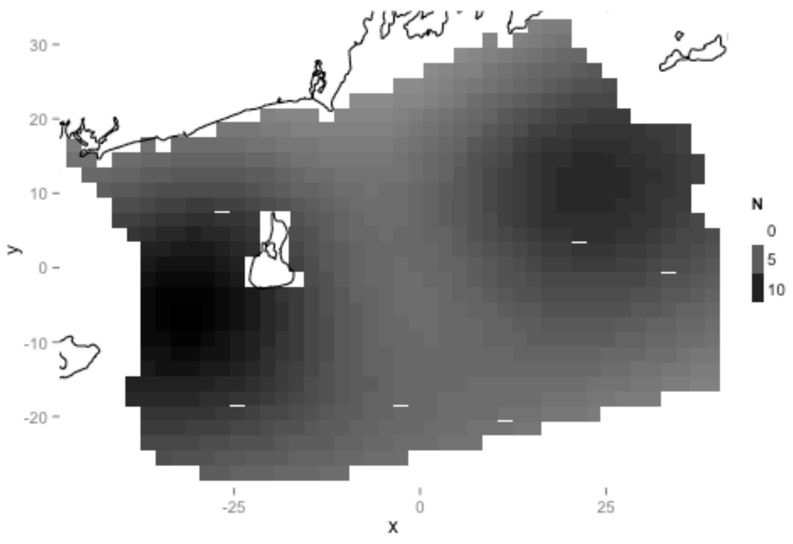


GAM check plot for the DSM of all data, using a detection function with observed group size and species, Tweedie response and bivaraite smooth of location

```
##
## Method: REML   Optimizer: outer newton
## full convergence after 6 iterations.
## Gradient range [-0.0001458,5.351e-07]
## (score 3467 & scale 10.89).
## Hessian positive definite, eigenvalue range [0.0001458,1150].
##
## 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(x,y) 14.000  9.521   0.824       1
```

```
plot.preds(alcid.all.covars.tw.xy)
```

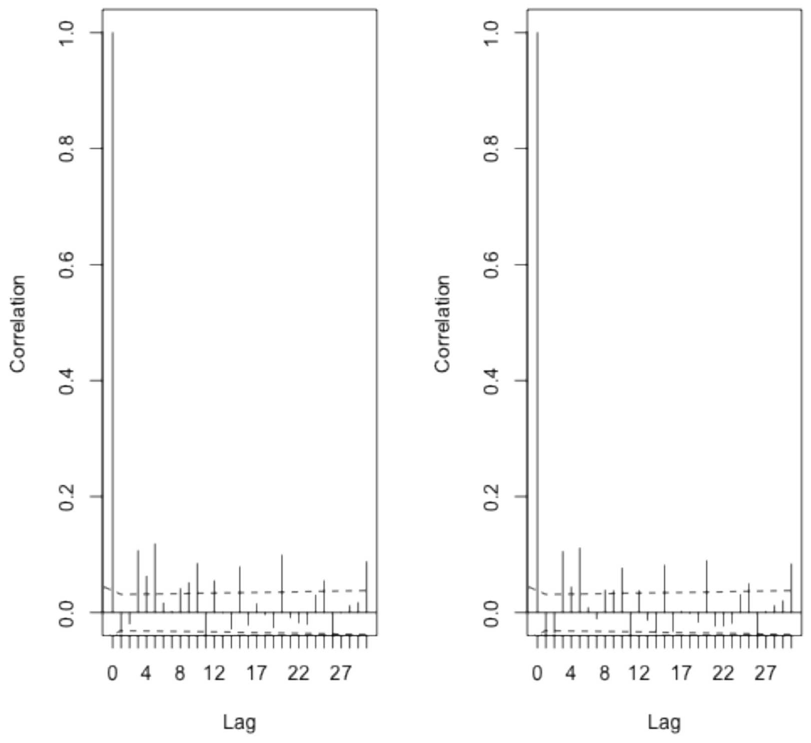
```
## Abundance = 5383
```



Predictions from the DSM using all species and a Tweedie response distribution and bivariate smooth of location.

We can also compare the residual autocorrelation between the models:

```
par(mfrow = c(1, 2))
dsm.cor(alcid.all.covars.tw, max.lag = 30)
dsm.cor(alcid.all.covars.tw.xy, max.lag = 30)
```



Autocorrelograms for the alcid DSM without (left) and with (right) a bivariate smooth of location.

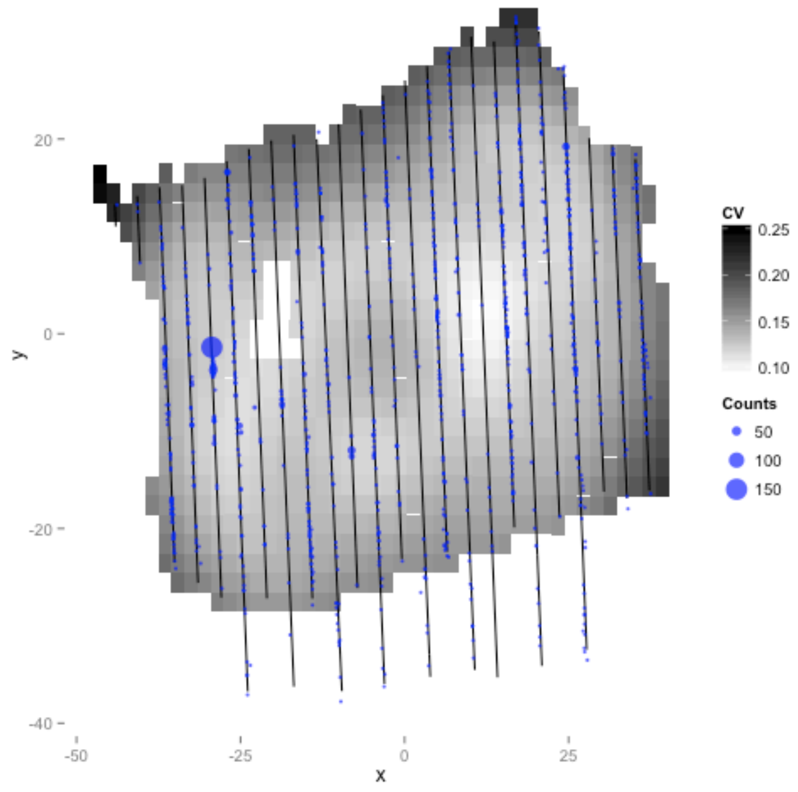
The bivariate smooth fo location has a much higher R^2, so we choose it over the model with separate terms for `x` and `y`.

Calculating the variance:

```
alcid.var <- dsm.var.gam(alcid.all.covars.tw.xy, pred, pred$cellarea)
summary(alcid.var)
```

```
## Summary of uncertainty in a density surface model calculated
##  analytically for GAM, with delta method
##
## Approximate asymptotic confidence interval:
##   5% Mean  95%
## 4803 5383 6033
## (Using delta method)
##
## Point estimate                 : 5383
## Standard error                 : 241
## CV of detection function       : 0.03722
## CV from GAM                    : 0.04477
## Total coefficient of variation : 0.0582
```

```
pred.grid <- split(pred, 1:nrow(pred))
alcid.var.grid <- dsm.var.gam(alcid.all.covars.tw.xy, pred.grid, pred$cellarea)
plot(alcid.var.grid)
```



Plot of uncertainty in the abundance distribution for alcids

End

Save the results!

```
save.image("alcids-models.RData")
```

Save the predictions/cv/variance for Zonation.

```
alcid.preds <- predict(alcid.all.covars.tw.xy, pred, pred$cellaream)



alcid.pred.data <- data.frame(cellid = 1:920, pred = alcid.preds, var = diag(alcid.var.grid$pred.var),
    cv = sqrt(diag(alcid.var.grid$pred.var))/alcid.preds, season = rep("Winter",
        920))
save(alcid.pred.data, file = "alcid-preds.RData")
```

References

Xie Y (2013) knitr: A general-purpose package for dynamic report generation in R. R package version 1.0. http://CRAN.R-project.org/package=knitr

Scoter analysis

This document is a record of modelling for the scoter data from the URI aerial line transect survey of the OSAMP area off the coast of Rhode Island. It is provided as a `knitr` (Xie 2013) file, that includes all the necessary code to re-create the analysis.

Preamble

```
suppressPackageStartupMessages(library(osampuri))
```

Loading up the data and trimming for what we want...

```
data("uri-lt-data")
obs.scoter <- obs[obs$Group == "Scoter", ]
rm(obs)
```

It's almost a 50/50 split between on water and flying:

```
table(obs.scoter$Location)/nrow(obs.scoter)
```

```
##
##   Flying On Water
##   0.4009   0.5991
```

Selecting those on water (but saving the flying birds)

```
obs.scoter.flying <- obs.scoter[obs.scoter$Location == "Flying", ]
obs.scoter <- obs.scoter[obs.scoter$Location == "On Water", ]
```

Now looking at the split per season, then per season-year:

```
table(obs.scoter$Season)
```

```
##
##   Fall Spring Summer Winter
##      3     27      0    103
```

```
table(obs.scoter$SeasonYear)
```

```
##
##   Fall2010   Fall2011 Spring2011 Spring2012 Summer2011 Summer2012
##          1          2         21          6          0          0
## Winter1011 Winter1112
##         50         53
```

looks like we need to combine the years for winter...

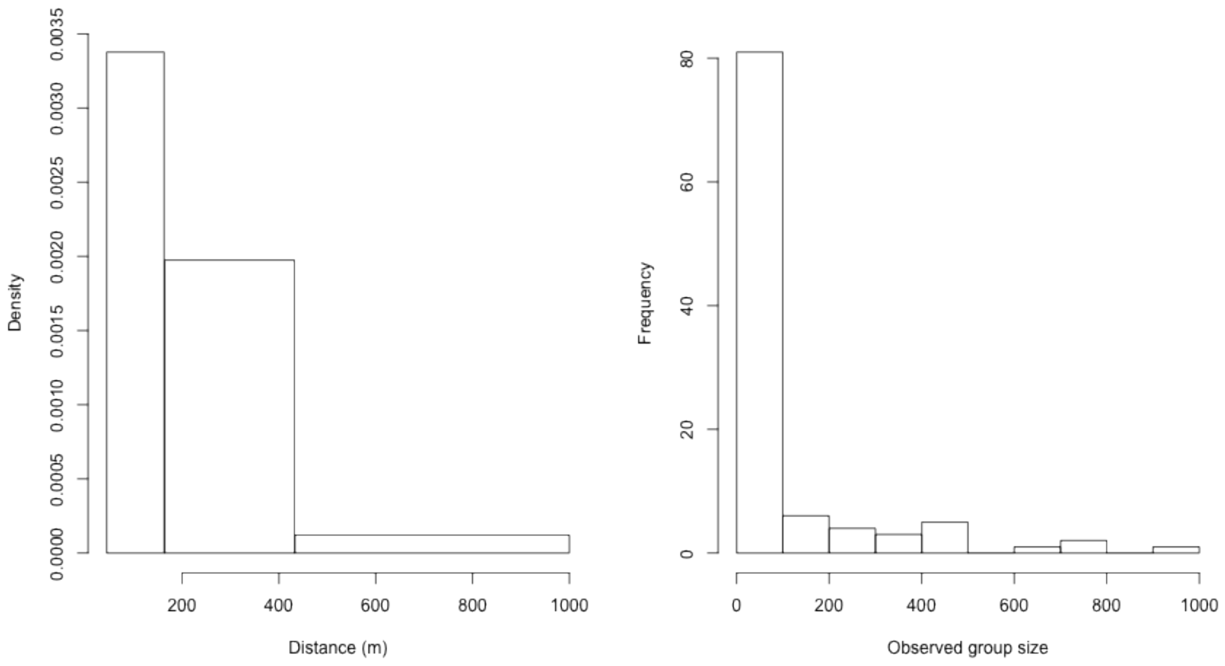
Pull out the Winter observations...

```
obs.scoter <- obs.scoter[obs.scoter$Season == "Winter", ]
effort <- effort[effort$Season == "Winter", ]
```

EDA plots

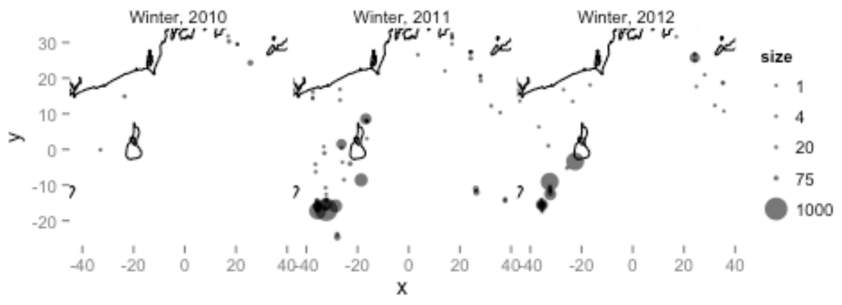
Before fitting a detection function, we first plot a histogram of distances and group sizes:

```
par(mfrow = c(1, 2))
hist(obs.scoter$distance, breaks = sort(unique(c(obs.scoter$distbegin, obs.scoter$distend))),
    main = "", xlab = "Distance (m)")
hist(obs.scoter$size, main = "", xlab = "Observed group size")
```



Histogram of distances (left) and observed group sizes (right).

```
plot.uri.data(obs.scoter, facet = eval(parse(text = "Season~Year")))
```



Plot of scoter observations per season and year, overlaid on the OSAMP area.

Detection function fitting

Fitting half-normal and hazard rate detection functions with (rescaled) size covariate.

```
hn.df <- ds(obs.scoter, adjustment = NULL, truncation = list(right = 1000, left = 44))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting half-normal key function
```

```
## AIC= 184.238
```

```
## No survey area information supplied, only estimating detection function.
```

```
hr.df <- ds(obs.scoter, adjustment = NULL, truncation = list(right = 1000, left = 44),
    key = "hr")
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function
```

```
## AIC= 185.722
## No survey area information supplied, only estimating detection function.
```

```
obs.scoter.scaled <- obs.scoter
obs.scoter.scaled$size <- obs.scoter.scaled$size/sd(obs.scoter.scaled$size)^2

hn.df.size <- ds(obs.scoter.scaled, formula = ~size, adjustment = NULL, truncation = list(right = 1000,
    left = 44))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting half-normal key function
```

```
## AIC= 184.595
```

```
## No survey area information supplied, only estimating detection function.
```

```
hr.df.size <- ds(obs.scoter.scaled, formula = ~size, adjustment = NULL, truncation = list(right = 1000,
    left = 44), key = "hr")
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

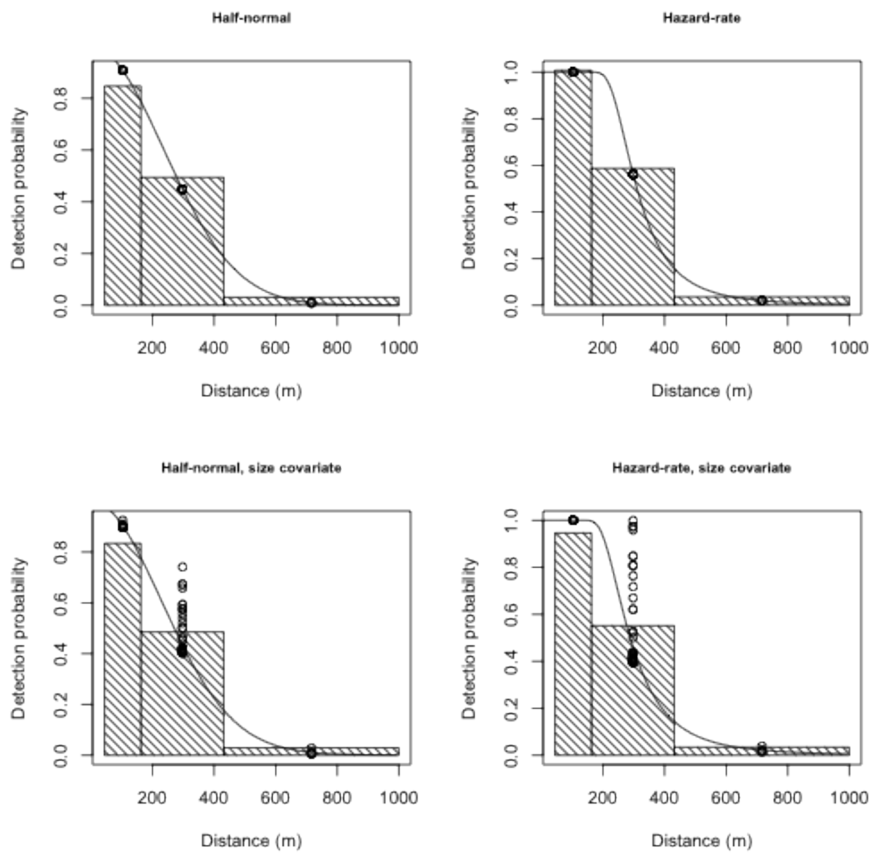
```
## Fitting hazard-rate key function
```

```
## AIC= 184.827
```

```
## No survey area information supplied, only estimating detection function.
```

	Detection function	Adjustments	Covariates	AIC	Δ AIC	# pars	p	CV(p)	C-vM p	KS p
1	hn			184.238	0	1	0.251	0.096	0	0
2	hn		size	184.595	0.357	2	0.247	0.095	0	0
3	hr		size	184.827	0.589	3	0.28	0.118	0	0
4	hr			185.722	1.484	2	0.298	0.121	0	0

```
par(mfrow = c(2, 2))
plot(hn.df, main = "Half-normal", xlab = "Distance (m)")
plot(hr.df, main = "Hazard-rate", xlab = "Distance (m)")
plot(hn.df.size, main = "Half-normal, size covariate", xlab = "Distance (m)")
plot(hr.df.size, main = "Hazard-rate, size covariate", xlab = "Distance (m)")
```



Plot of detection functions fitted to <>.

From the above plots and table, there is not much between the detection function is terms of AIC.

Density surface model

Allocating the effort

```
correct.effort <- allocate.effort(obs.scoter, seg)
obs.scoter <- correct.effort$obs
seg <- correct.effort$seg
```

Now fitting using the AIC-best detection function (half-normal, no covariates):

```
k1 <- 7
k2 <- 20
scoter.model <- dsm(Nhat~ s(roughness,k=k1)+
                          #s(gchl_winter,k=k1)+
                          #gchl_winter+
                          #s(gchl_fall,k=k1)+
                          #s(phimedian,k=k1)+
                          #s(distancelandkm,k=k1)+
                          s(depthm,k=k2)+
                          #s(x,y,k=k2),#+
                          s(x,k=k2)+
                          s(y,k=k1),
                hn.df, seg, obs.scoter,
                family=negbin(theta=c(0.013,0.015)),
                #family=negbin(theta=0.014),
                select=TRUE,method="REML")
```

```
## Warning: glm.fit: algorithm did not converge
```

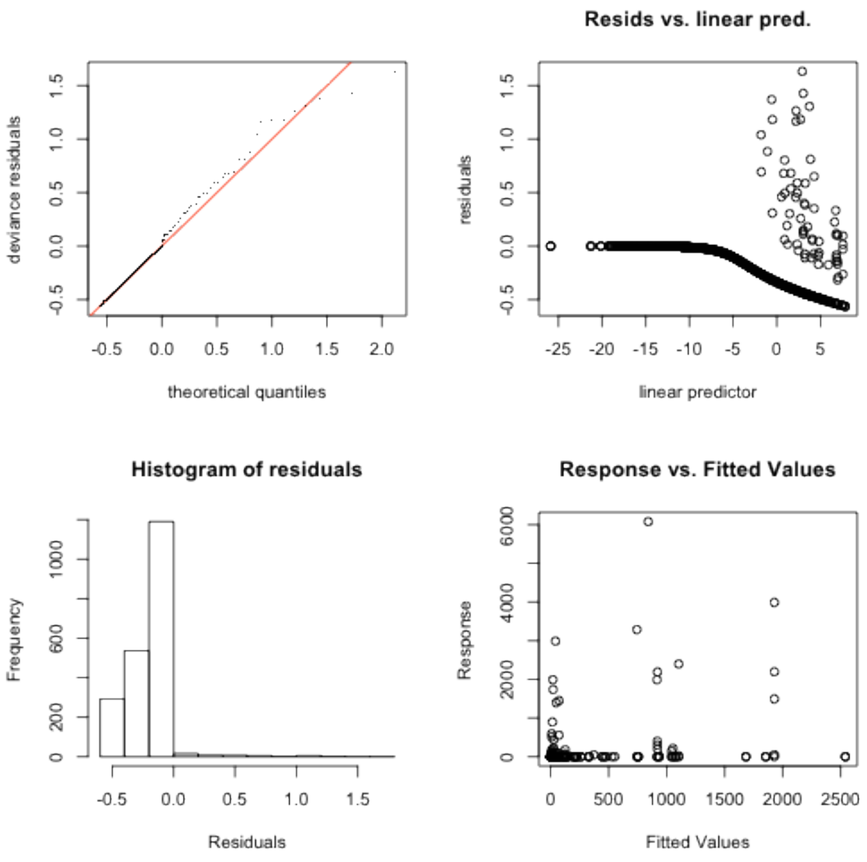
```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: fitting to calculate the null deviance did not converge --
## increase 'maxit'?
```

```
summary(scoter.model)
```

```
##
## Family: Negative Binomial(0.013)
## Link function: log
##
## Formula:
## Nhat ~ s(roughness, k = k1) + s(depthm, k = k2) + s(x, k = k2) +
##     s(y, k = k1) + offset(off.set)
## <environment: 0x10a2baee0>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -20.376      0.664   -30.7   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##               edf Ref.df Chi.sq p-value
## s(roughness) 2.02      6   14.6 0.00019 ***
## s(depthm)    3.90     19   40.6 1.3e-10 ***
## s(x)         7.48     19   83.1 < 2e-16 ***
## s(y)         4.80      6   51.0 5.3e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  -0.271   Deviance explained = 98.9%
## REML score = 703.13  Scale est. = 1         n = 2067
```

```
gam.check(scoter.model)
```



plot of chunk unnamed-chunk-11

```
##
## Method: REML   Optimizer: outer newton
## step failed after 1 iteration.
## Gradient range [-0.02968,-1.856e-05]
## (score 703.1 & scale 1).
## Hessian positive definite, eigenvalue range [1.856e-05,2.384].
##
## 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(roughness)  6.000  2.024   0.353       0
## s(depthm)    19.000  3.902   0.357       0
## s(x)         19.000  7.481   0.315       0
## s(y)          6.000  4.797   0.359       0
```

Trying a bivariate smooth of location.

```
scoter.model.xy <- dsm(Nhat~ s(roughness,k=k1)+
                             #s(gchl_winter,k=k1)+
                             #s(gchl_fall,k=k1)+
                             #s(phimedian,k=k1)+
                             s(distancelandkm,k=k1)+
                             s(depthm,k=k1)+
                             s(x,y,k=k2),
                hn.df, seg, obs.scoter,
                family=negbin(theta=c(0.013,0.015)),
                select=TRUE,method="REML")
```

```
## Warning: glm.fit: algorithm did not converge
```

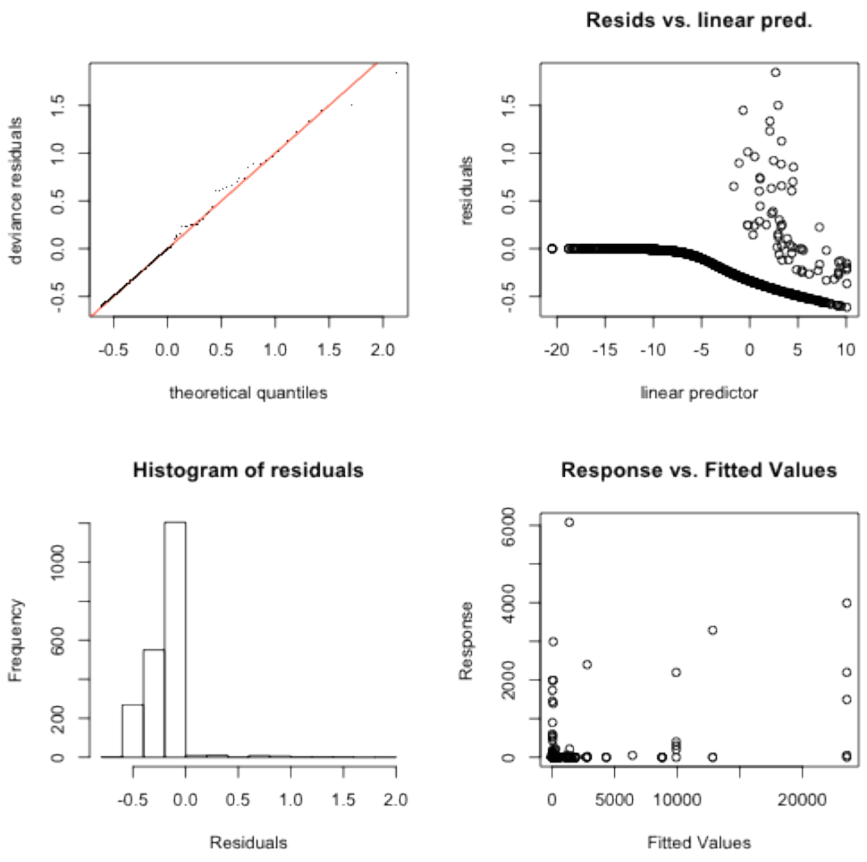
```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: fitting to calculate the null deviance did not converge --
## increase 'maxit'?
```

```
summary(scoter.model.xy)
```

```
##
## Family: Negative Binomial(0.013)
## Link function: log
##
## Formula:
## Nhat ~ s(roughness, k = k1) + s(distancelandkm, k = k1) + s(depthm,
##     k = k1) + s(x, y, k = k2) + offset(off.set)
## <environment: 0x11306a298>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -20.070      0.781   -25.7   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                     edf Ref.df Chi.sq p-value
## s(roughness)       2.04      6   18.2 2.0e-05 ***
## s(distancelandkm)  3.69      6   13.0 0.00045 ***
## s(depthm)          3.01      6   24.4 9.2e-07 ***
## s(x,y)            11.85     19  123.0 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  -35.5   Deviance explained = 98.9%
## REML score = 704.39  Scale est. = 1         n = 2067
```

```
gam.check(scoter.model.xy)
```



plot of chunk unnamed-chunk-12

```
##
## Method: REML   Optimizer: outer newton
## step failed after 3 iterations.
## Gradient range [-0.01034,2.336e-05]
## (score 704.4 & scale 1).
## eigenvalue range [-2.337e-05,3.163].
##
## 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(roughness)       6.000  2.043   0.341       0
## s(distancelandkm)  6.000  3.693   0.332       0
## s(depthm)          6.000  3.013   0.338       0
## s(x,y)            19.000 11.854   0.303       0
```

Model diagnostics indicate that something is going wrong here (negative R^2 with ~99% deviance explained). Let's switch to a presence/absence model and see if that gives more reasonable answers.

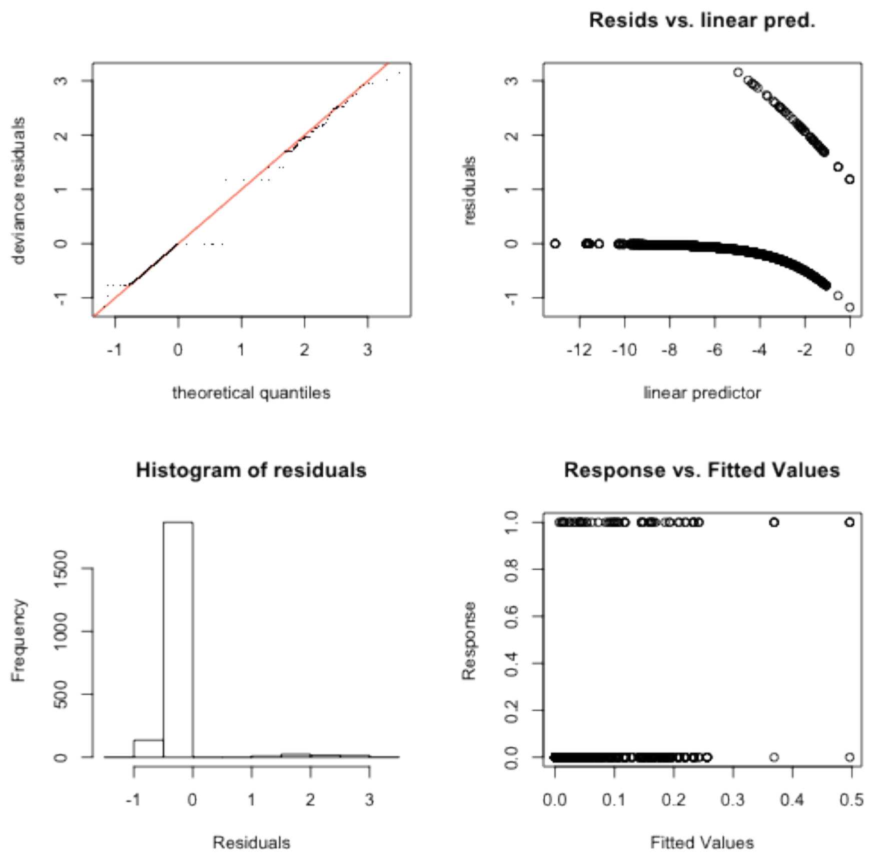
Spatial presence/absence model

Fitting a presence/absence model:

```
k1 <- 7
k2 <- 20
scoter.pres <- dsm(presence~ #s(roughness,k=k1)+
                             #s(gchl_winter,k=k1)+
                             #s(gchl_fall,k=k1)+
                             #s(phimedian,k=k1)+
                             #s(distancelandkm,k=k1)+
                             s(depthm,k=k1)+
                             s(x,k=k1)+
                             s(y,k=k1),
                NULL, seg, obs.scoter,
                family=binomial(),
                select=TRUE,method="REML")
summary(scoter.pres)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## presence ~ s(depthm, k = k1) + s(x, k = k1) + s(y, k = k1)
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -5.213      0.366   -14.2   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##            edf Ref.df Chi.sq p-value
## s(depthm) 2.98      6   35.4 4.9e-10 ***
## s(x)      4.08      6   35.0 2.4e-08 ***
## s(y)      3.10      6   26.9 5.1e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =   0.14   Deviance explained = 28.3%
## REML score = 232.04  Scale est. = 1         n = 2067
```

```
gam.check(scoter.pres)
```

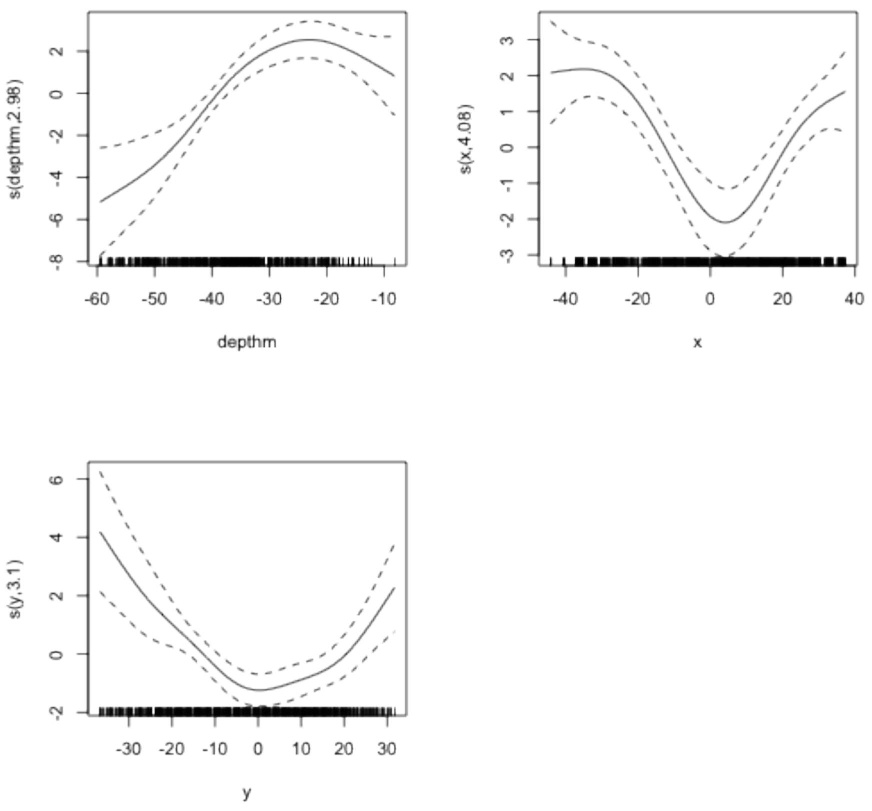


GAM check plot for the presence/absence model for scoter

```
##
## Method: REML   Optimizer: outer newton
## full convergence after 12 iterations.
## Gradient range [-4.515e-07,1.757e-05]
## (score 232 & scale 1).
## eigenvalue range [-1.757e-05,1.3].
##
## 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(depthm) 6.000 2.981   0.856       0
## s(x)      6.000 4.084   0.835       0
## s(y)      6.000 3.097   0.831       0
```

Looking at the smooth terms...

```
plot(scoter.pres, scale = 0, pages = 1)
```



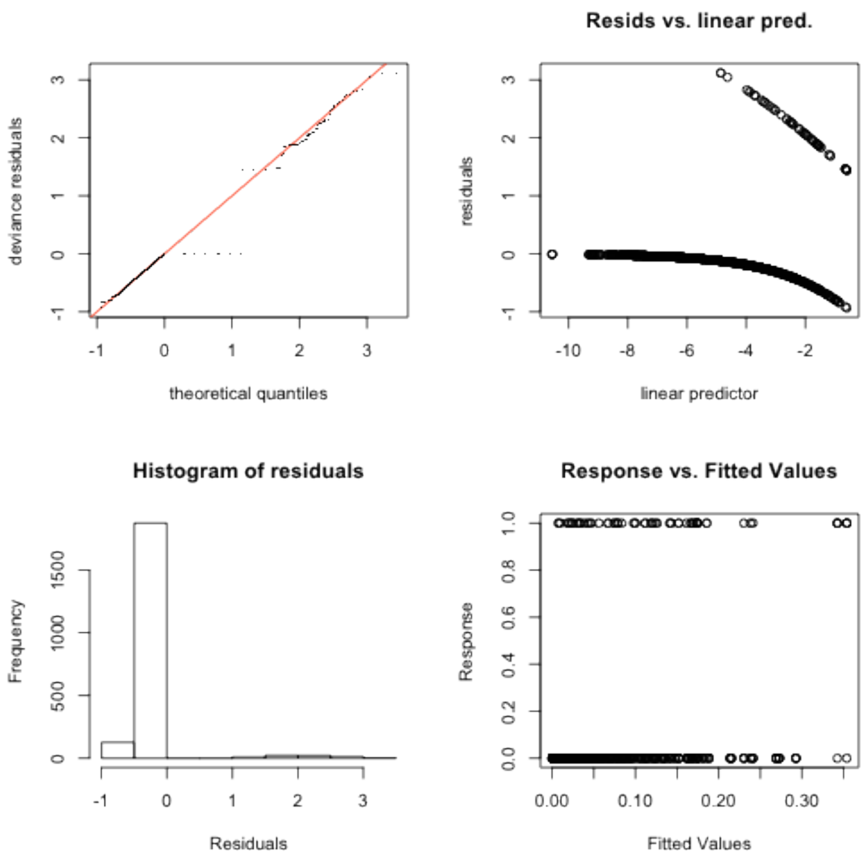
plot of chunk unnamed-chunk-15

What about a bivariate smooth, since we have both `x` and `y` in our model?

```
scoter.pres.xy <- dsm(presence~ #s(roughness,k=k1)+
#                  s(gchl_winter,k=k1)+
#                  s(gchl_fall,k=k1)+
#                  s(phimedian,k=k1)+
#                  s(distancelandkm,k=k1)+
                  s(depthm,k=k1)+
                  s(x,y,k=10),
                NULL, seg, obs.scoter,
                family=binomial(),
                select=TRUE,method="REML")
summary(scoter.pres.xy)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## presence ~ s(depthm, k = k1) + s(x, y, k = 10)
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -4.919      0.348   -14.2   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##            edf Ref.df Chi.sq p-value
## s(depthm) 2.49      6   23.8 7.9e-07 ***
## s(x,y)    6.10      9   49.5 3.0e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.114   Deviance explained = 25.8%
## REML score = 233.46  Scale est. = 1         n = 2067
```

```
gam.check(scoter.pres.xy)
```



GAM check plot for the presence/absence model (with bivariate smooth of location) for scoter

```
##
## Method: REML   Optimizer: outer newton
## full convergence after 11 iterations.
## Gradient range [-1.571e-06,1.563e-06]
## (score 233.5 & scale 1).
## Hessian positive definite, eigenvalue range [0.06396,2.214].
##
## 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(depthm) 6.000 2.494   0.833       0
## s(x,y)    9.000 6.097   0.796       0
```

Looking at the REML scores (which are named `gcv.ubre` in `mgcv` for legacy reasons)

```
scoter.pres$gcv.ubre
```

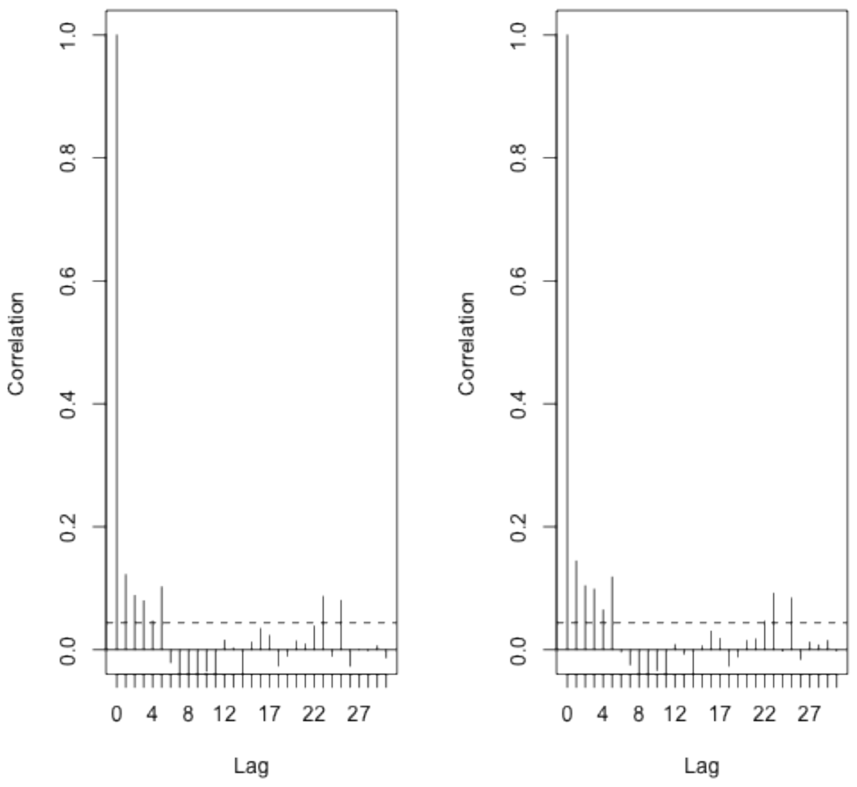
```
## [1] 232
## attr(,"Dp")
## [1] 214.4
```

```
scoter.pres.xy$gcv.ubre
```

```
## [1] 233.5
## attr(,"Dp")
## [1] 220.8
```

We can also compare the residual autocorrelation for the two presence/absence models:

```
par(mfrow = c(1, 2))
dsm.cor(scoter.pres, max.lag = 30)
dsm.cor(scoter.pres.xy, max.lag = 30)
```

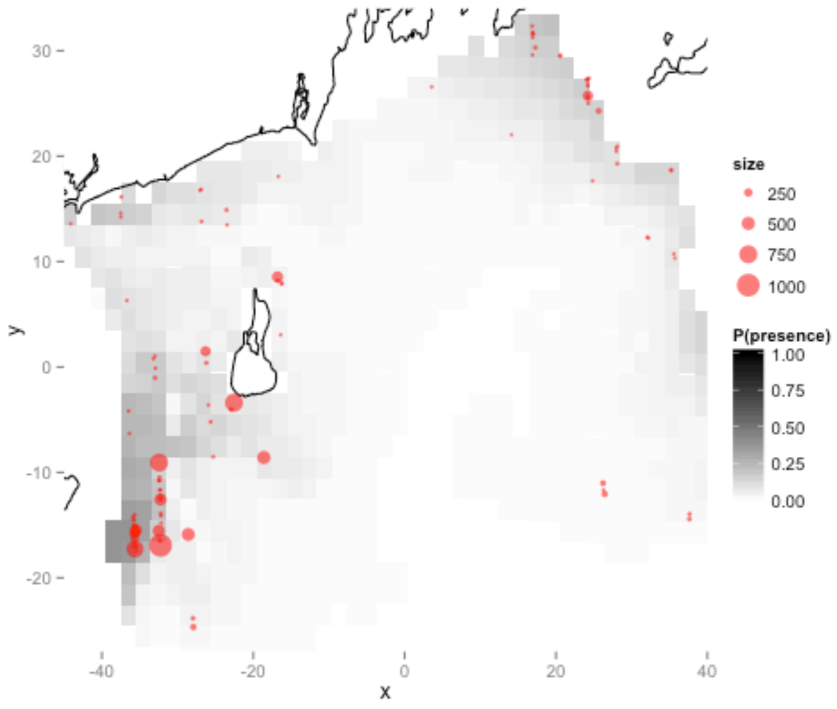


Autocorrelograms for the presence/absence models without (left) and with (right) bivariate smooths of location.

The bivariate smooth has a better score.

Finally, we can look at the predicted probability of presence as a surface

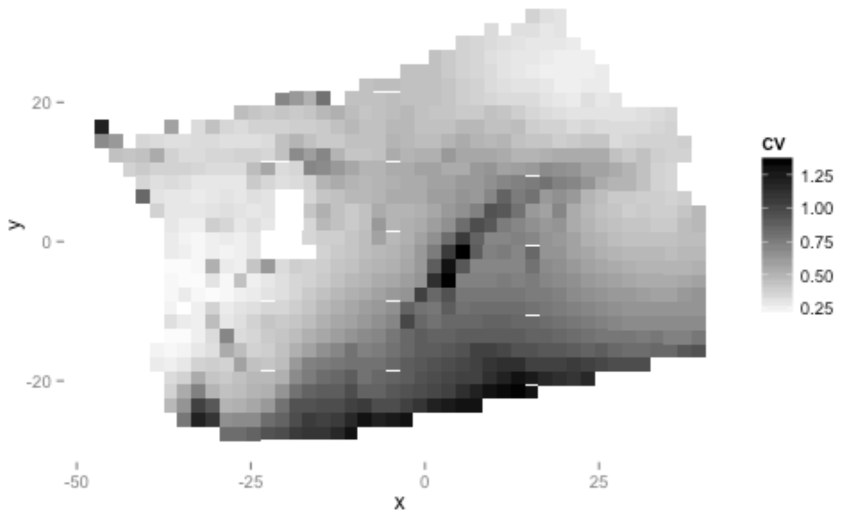
```
scoter.predict <- predict(scoter.pres.xy, pred, 1)
scoter.predict <- cbind(pred, p = scoter.predict)
p <- ggplot(scoter.predict, aes(x, y))
p <- p + p.opts.geo
p <- p + geom_tile(aes(fill = p, height = 2, width = 2))
p <- p + geom_polygon(aes(x = x, y = y, group = group), colour = "black", fill = NA,
    data = coast)
p <- p + coord_equal(xlim = xlims, ylim = ylims)
p <- p + scale_fill_gradient(low = "white", high = "black", limits = c(0, 1))
p <- p + geom_point(aes(x = x, y = y, size = size), colour = "red", data = obs.scoter,
    alpha = 0.6)
p <- p + labs(fill = "P(presence)")
print(p)
```



Plot of predicted probability of presence of scoter in the OSAMP area.

and the uncertainty in that surface:

```
pred$width <- pred$height <- 2
pred.grid <- split(pred, 1:nrow(pred))
scoter.var.grid <- dsm.var.gam(scoter.pres.xy, pred.grid, split(rep(1, nrow(pred)),
    1:nrow(pred)))
plot(scoter.var.grid, observations = FALSE)
```



Plot of uncertainty in predicted probability of presence of scoter in the OSAMP area.

Save everything

```
save.image("scoter-pres-models.RData")
```

```
scoter.preds <- scoter.predict$p



scoter.pred.data <- data.frame(cellid = 1:920, pred = scoter.preds, var = diag(scoter.var.grid$pred.var),
    cv = sqrt(diag(scoter.var.grid$pred.var))/scoter.preds, season = rep("Winter",
        920))
save(scoter.pred.data, file = "scoter-preds.RData")
```

References

Xie Y (2013) knitr: A general-purpose package for dynamic report generation in R. R package version 1.0. http://CRAN.R-project.org/package=knitr

Tern analysis

This document is a record of modelling for the tern data from the URI aerial line transect survey of the OSAMP area off the coast of Rhode Island. It is provided as a `knitr` (Xie 2013) file, that includes all the necessary code to re-create the analysis.

Preamble

Loading packages and selecting the correct data...

```
suppressPackageStartupMessages(library(osampuri))
data("uri-lt-data")
```

First, select the terns:

```
obs.tern <- obs[obs$Group == "Tern", ]
```

How many are on the water versus flying...

```
table(obs.tern$Location)
```

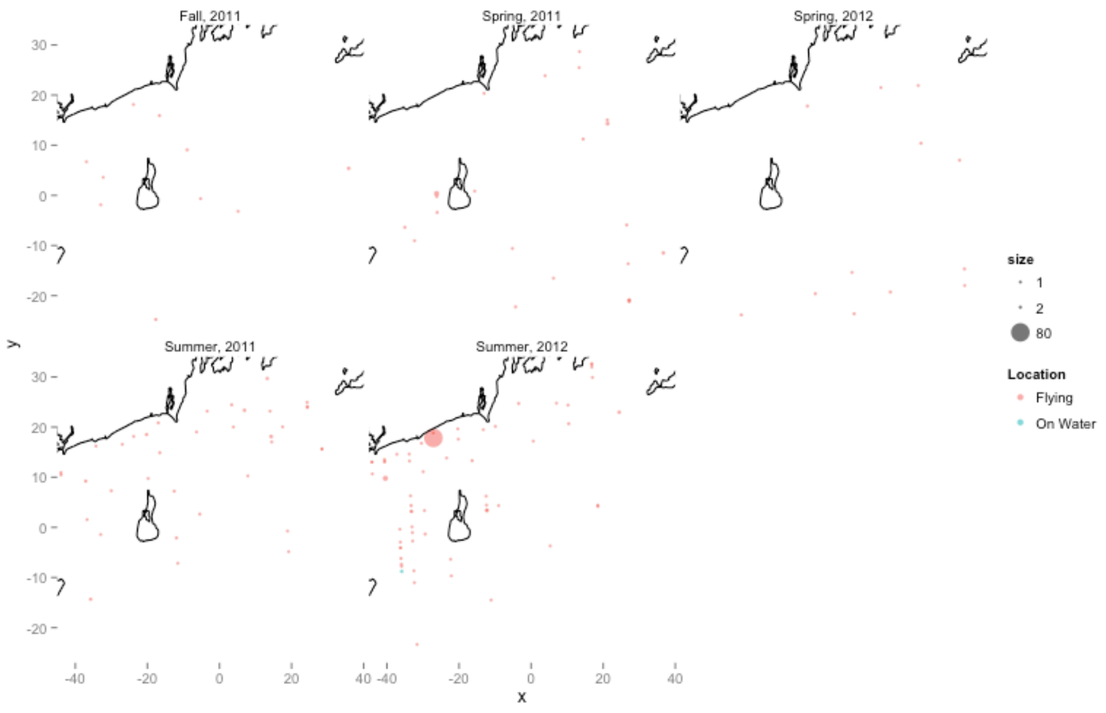
```
##
##   Flying On Water
##      148        1
```

we are going to assume that since the terns fly close to the surface of the water, we can include the flying birds in our analysis without violating the bins.

No observations had "Not recorded" bins.

What does the spatio-temporal distribution look like?

```
plot.uri.data(obs.tern, facet = eval(parse(text = "Season~Year")))
```



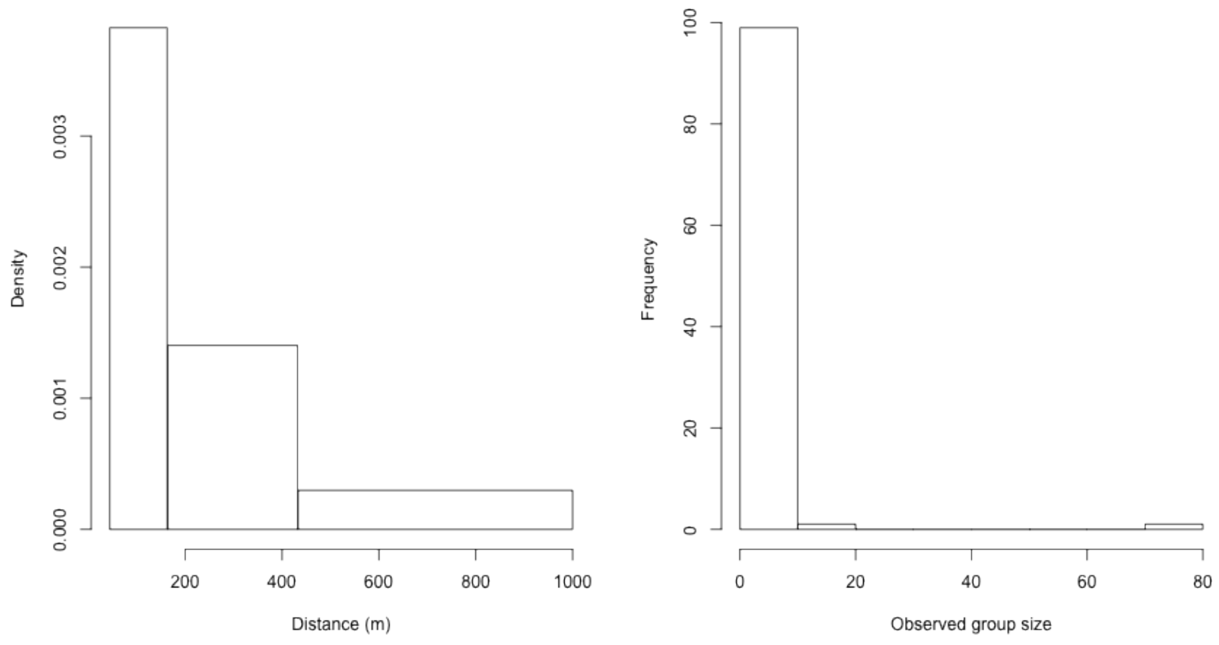
Tern observations, overlaid on a map of the OSAMP area

Looks like we want to model Summer...

```
obs.tern <- obs.tern[obs.tern$Season == "Summer", ]
effort <- effort[effort$Season == "Summer", ]
```

Before fitting a detection function, we first plot a histogram of distances and group sizes:

```
par(mfrow = c(1, 2))
hist(obs.tern$distance, breaks = sort(unique(c(obs.tern$distbegin, obs.tern$distend))),
    main = "", xlab = "Distance (m)")
hist(obs.tern$size, main = "", xlab = "Observed group size")
```



Histogram of distances (left) and observed group sizes (right).

Detection function

Let's fit some detection functions...

```
hn.df <- ds(obs.tern, adjustment = NULL, truncation = list(left = 44, right = 1000))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting half-normal key function
```

```
## AIC= 216.204
```

```
## No survey area information supplied, only estimating detection function.
```

```
hr.df <- ds(obs.tern, adjustment = NULL, truncation = list(left = 44, right = 1000),
    key = "hr")
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function
```

```
## AIC= 211.94
```

```
## No survey area information supplied, only estimating detection function.
```

```


obs.tern$ssize <- obs.tern$size/sd(obs.tern$size)


hn.df.size <- ds(obs.tern, adjustment = NULL, truncation = list(left = 44, right = 1000),
    key = "hr", formula = ~ssize)
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function
```

```
## AIC= 211.804
```

```
## No survey area information supplied, only estimating detection function.
```

```
hr.df.size <- ds(obs.tern, adjustment = NULL, truncation = list(left = 44, right = 1000),
    formula = ~ssize)
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

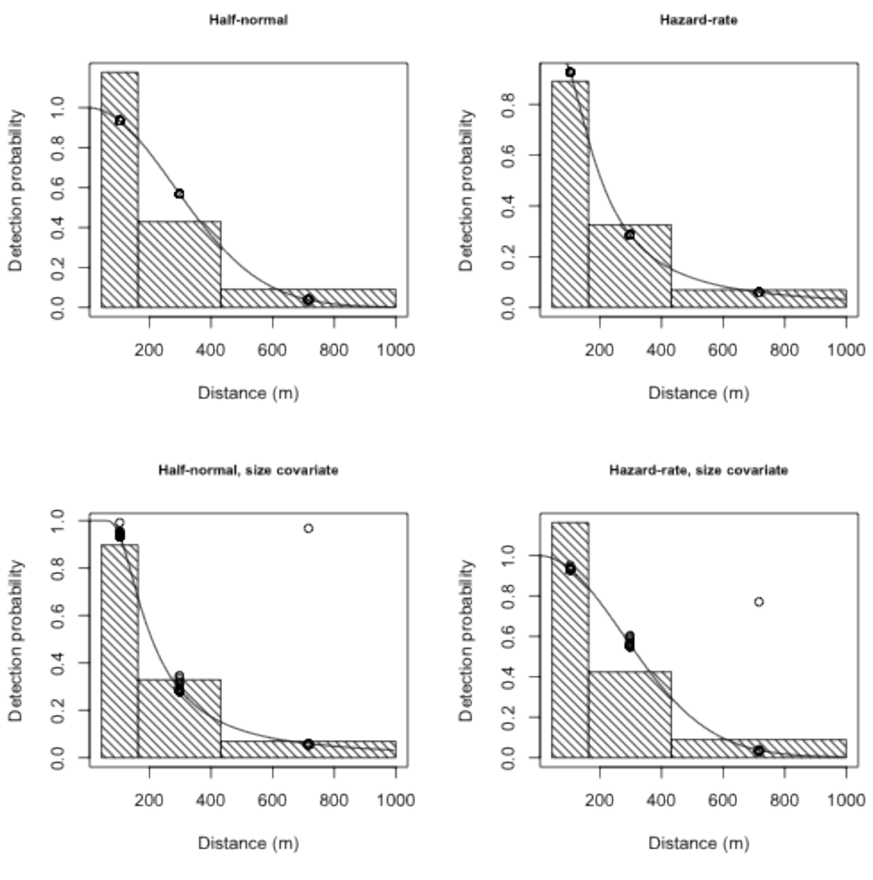
```
## Fitting half-normal key function
```

```
## AIC= 215.958
```

```
## No survey area information supplied, only estimating detection function.
```

	Detection function	Adjustments	Covariates	AIC	Δ AIC	# pars	p	CV(p)	C-vM p	KS p
1	hr		ssize	211.804	0	3	0.235	0.219	0	0
2	hr			211.94	0.136	2	0.233	0.232	0	0
3	hn		ssize	215.958	4.154	2	0.304	0.083	0	0
4	hn			216.204	4.4	1	0.307	0.083	0	0

```
par(mfrow = c(2, 2))
plot(hn.df, main = "Half-normal", xlab = "Distance (m)")
plot(hr.df, main = "Hazard-rate", xlab = "Distance (m)")
plot(hn.df.size, main = "Half-normal, size covariate", xlab = "Distance (m)")
plot(hr.df.size, main = "Hazard-rate, size covariate", xlab = "Distance (m)")
```



Plot of detection functions fitted to the tern data.

Spatial modelling

Allocate the effort correctly...

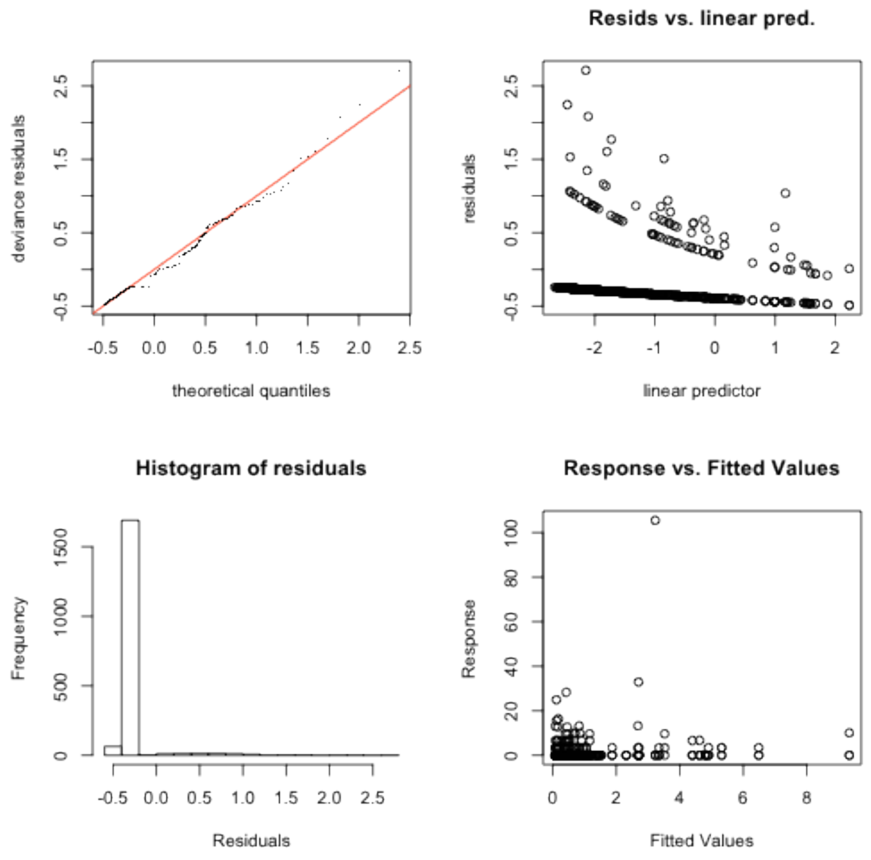
```
new.segobs <- allocate.effort(obs.tern, seg)
obs.tern <- new.segobs$obs
seg <- new.segobs$seg
```

First fitting a DSM without a bivariate smooth of location

```
k1 <- 7
k2 <- 20
tern.model <- dsm(Nhat~ #s(roughness,k=k1)+
                        #s(gchl_summer,k=k1)+
                        s(gchl_spring,k=k1),#+
                        #s(phimedian,k=k1)+
                        #s(distancelandkm,k=k1)+
                        #s(depthm,k=k1),#+
                        #s(x,y,k=k2)+
                        #s(x,k=k1)+
                        #s(y,k=k1),
                 hr.df.size, seg, obs.tern,
#                family=negbin(theta=0.06),
#                family=Tweedie(p=1.8),
                family=negbin(theta=c(0.01,0.03)),
        select=TRUE,method="REML")
summary(tern.model)
```

```
##
## Family: Negative Binomial(0.019)
## Link function: log
##
## Formula:
## Nhat ~ s(gchl_spring, k = k1) + offset(off.set)
## <environment: 0x108915608>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -16.953      0.179   -94.6   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                  edf Ref.df Chi.sq p-value
## s(gchl_spring) 1.66      6   28.3 3.7e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0203   Deviance explained = 15.8%
## REML score =  581.3  Scale est. = 1         n = 1836
```

```
gam.check(tern.model)
```



GAM check plots for the DSM without a bivariate smooth of location

```
##
## Method: REML   Optimizer: outer newton
## full convergence after 1 iteration.
## Gradient range [-0.0001207,0.0001952]
## (score 581.3 & scale 1).
## Hessian positive definite, eigenvalue range [0.1855,0.3762].
##
## 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(gchl_spring) 6.000 1.657   0.432     0.2
```

Re-fitting with a bivariate smooth of location:

```
tern.model.xy <- dsm(Nhat~ #s(roughness,k=k1)+
                        #s(gchl_summer,k=k1)+
                        #s(gchl_spring,k=k1)+
                        gchl_spring+
                        #s(phimedian,k=k1)+
                        #phimedian+
                        #s(distancelandkm,k=k1)+
                        #s(depthm,k=k1)+
                        s(x,y,k=k2),#+
                 hr.df.size, seg, obs.tern,
                family=negbin(theta=c(0.01,0.03)),
        select=TRUE,method="REML")
summary(tern.model.xy)
```

```
##
## Family: Negative Binomial(0.023)
## Link function: log
##
## Formula:
## Nhat ~ gchl_spring + s(x, y, k = k2) + offset(off.set)
## <environment: 0x1141628a0>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -19.656      0.625  -31.45  < 2e-16 ***
## gchl_spring    0.842      0.219    3.84  0.00012 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##         edf Ref.df Chi.sq p-value
## s(x,y) 8.89     19   27.7 0.00015 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  -1.78   Deviance explained = 27.8%
## REML score = 576.77  Scale est. = 1         n = 1836
```

Note that the model with `gchl_spring` gives very unreliable estimates of abundance.

```
sum(predict(tern.model.xy, pred, pred$cellaream))
```

```
## [1] 1133161
```

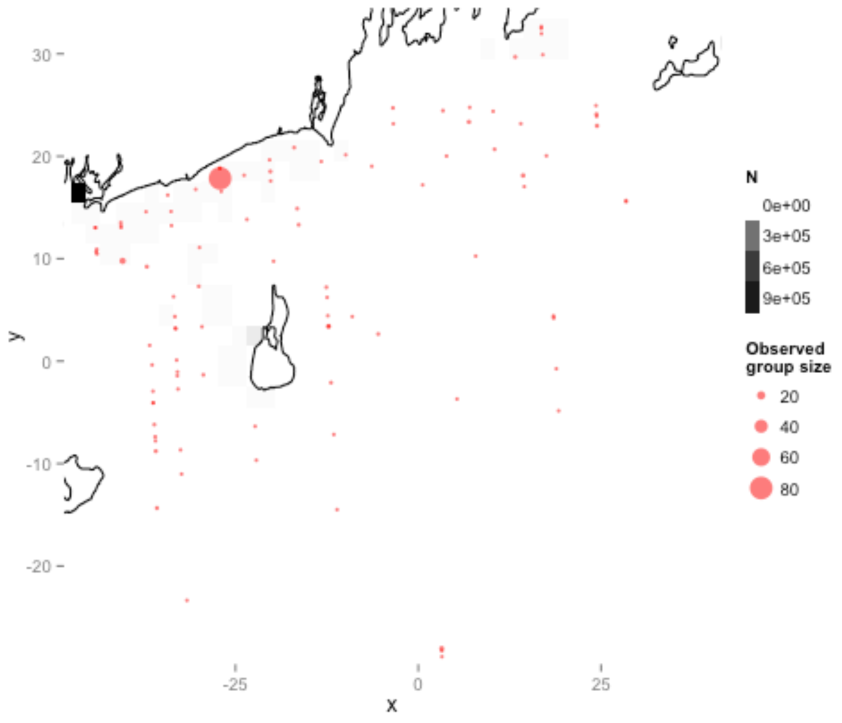
```
max(predict(tern.model.xy, pred, pred$cellaream))
```

```
## [1] 1127188
```

As we can see, the abundance estimate is being thrown out by the large observations in the top left corner of the SAMP area...

```
plot.preds(tern.model.xy, obs.tern)
```

```
## Abundance = 1133161
```



Plot of predicted abundance over the SAMP area.

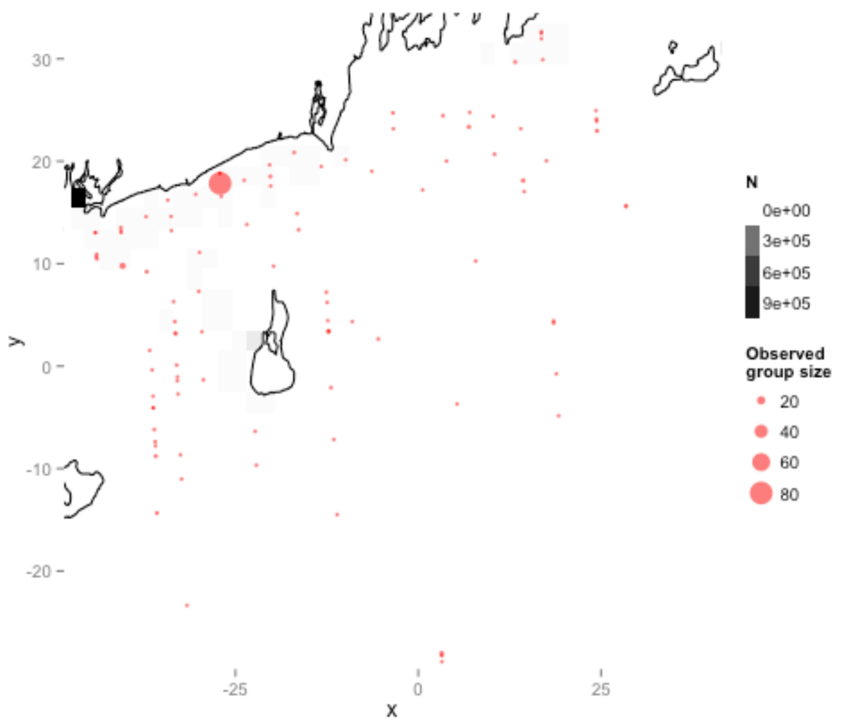
```
tern.model.xy.nospring <- dsm(Nhat~ #s(roughness,k=k1)+
                                    #s(gchl_summer,k=k1)+
                                    #s(phimedian,k=k1)+
                                    #s(distancelandkm,k=k1)+
                                    #s(depthm,k=k1)+
                                    s(x,y,k=k2),
                 hr.df.size, seg, obs.tern,
                family=negbin(theta=c(0.01,0.03)),
        select=TRUE,method="REML")
summary(tern.model.xy.nospring)
```

```
##
## Family: Negative Binomial(0.024)
## Link function: log
##
## Formula:
## Nhat ~ s(x, y, k = k2) + offset(off.set)
## <environment: 0x10b990378>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -17.527      0.191     -92   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##         edf Ref.df Chi.sq p-value
## s(x,y) 11.6     19   73.8 9.5e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0254   Deviance explained =   30%
## REML score = 578.59  Scale est. = 1         n = 1836
```

Let's see what the predicted surface looks like:

```
plot.preds(tern.model.xy, obs.tern)
```

```
## Abundance = 1133161
```



Plot of predicted abundance over the SAMP area.

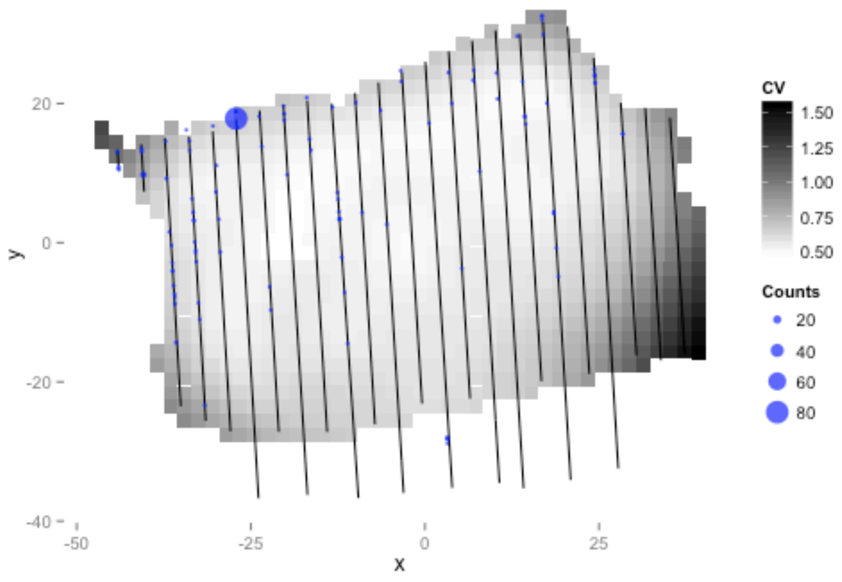
This abundance estimate seems much more reasonable, the associated uncertainty can also be calculated:

```
pred$height <- pred$width <- 2
tern.model.xy.var <- dsm.var.gam(tern.model.xy.nospring, split(pred, 1:nrow(pred)),
    pred$cellaream)
summary(tern.model.xy.var)
```

```
## Summary of uncertainty in a density surface model calculated
##  analytically for GAM, with delta method
##
## Approximate asymptotic confidence interval:
##    5%  Mean   95%
## 159.4 286.0 513.2
## (Using delta method)
##
##
## Point estimate                 : 286
## Standard error                 : 83.91
## CV of detection function       : 0.08349
## CV from GAM                    : 0.2934
## Total coefficient of variation : 0.305
```

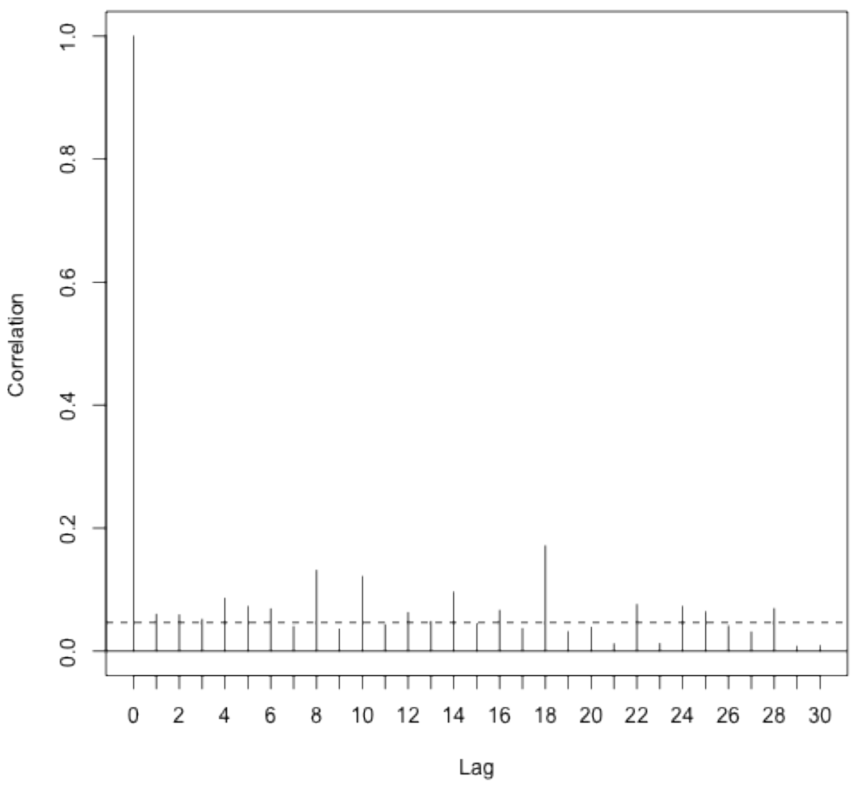
and then plotted:

```
plot(tern.model.xy.var)
```



Uncertainty in predictions for terns.

```
dsm.cor(tern.model.xy.nospring, max.lag = 30)
```



Autocorrelogram for the residuals from the tern model.

The autocorrelogram shows minimal residual autocorrelation.

Finish up

Save the analyses

```
save.image("tern-models.RData")
```

Save predictions for Zonation:

```
tern.preds <- predict(tern.model.xy.nospring, pred, pred$cellaream)

pred.grid <- split(pred, 1:nrow(pred))
tern.var.grid <- dsm.var.gam(tern.model.xy.nospring, pred.grid, pred$cellarea)

tern.pred.data <- data.frame(cellid = 1:920, pred = tern.preds, var = diag(tern.var.grid$pred.var),
    cv = sqrt(diag(tern.var.grid$pred.var))/tern.preds, season = rep("Summer",
        920))
save(tern.pred.data, file = "tern-preds.RData")
```

References

Xie Y (2013) knitr: A general-purpose package for dynamic report generation in R. R package version 1.0. http://CRAN.R-project.org/package=knitr

Gannet analysis

This document is a record of modelling for the gannet data from the URI aerial line transect survey of the OSAMP area off the coast of Rhode Island. It is provided as a `knitr` (Xie 2013) file, that includes all the necessary code to re-create the analysis.

Preamble

First, loading the data and selecting the correct parts of it.

```
suppressPackageStartupMessages(library(osampuri))
data("uri-lt-data")
obs.gannet <- obs[obs$Group == "Gannet", ]
rm(obs)
```

Remove observations without bins:

```
obs.gannet <- obs.gannet[obs.gannet$Bin != "Not recorded", ]
```

Looking at the numbers on water versus on the wing:

```
table(obs.gannet$Location)
```

```
##
##   Flying On Water
##      894      547
```

We can only really model those birds on water...

```
obs.gannet <- obs.gannet[obs.gannet$Location == "On Water", ]
```

Sightings per season, year and season-year:

```
table(obs.gannet$Season)
```

```
##
##   Fall Spring Summer Winter
##     82    192      1    272
```

```
table(obs.gannet$Year)
```

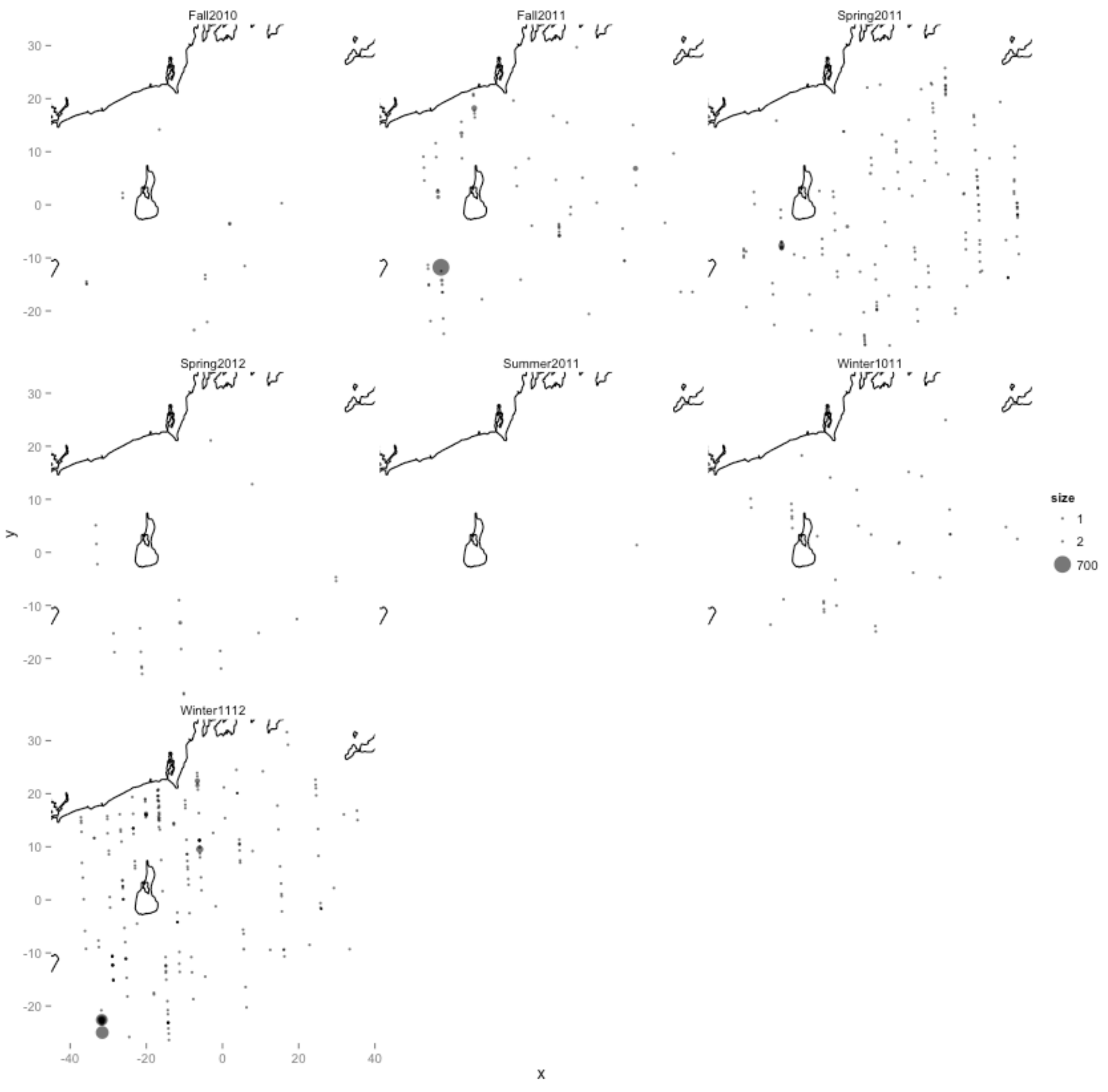
```
##
## 2010 2011 2012
##   32  340  175
```

```
table(obs.gannet$SeasonYear)
```

```
##
##   Fall2010   Fall2011 Spring2011 Spring2012 Summer2011 Summer2012
##         15         67        162         30          1          0
## Winter1011 Winter1112
##         35        237
```

Looking at the raw observatiosn, Spring 2011 and Winter 2011-2012 look most promising.

```
plot.uri.data(obs.gannet, eval(parse(text = "~SeasonYear")))
```



Plot of raw observations of gannets, per season-year combination over the OSAMP area.

Winter only

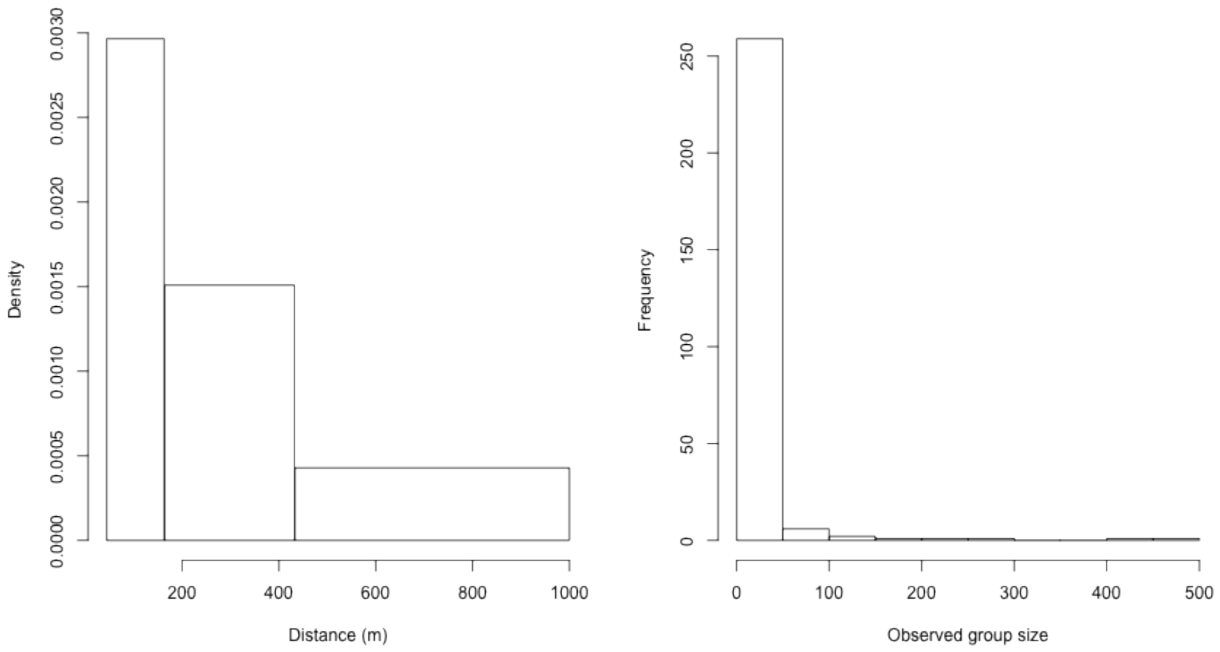
Let's start with a Winter only model, and see what happens...

```
obs.gannet.winter <- obs.gannet[obs.gannet$Season == "Winter", ]
effort <- effort[effort$Season == "Winter", ]
```

EDA plots

Before fitting a detection function, we first plot a histogram of distances and group sizes:

```r
par(mfrow = c(1, 2))
hist(obs.gannet.winter$distance, breaks = sort(unique(c(obs.gannet.winter$distbegin,
    obs.gannet.winter$distend))), main = "", xlab = "Distance (m)")
hist(obs.gannet.winter$size, main = "", xlab = "Observed group size")
```



Histogram of distances (left) and observed group sizes (right).

Detection function fitting

Fitting half-normal and hazard-rate detection functions, with and without size as a covariate.

As usual we must re-scale the size covariate before fitting to avoid numerical issues.

```r
obs.gannet.winter$ssize <- obs.gannet.winter$size/sd(obs.gannet.winter$size)
```

```r
hn.df <- ds(obs.gannet.winter, adjustment = NULL, truncation = list(right = 1000,
    left = 44))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting half-normal key function
```

```
## AIC= 596.215
```

```
## No survey area information supplied, only estimating detection function.
```

```
hr.df <- ds(obs.gannet.winter, adjustment = NULL, truncation = list(right = 1000,
    left = 44), key = "hr")
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function
```

```
## AIC= 591.901
```

```
## No survey area information supplied, only estimating detection function.
```

```
hn.df.size <- ds(obs.gannet.winter, formula = ~ssize, adjustment = NULL, truncation = list(right = 1000,
    left = 44))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting half-normal key function
```

```
## AIC= 598.126
```

```
## No survey area information supplied, only estimating detection function.
```

```
hr.df.size <- ds(obs.gannet.winter, formula = ~ssize, adjustment = NULL, truncation = list(right = 1000,
    left = 44), key = "hr")
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function
```

```
## AIC= 593.59
```

```
## No survey area information supplied, only estimating detection function.
```

	Detection function	Adjustments	Covariates	AIC	Δ AIC	# pars	p	CV(p)	C-vM p	KS p
1	hr			591.901	0	2	0.321	0.128	0	0
2	hr		ssize	593.59	1.689	3	0.321	0.129	0	0
3	hn			596.215	4.314	1	0.384	0.056	0	0
4	hn		ssize	598.126	6.225	2	0.384	0.056	0	0

```
par(mfrow = c(2, 2))
plot(hn.df, main = "Half-normal", xlab = "Distance (m)")
```

```
## Error: formal argument "xlab" matched by multiple actual arguments
```

```
plot(hr.df, main = "Hazard-rate", xlab = "Distance (m)")
```

```
## Error: formal argument "xlab" matched by multiple actual arguments
```

```
plot(hn.df.size, main = "Half-normal, size covariate", xlab = "Distance (m)")
```

```
## Error: formal argument "xlab" matched by multiple actual arguments
```

```
plot(hr.df.size, main = "Hazard-rate, size covariate", xlab = "Distance (m)")
```

```
## Error: formal argument "xlab" matched by multiple actual arguments
```

Spatial model

Allocate the effort

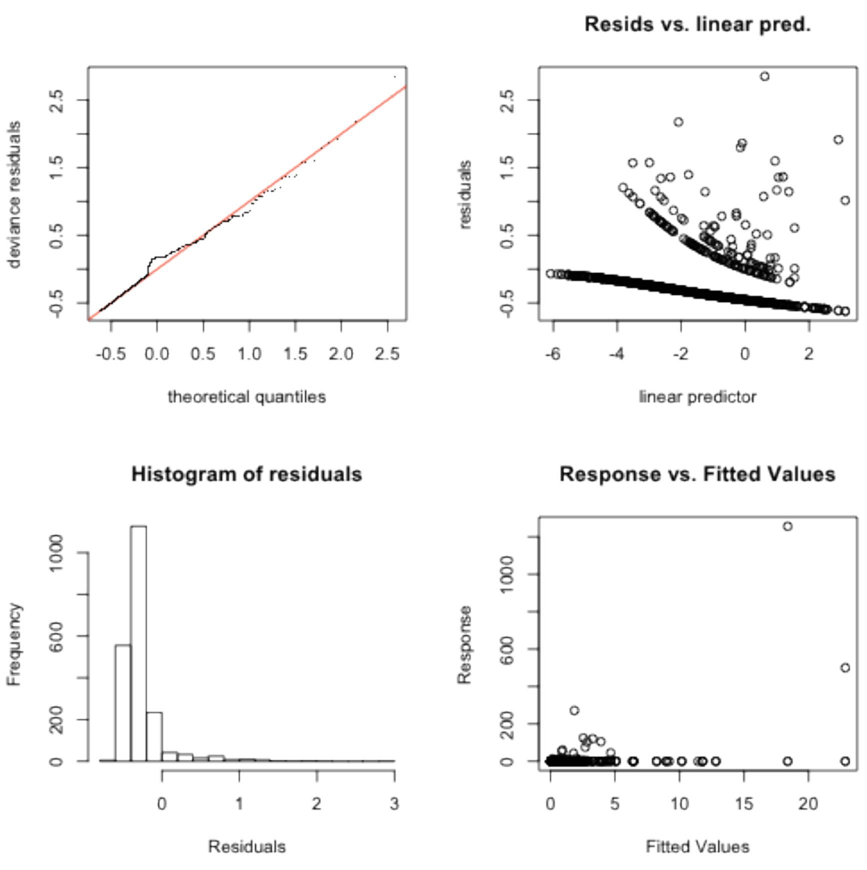
```
seg.save <- seg
correct.effort <- allocate.effort(obs.gannet.winter, seg)
obs.gannet.winter <- correct.effort$obs
seg <- correct.effort$seg
```

Fitting a DSM...

```
k1 <- 7
k2 <- 20
gannet.model.winter <- dsm(N~ #s(roughness,k=k1)+
                              #s(gchl_winter,k=k1)+
                              #s(gchl_fall,k=k1)+
                              #s(phimedian,k=k1)+
                              #phimedian+
                              #s(distancelandkm,k=k1)+
                              s(depthm,k=12)+
                              #s(x,k=k1)+
                              #s(y,k=k1)+
                              #y+
                              s(x,y,k=k2),
                hr.df, seg, obs.gannet.winter,
#                family=negbin(theta=c(0.01,0.03)),
                family=negbin(theta=0.029),
                method="REML",select=TRUE)
summary(gannet.model.winter)
```

```
##
## Family: Negative Binomial(0.029)
## Link function: log
##
## Formula:
## N ~ s(depthm, k = 12) + s(x, y, k = k2) + offset(off.set)
## <environment: 0x10b8d16a8>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -15.771      0.154    -103   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##             edf Ref.df Chi.sq p-value
## s(depthm)  5.12     11   28.6 6.9e-07 ***
## s(x,y)    10.70     19   83.5 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0256   Deviance explained =   42%
## REML score = 933.17  Scale est. = 1         n = 2067
```

```
gam.check(gannet.model.winter)
```



GAM check plots for 2 winter model.

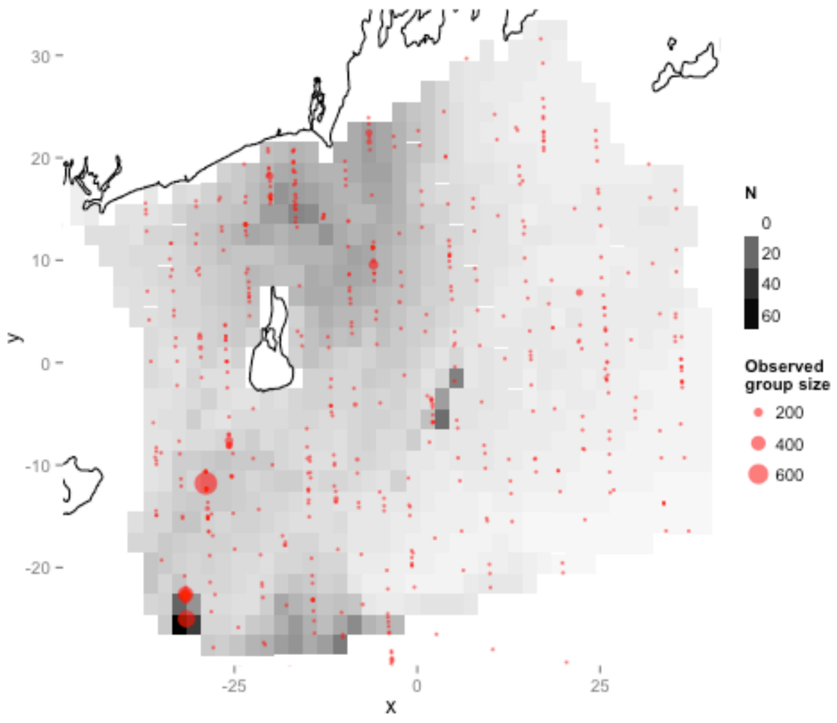
```
##
## Method: REML   Optimizer: outer newton
## step failed after 11 iterations.
## Gradient range [-0.0007628,0.009668]
## (score 933.2 & scale 1).
## Hessian positive definite, eigenvalue range [0.2959,2.915].
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                k'    edf k-index p-value
## s(depthm) 11.000  5.122   0.494    0.01
## s(x,y)    19.000 10.701   0.495    0.02
```

Check plots look good but the predictive power is not so great.

Looking at the point estimate for the abundance and the uncertainty...

```
plot.preds(gannet.model.winter, obs = obs.gannet)
```

```
## Abundance = 1360
```



Predicted abundance over the OSAMP area for 2 winter model.

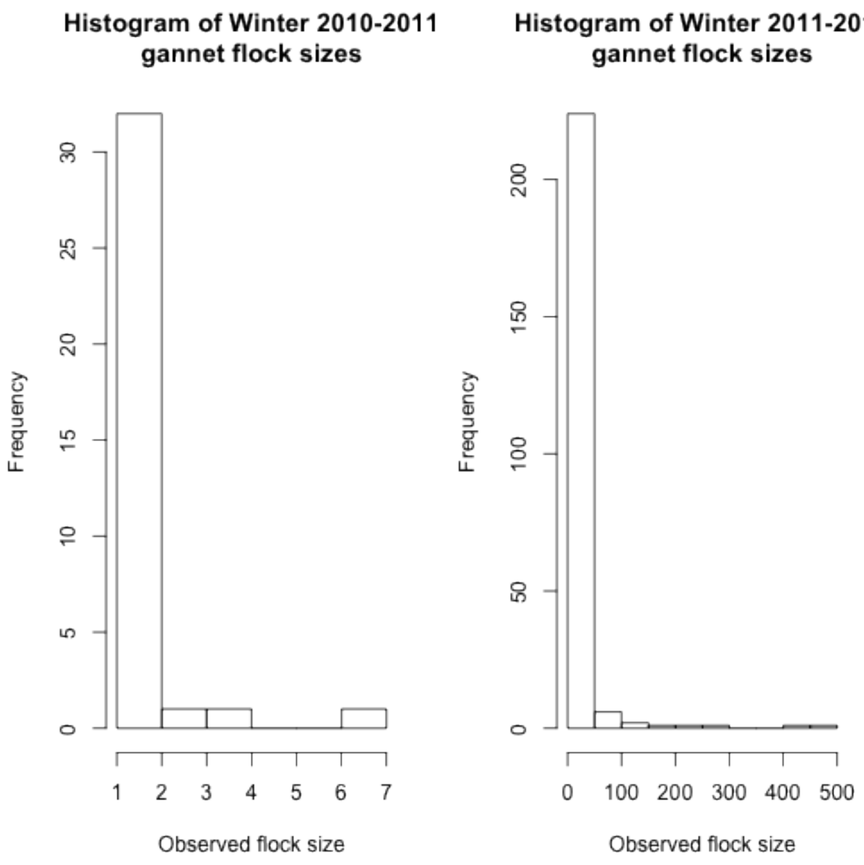
```
# gannet.var.winter <- dsm.var.gam(gannet.model.winter,pred,pred$cellarea)
# summary(gannet.var.winter)
```

Note here that there are only 35 observations in the Winter 2010-2011. This might be unduly effecting the model.

```
table(obs.gannet$SeasonYear)
```

```
##
##   Fall2010   Fall2011 Spring2011 Spring2012 Summer2011 Summer2012
##         15         67        162         30          1          0
## Winter1011 Winter1112
##         35        237
```

```
par(mfrow = c(1, 2))
hist(obs.gannet.winter$size[obs.gannet.winter$SeasonYear == "Winter1011"], main = "Histogram of Winter 20
    xlab = "Observed flock size")
hist(obs.gannet.winter$size[obs.gannet.winter$SeasonYear == "Winter1112"], main = "Histogram of Winter 20
    xlab = "Observed flock size")
```



Flock sizes per winter.

Restricting to winter 2011-2012

Given the large differences between the two seasons, it's probably better to model the winter of 2011-2012 on its own...

```
obs.gannet.1w <- obs.gannet[obs.gannet$SeasonYear == "Winter1112", ]
correct.effort <- allocate.effort(obs.gannet.1w, seg.save, effort[effort$SeasonYear ==
    "Winter1112", ])
obs.gannet.1w <- correct.effort$obs
seg.1w <- correct.effort$seg
```

```
hr.df.1w <- ds(obs.gannet.1w,adjustment=NULL,truncation=list(right=1000, left=44),key="hr")
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
```

```
## Fitting hazard-rate key function
```

```
## AIC= 520.945
```

```
## No survey area information supplied, only estimating detection function.
```

```
gannet.model.1w <- dsm(N~ #s(roughness,k=k1)+
                          #s(gchl_winter,k=k1)+
                          #s(gchl_fall,k=k1)+
                          #s(phimedian,k=k1)+
                          #s(distancelandkm,k=k1)+
                          s(depthm,k=12)+
                          #s(x,k=k1)+
                          #s(y,k=k1)+
                          #y,#+
                          s(x,y,k=k2),
                hr.df.1w, seg.1w, obs.gannet.1w,
#                family=negbin(theta=c(0.01,0.07)),
                family=negbin(theta=c(0.058,0.07)),
#                family=negbin(theta=0.057),
                method="REML",select=TRUE)
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: algorithm did not converge
```

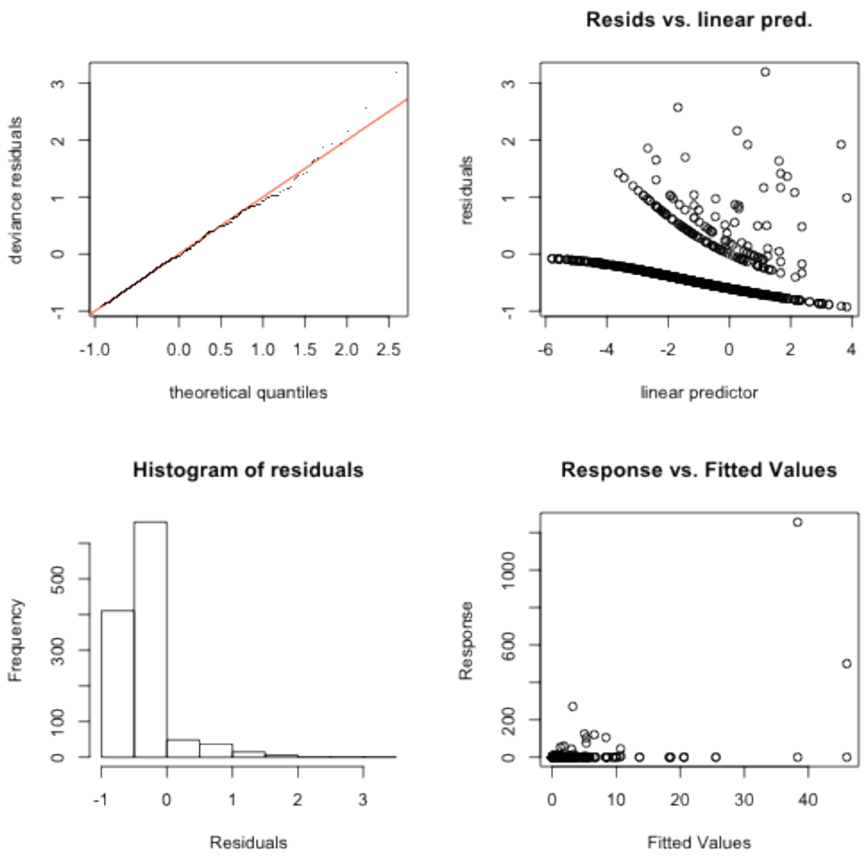
```
## Warning: fitting to calculate the null deviance did not converge --
## increase maxit?
```

```
summary(gannet.model.1w)
```

```
##
## Family: Negative Binomial(0.065)
## Link function: log
##
## Formula:
## N ~ s(depthm, k = 12) + s(x, y, k = k2) + offset(off.set)
## <environment: 0x10d7ef720>
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -15.486      0.147    -105   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##              edf Ref.df Chi.sq p-value
## s(depthm)   5.9     11   42.2 8.5e-10 ***
## s(x,y)     12.2     19  100.2 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0532   Deviance explained = 48.3%
## REML score = 736.62  Scale est. = 1         n = 1179
```

This has improved the predictive power and we have an equally good looking Q-Q plot.

```
gam.check(gannet.model.1w)
```

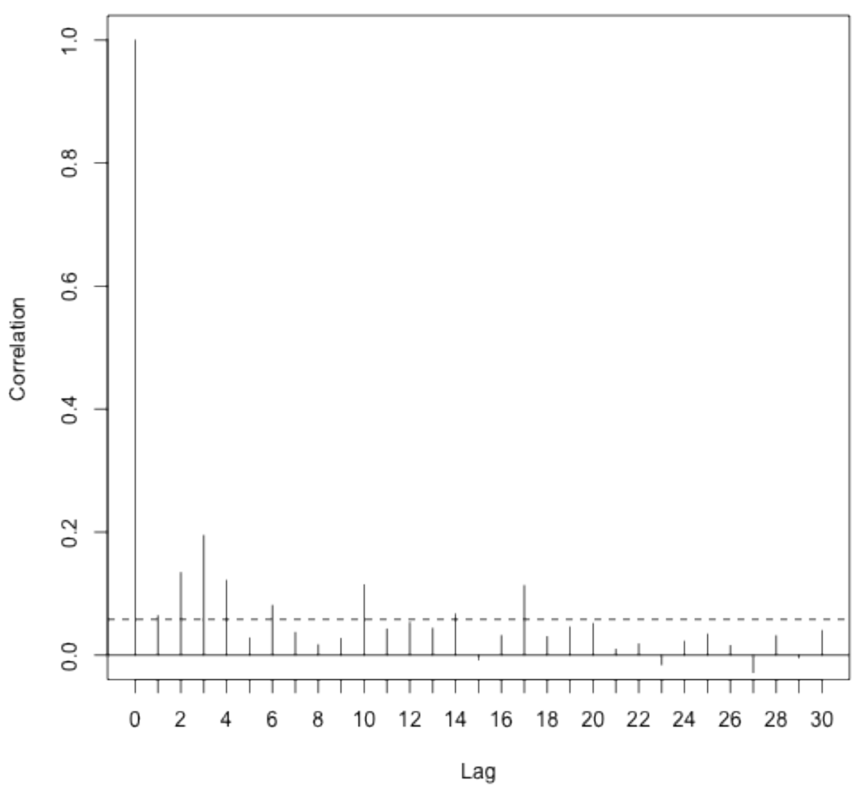


GAM check plot for 1 winter model.

```
##
## Method: REML   Optimizer: outer newton
## step failed after 1 iteration.
## Gradient range [-0.00545,0.005806]
## (score 736.6 & scale 1).
## Hessian positive definite, eigenvalue range [0.3155,3.208].
##
## 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(depthm) 11.000  5.896   0.579    0.12
## s(x,y)    19.000 12.163   0.518    0.00
```

Checking for residual autocorrelation in the model:

```
dsm.cor(gannet.model.1w, max.lag = 30)
```



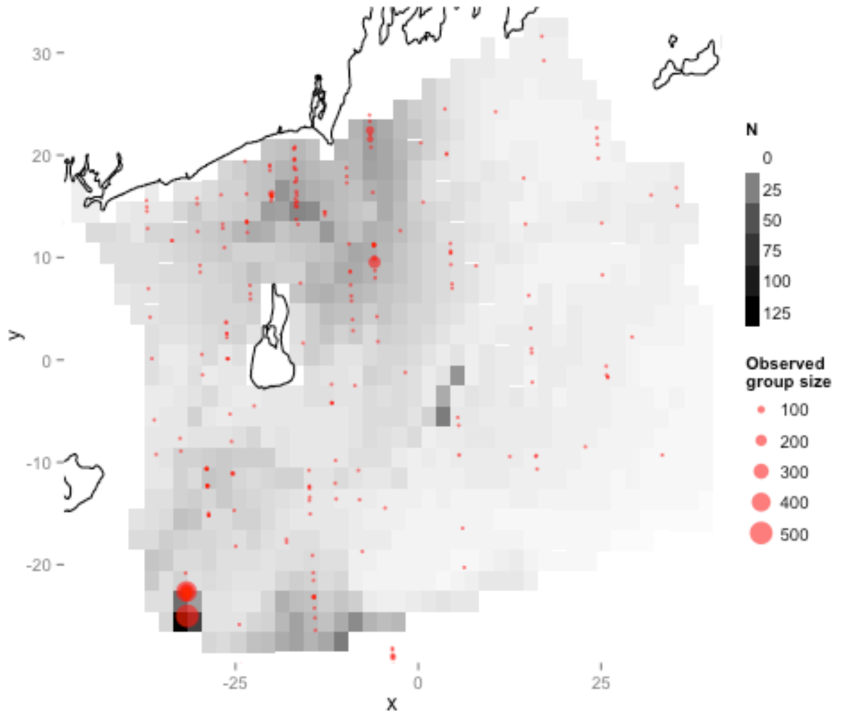
Autocorrelogram for the 1 winter gannet model.

We see some mild autocorrelation in the residuals.

We can now plot the predicted abundance map:

```
plot.preds(gannet.model.1w, obs = obs.gannet.1w)
```

```
## Abundance = 1993
```



Predicted abundance over the OSAMP area for 1 winter model.

And calulcate the associated uncertainty:

```
pred$height <- pred$width <- 2
pred.grid <- split(pred, 1:nrow(pred))
gannet.var.grid <- dsm.var.prop(gannet.model.1w, pred.grid, pred$cellarea)
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: fitting to calculate the null deviance did not converge --
## increase 'maxit'?
```

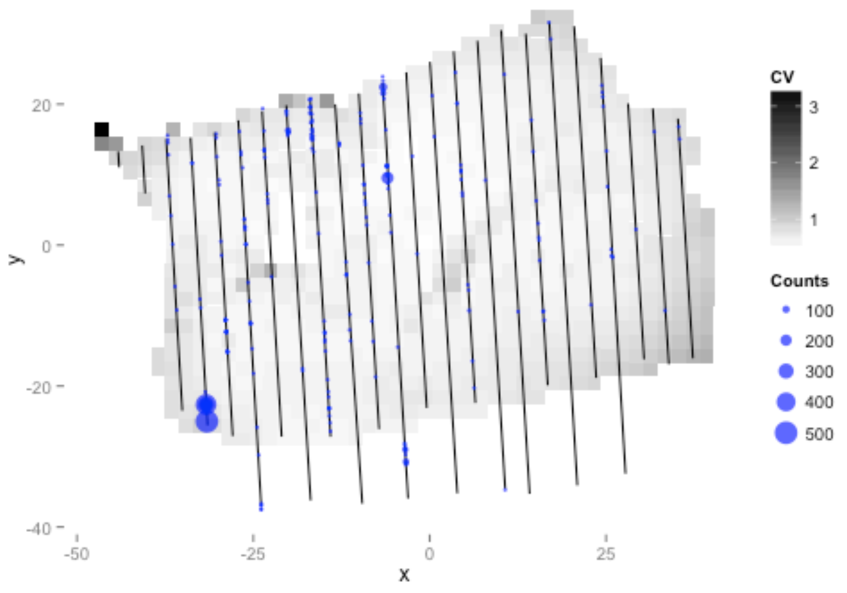
```
summary(gannet.var.grid)
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: fitting to calculate the null deviance did not converge --
## increase 'maxit'?
## Summary of uncertainty in a density surface model calculated
##  by variance propagation.
##
## Quantiles of differences between fitted model and variance model
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -0.03370 -0.00007  0.00004  0.00041  0.00046  0.04240
##
## Approximate asymptotic confidence interval:
##   5% Mean  95%
## 1054 1993 3766
## (Using delta method)
##
##
## Point estimate                 : 1993
## Standard error                 : 664.6
## Coefficient of variation       : 0.3336
```

```
plot(gannet.var.grid)
```



Plot of uncertainty in predicted abundance over the OSAMP area for 1 winter model.

Save everything

```
save.image("gannet-models.RData")
```

Save predictions for Zonation:

```
gannet.preds <- predict(gannet.model.1w, pred, pred$cellaream)

gannet.pred.data <- data.frame(cellid = 1:920, pred = gannet.preds, var = diag(gannet.var.grid$pred.var),
    cv = sqrt(diag(gannet.var.grid$pred.var))/gannet.preds, season = rep("Winter",
        920))
save(gannet.pred.data, file = "gannet-preds.RData")
```

References

Xie Y (2013) knitr: A general-purpose package for dynamic report generation in R. R package version 1.0. http://CRAN.R-project.org/package=knitr

Eider analysis

This document is a record of modelling for the eider data from the URI aerial line transect survey of the OSAMP area off the coast of Rhode Island. It is provided as a `knitr` (Xie 2013) file, that includes all the necessary code to re-create the analysis.

```
suppressPackageStartupMessages(library(osampuri))
data("uri-lt-data")
```

Loading up the data and trimming for what we want...

```
obs.eider <- obs[obs$Group == "Eider", ]
rm(obs)
```

Looking at only those Eider who were on water...

```
obs.eider <- obs.eider[obs.eider$Location == "On Water", ]
```

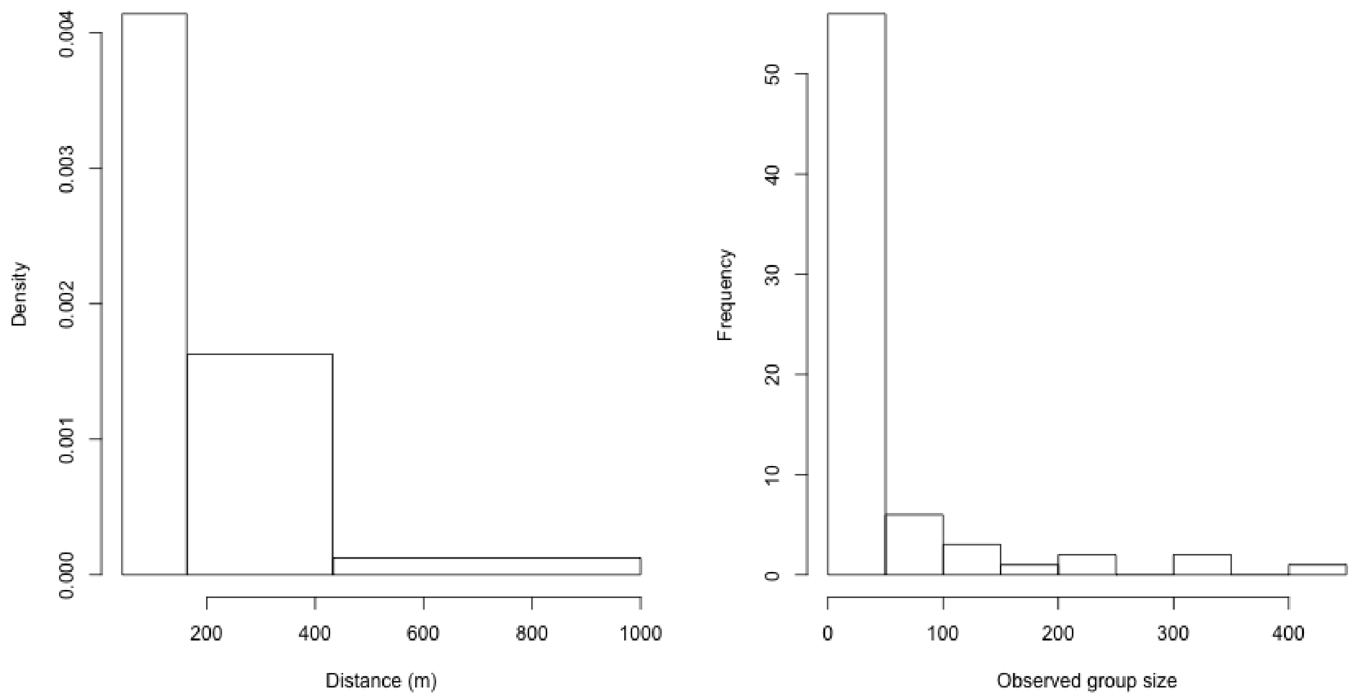
and in the Winter...

```
obs.eider <- obs.eider[obs.eider$Season == "Winter", ]  # |
effort <- effort[effort$Season == "Winter", ]
```

EDA plots

Before fitting a detection function, we first plot a histogram of distances and group sizes:

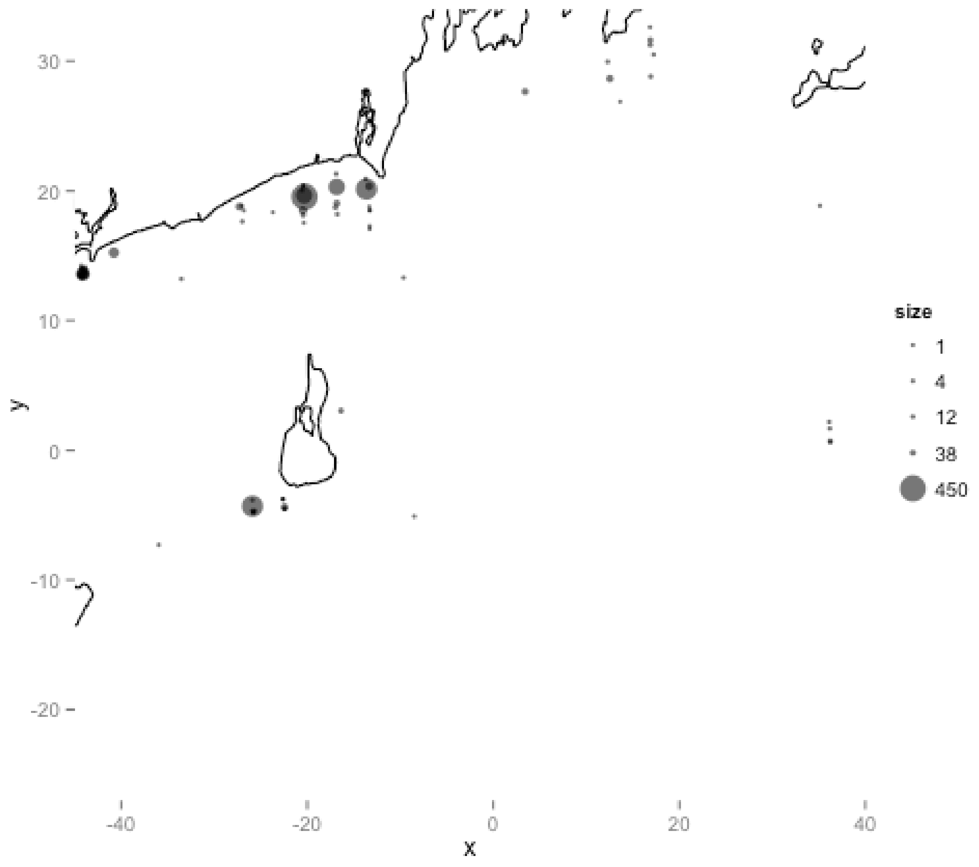
```
par(mfrow = c(1, 2))
hist(obs.eider$distance, breaks = sort(unique(c(obs.eider$distbegin, obs.eider$distend))),
    main = "", xlab = "Distance (m)")
hist(obs.eider$size, xlab = "Observed group size", main = "")
```



Histogram of distances (left) and observed group sizes (right).

Plotting the raw observations:

```
plot.uri.data(obs.eider)
```



Raw observations of eider.

Detection function fitting

Half-normal and hazard rate without adjustments:

```
hn.df <- ds(obs.eider, adjustment = NULL, truncation = list(right = 1000, left = 44))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...

## Fitting half-normal key function

## AIC= 130.693

## No survey area information supplied, only estimating detection function.
```

```
hr.df <- ds(obs.eider, adjustment = NULL, truncation = list(right = 1000, left = 44),
    key = "hr")
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...
## Fitting hazard-rate key function

## AIC= 132.037

## No survey area information supplied, only estimating detection function.
```

Group size as covariate

Using group size as a covariate, we must first standardise the group size variable to avoid numerical problems in fitting:

```
obs.eider$ssize <- obs.eider$size/sd(obs.eider$size)
hn.df.size <- ds(obs.eider, formula = ~ssize, adjustment = NULL, truncation = list(right = 1000,
    left = 44))
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...

## Fitting half-normal key function

## AIC= 125.117

## No survey area information supplied, only estimating detection function.
```

```
hr.df.size <- ds(obs.eider, formula = ~ssize, adjustment = NULL, truncation = list(right = 1000,
    left = 44), key = "hr")
```

```
## Warning: No cutpoints specified but distbegin and distend are columns in
## data. Performing a binned analysis...

## Fitting hazard-rate key function

## AIC= 127.44

## No survey area information supplied, only estimating detection function.
```

Size as a covariate with a half-normal key gives a significantly lower AIC.

Spatial model

To begin the spatial modelling, we must first allocate the effort:

```
correct.effort <- allocate.effort(obs.eider, seg)
obs.eider <- correct.effort$obs
seg <- correct.effort$seg
```

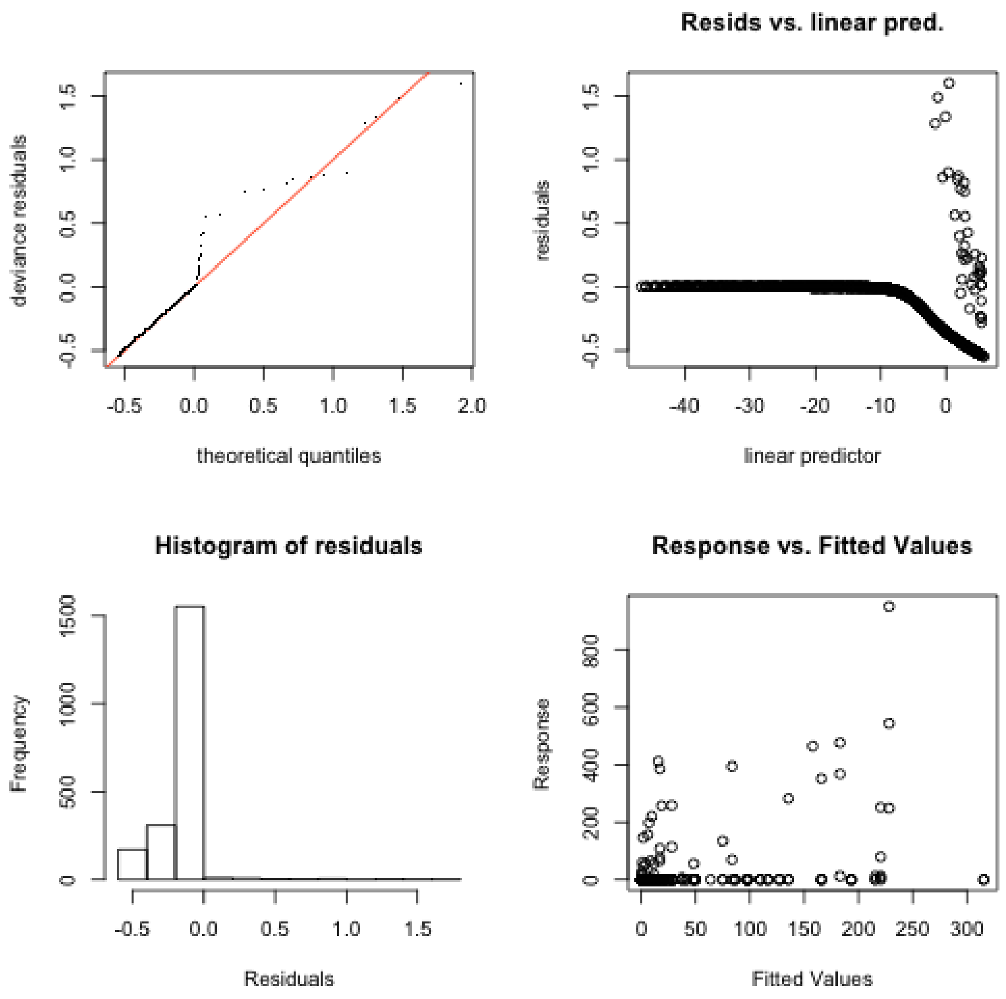
```
k1 <- 7
k2 <- 20
eider.model <- dsm(Nhat~ #s(roughness,k=k1)+
#                  s(gchl_winter,k=k1)+
#                  s(gchl_fall,k=k1)+
#                  s(phimedian,k=k1)+
#                  s(distancelandkm,k=k1)+
                  distancelandkm+
                  s(depthm,k=k1)+
#                  depthm+
                  s(x,y,k=k2),#+
#                  s(x,k=k1),#+
#                  s(y,k=k1),
                hn.df.size, seg, obs.eider,
#                family=negbin(theta=0.012),
#                family=negbin(theta=0.029),
#                family=Tweedie(p=1.1),
                family=negbin(theta=c(0.01,0.015)),
                select=TRUE,method="REML") #,control=list(maxit=5000))
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: algorithm did not converge

## Warning: fitting to calculate the null deviance did not converge --
## increase maxit?
```

```
summary(eider.model)
```

```
##
## Family: Negative Binomial(0.015)
## Link function: log
##
## Formula:
## Nhat ~ distancelandkm + s(depthm, k = k1) + s(x, y, k = k2) +
##     offset(off.set)
## <environment: 0x10b193718>
##
## Parametric coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -14.491      2.621   -5.53  3.2e-08 ***
## distancelandkm   -0.844      0.141   -5.99  2.1e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##            edf Ref.df Chi.sq p-value
## s(depthm) 3.07      9   25.8 7.8e-07 ***
## s(x,y)    4.75     19   16.9 0.00084 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0683   Deviance explained = 95.3%
## REML score = 415.34  Scale est. = 1         n = 2067
```

```
gam.check(eider.model)
```



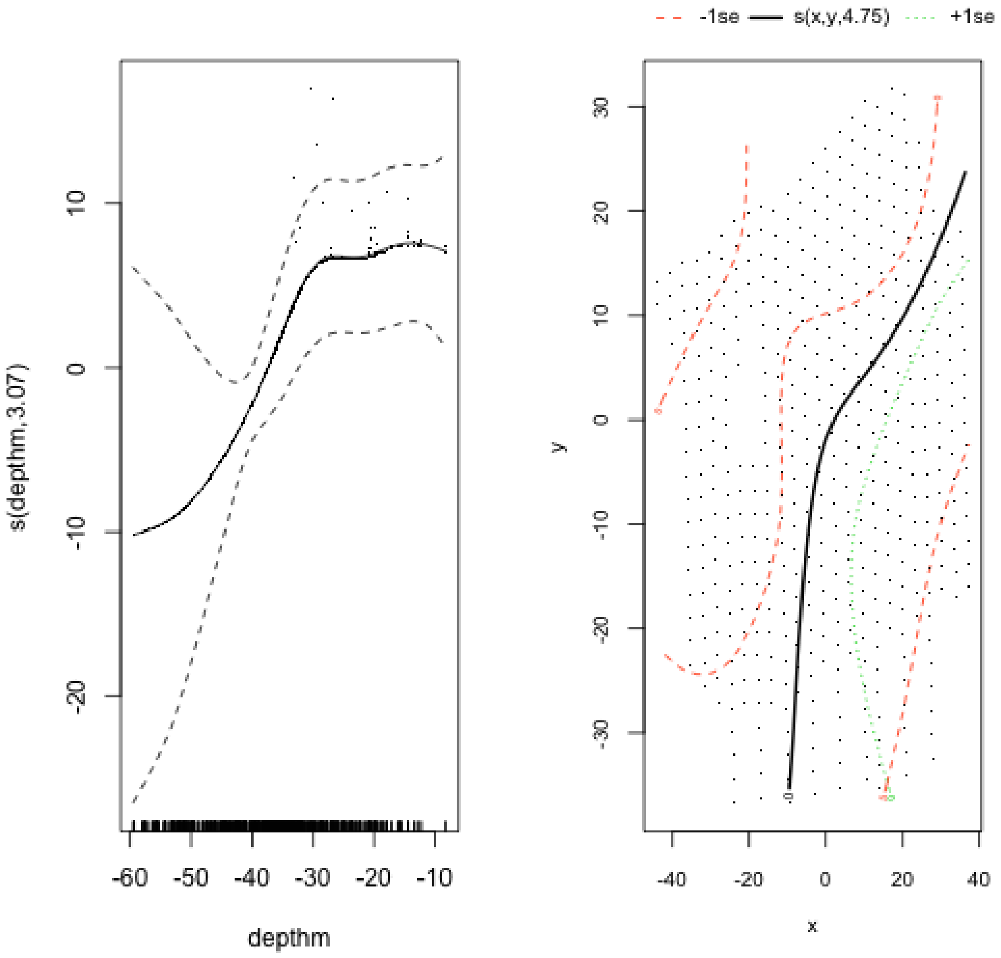
GAM check plots for DSM.

```
##
## Method: REML   Optimizer: outer newton
## step failed after 1 iteration.
## Gradient range [-0.007737,0.0007353]
## (score 415.3 & scale 1).
## Hessian positive definite, eigenvalue range [1.186e-05,1.177].
##
## 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(depthm)   9.000  3.067   0.374       0
## s(x,y)     19.000  4.746   0.367       0
```

Percentage deviance explained seems extremely high: this is explained by the error `fitting to calculate the null deviance did not converge -- increase maxit?`. Increasing `maxit` doesn't appear to help here and the deviance is not correctly calculated.

What does a plot of the smooths look like?

```
plot(eider.model, scale = 0, pages = 1, residuals = TRUE)
```

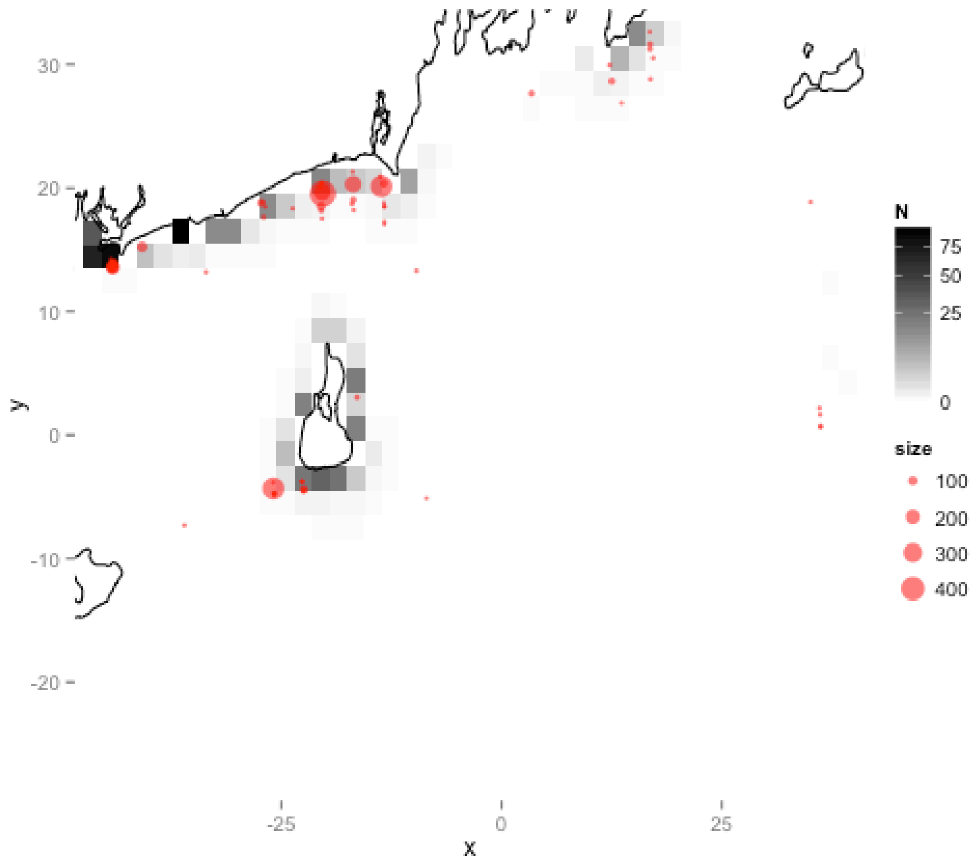


Smooth terms in `eider.model`.

Looking at a plot of the predictive map:

```
plot.preds(eider.model, obs.eider)
```

```
## Abundance = 551.5
```



Predictions from `eider.model`.

In the northwest corner of the SAMP grid there is a very high observation, this has a value of:

```
max(predict(eider.model, pred, pred$cellaream))
```

```
## [1] 92.84
```

```
which.max(predict(eider.model, pred, pred$cellaream))
```

```
## 766
## 767
```

this coincides with the maximum depth value:

```
max(pred$depth)
```

```
## [1] -7.074
```

```
which.max(pred$depth)
```

```
## [1] 766
```

which is quite far from the next biggest value:

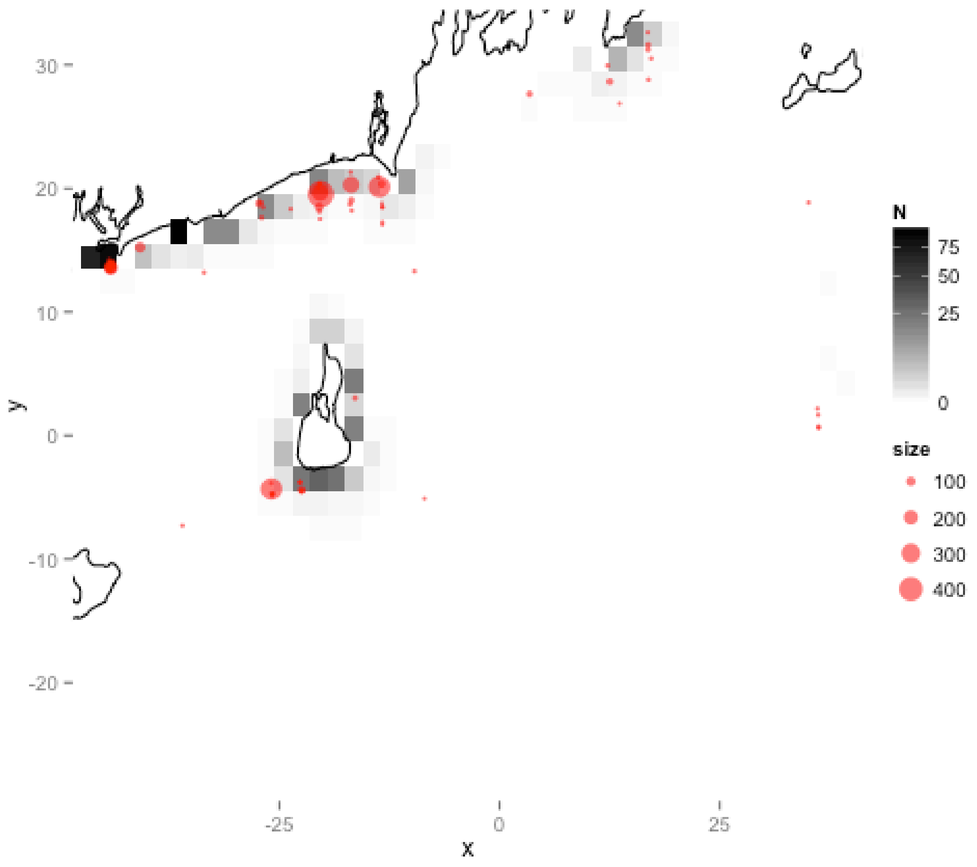
```
max(pred$depth[-766])
```

```
## [1] -29.84
```

So, removing that prediction from the grid and re-estimating the abundance map and the estimate:

```
pred.save <- pred
pred <- pred[-766, ]
plot.preds(eider.model, obs.eider)
```

```
## Abundance = 515.8
```



Plot of predicted model with high depth value removed.

```
pred <- pred.save
```

The abundance has been significantly reduced by removing one cell in the prediction grid. Although this seems like a much more reasonable prediction in the sense that 91% of the predicted abundance does not lie at one cell. However, this would tend to indicate the using a DSM is inadvisable if the predictions are so sensitive.

(A model with bivariate smooth of location will not fit due to numerical issues.)

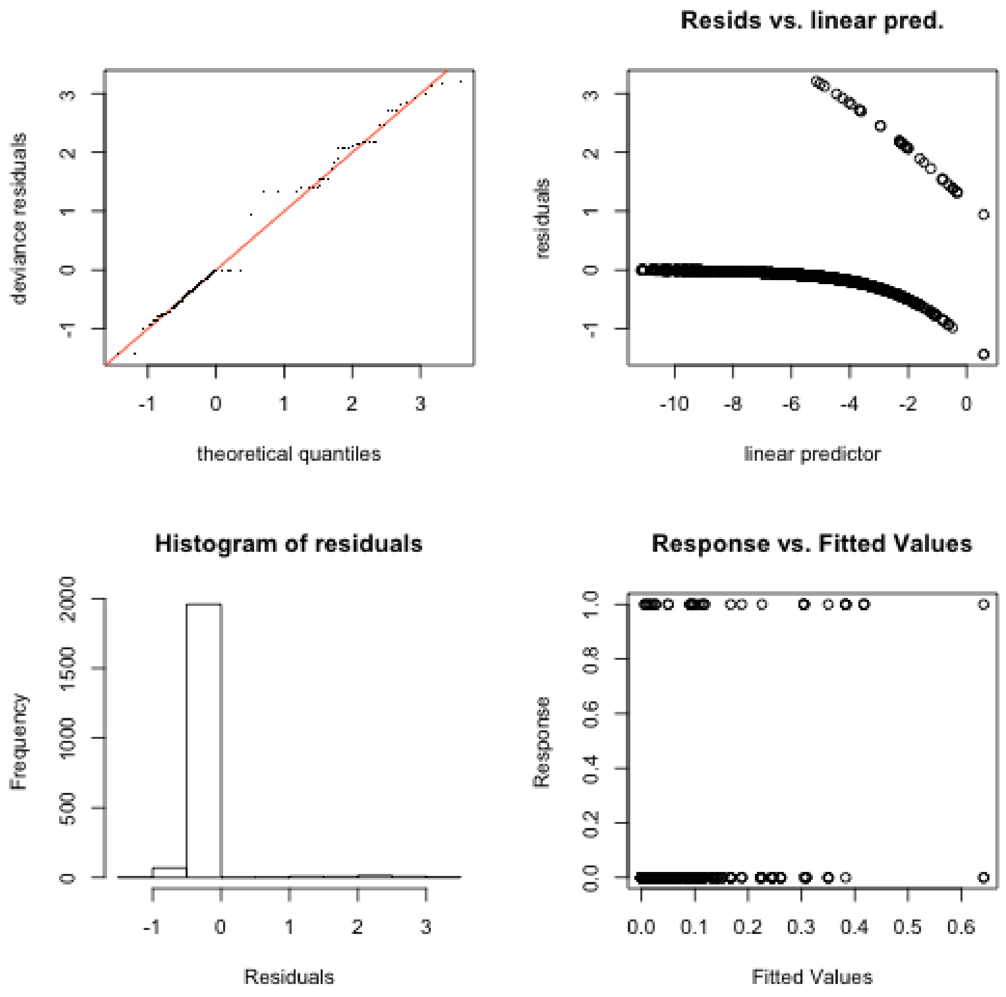
Presence/absence model

We could build a presence/absence model instead, if we don't have confidence in the DSM...

```
k1<-10
k2<-20
eider.model.pr <- dsm(presence~ #s(roughness,k=k1)+
                                #s(gchl_winter,k=k1)+
                                #s(gchl_winter,k=k1)+
                                #s(phimedian,k=k1)+
                                #s(depthm,k=k1)+
                                depthm,#+
                                #s(x,k=k1)+
                                #s(y,k=k1)+
                                #s(x,y,k=k2),
                      hn.df, seg, obs.eider,
                      family=binomial(),
                      select=TRUE,method="REML")
summary(eider.model.pr)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## presence ~ depthm
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.4582     0.5657    4.35  1.4e-05 ***
## depthm        0.2284     0.0254    9.01  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## R-sq.(adj) =  0.156   Deviance explained = 33.8%
## REML score = 136.93  Scale est. = 1         n = 2067
```

```
gam.check(eider.model.pr)
```



GAM check plots for presence/absence model.

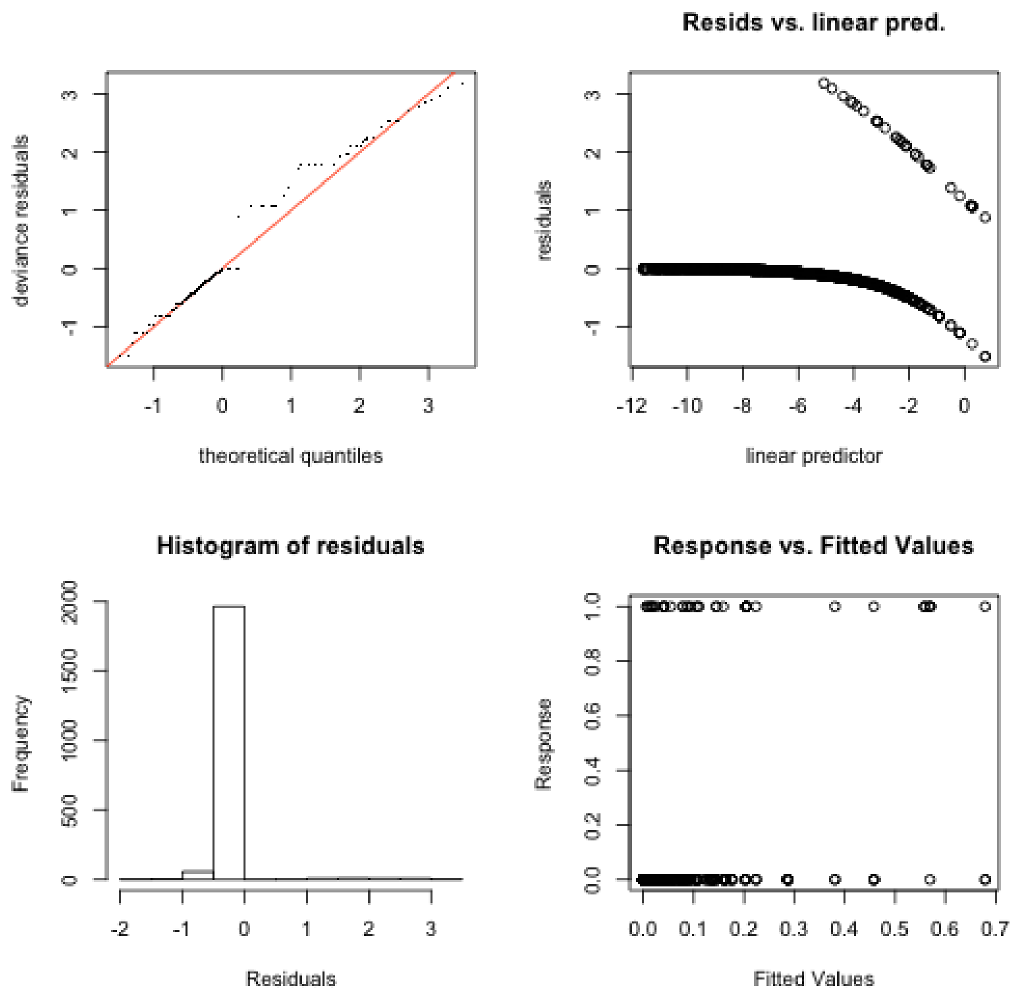
```
##
## Method: REML   Optimizer: outer newton
## Model required no smoothing parameter selection
```

Here `phimedian` is not significant at the 0.05 level. A linear model with only depth doesn't seem like a particularly useful model though. What about a bivariate smooth of location?

```
k1<-10
eider.model.pr.xy <- dsm(presence~ #s(roughness,k=k1)+
                                #s(gchl_winter,k=k1)+
                                #s(gchl_fall,k=k1)+
                                #s(phimedian,k=k1)+
                                #s(depthm,k=k1)+
                                depthm+
                                s(x,y,k=k1),
                      hn.df, seg, obs.eider,
                      family=binomial(),
                      select=TRUE,method="REML")
summary(eider.model.pr.xy)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## presence ~ depthm + s(x, y, k = k1)
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.7384     0.8733    1.99    0.047 *
## depthm        0.2120     0.0304    6.96  3.3e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##         edf Ref.df Chi.sq p-value
## s(x,y) 3.94      9   12.7  0.0055 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.193   Deviance explained = 37.9%
## REML score = 134.63  Scale est. = 1         n = 2067
```

```
gam.check(eider.model.pr.xy)
```

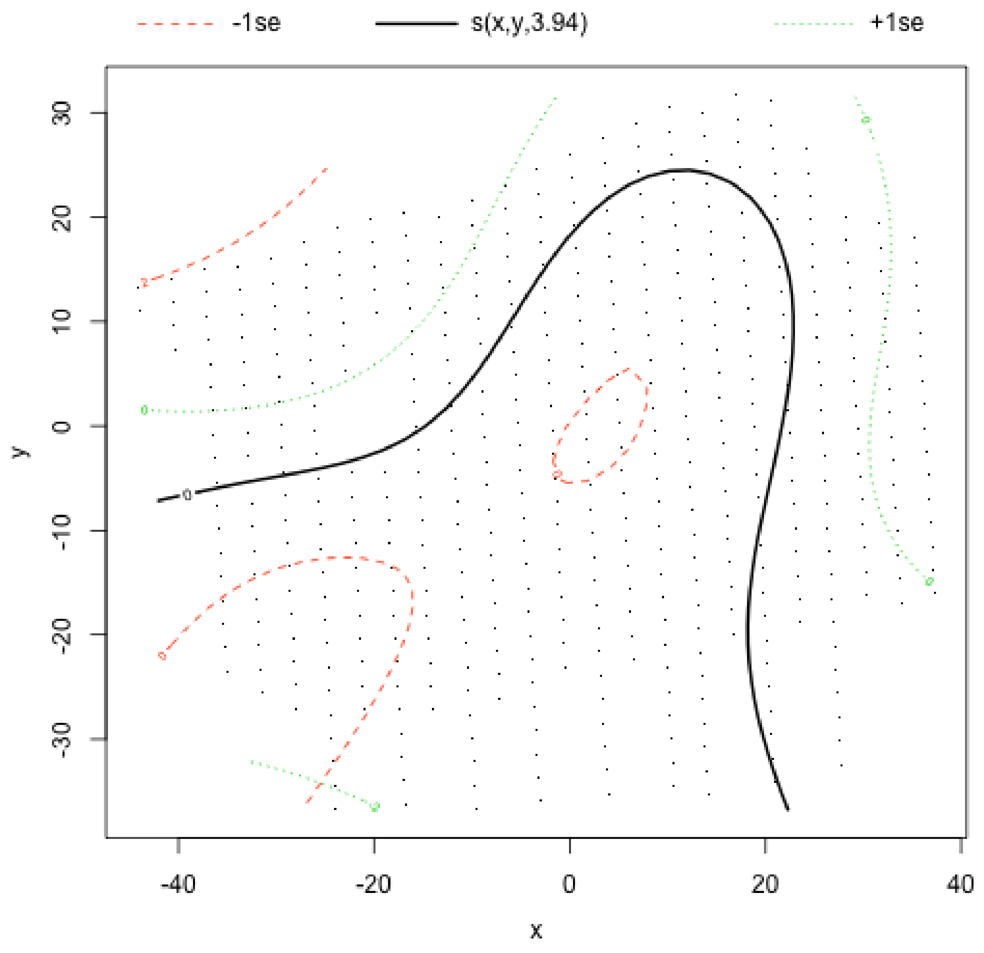


GAM check plots for presence/absence model with bivariate smooth of location.

```
##
## Method: REML   Optimizer: outer newton
## full convergence after 7 iterations.
## Gradient range [-9.319e-09,4.153e-09]
## (score 134.6 & scale 1).
## Hessian positive definite, eigenvalue range [0.2017,1.076].
##
## 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(x,y) 9.000 3.945   0.817       0
```

Well these seem much more reasonable. There is little difference in REML score (which in this case we can compare as we have the same fixed effects), but the model with a bivariate smooth of locations appears to explain slightly more of the deviance.

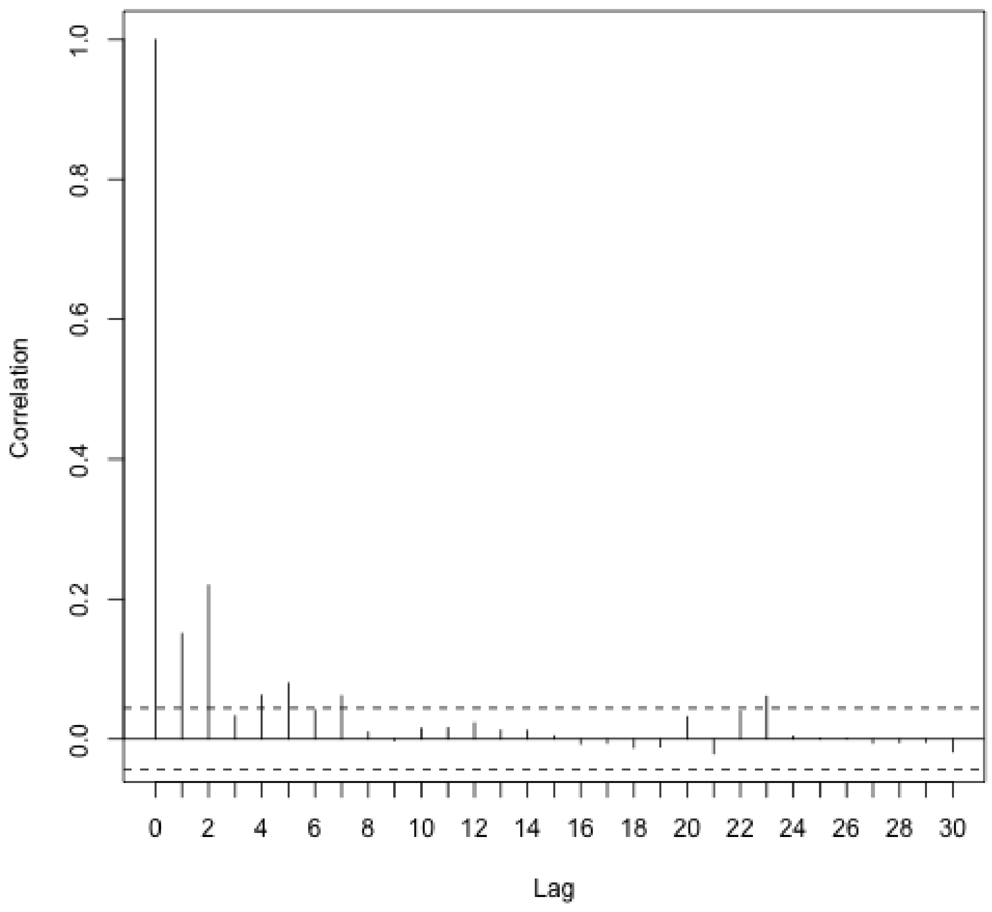
```
plot(eider.model.pr.xy, pages = 1, scale = 0)
```



Smooth terms in `eider.model.pr.xy`.

The correlogram shows that there is a small ammount of unmodelled autocorrelation between the residuals, at low lag values.

```
dsm.cor(eider.model.pr.xy, max.lag = 30)
```

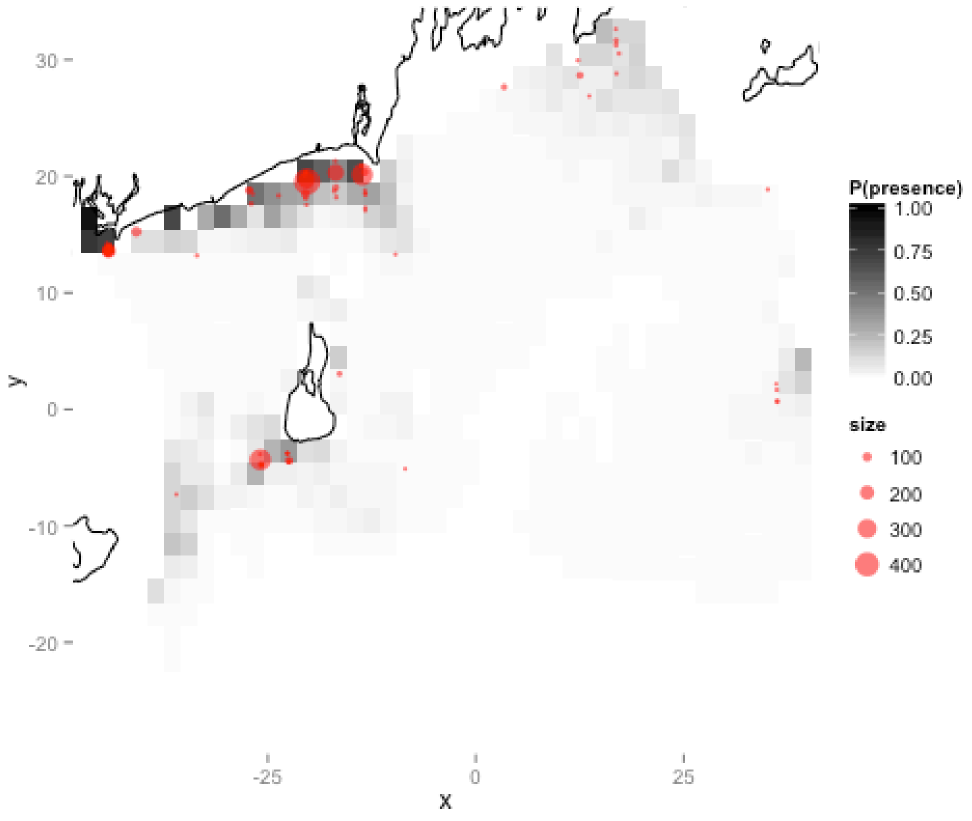


Correlogram for `eider.model.pr.xy`

Presence map

We can then produce a map of probability of presence

```
eider.predict.pr <- predict(eider.model.pr.xy, pred, 1)
eider.predict.pr <- cbind(pred, N = eider.predict.pr)
p <- ggplot(eider.predict.pr, aes(x, y))
p.opts.geo <- theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
    panel.background = element_blank(), strip.background = element_blank(),
    legend.key = element_blank(), aspect.ratio = 1)
p <- p + p.opts.geo
p <- p + geom_tile(aes(fill = N, height = 2, width = 2))
p <- p + geom_polygon(aes(x = x, y = y, group = group), colour = "black", fill = NA,
    data = coast)
xlims <- c(min(pred$x) - 2, max(pred$x) + 2)
ylims <- c(min(pred$y) - 2, max(pred$y) + 2)
p <- p + coord_equal(xlim = xlims, ylim = ylims)
p <- p + scale_fill_gradient(low = "white", high = "black", limits = c(0, 1))
p <- p + geom_point(aes(x = x, y = y, size = size), colour = "red", data = obs.eider,
    alpha = 0.6)
p <- p + labs(fill = "P(presence)")
print(p)
```



Map of predicted probability of presence.

Save everything

```
save.image("eider-models.RData")
```

And save files for Zonation input.

```
eider.preds <- eider.predict.pr$N

pred.grid <- split(pred, 1:nrow(pred))
eider.var.grid <- dsm.var.gam(eider.model.pr.xy, pred.grid, split(rep(1, nrow(pred)),
    1:nrow(pred)))
# plot(eider.var.grid)

eider.pred.data <- data.frame(cellid = 1:920, pred = eider.preds, var = diag(eider.var.grid$pred.var),
    cv = sqrt(diag(eider.var.grid$pred.var))/eider.preds, season = rep("Winter",
        920))
save(eider.pred.data, file = "eider-preds.RData")
```

References

Xie Y (2013) knitr: A general-purpose package for dynamic report generation in R. R package version 1.0. http://CRAN.R-project.org/package=knitr