Advanced topics in distance sampling

Workshop, 26-30 August 2019

Centre for Research into Ecological and Environmental Modelling

Exercise 7. Estimating precision of predictions from density surface models

Now we've fitted some models and estimated abundance, we can estimate the variance associated with the abundance estimate (and plot it).

Aims

By the end of this practical, you should feel comfortable:

- Knowing when to use `dsm.var.prop` and when to use `dsm.var.gam`
- Estimating variance for a given prediction area
- Estimating variance per-cell for a prediction grid
- Interpreting the `summary()` output for uncertainty estimates
- Making maps of the coefficient of variation in R
- Saving uncertainty information to a raster file to be read by ArcGIS

Load packages and data

```
library(dsm)
```

```
## Loading required package: mgcv
```

```
## Loading required package: nlme
```

```
## This is mgcv 1.8-28. For overview type 'help("mgcv-package")'.
```

```
## Loading required package: mrds
```

```
## This is mrds 2.2.1
## Built: R 3.6.1; ; 2019-07-17 13:15:38 UTC; windows
```

```
## Loading required package: numDeriv
```

```
## This is dsm 2.2.17
## Built: R 3.6.1; ; 2019-07-11 19:38:05 UTC; windows
```

```
library(raster)
```

```
## Loading required package: sp
```

```
##
## Attaching package: 'raster'
```

```
## The following object is masked from 'package:nlme':
##
##     getData
```

```
library(ggplot2)
library(viridis)
```

```
## Loading required package: viridisLite
```

```
library(plyr)
library(knitr)
library(rgdal)
```

```
## rgdal: version: 1.4-4, (SVN revision 833)
##  Geospatial Data Abstraction Library extensions to R successfully loaded
##  Loaded GDAL runtime: GDAL 2.2.3, released 2017/11/20
##  Path to GDAL shared files: C:/Users/louise/Documents/R/win-library/3.6/rgdal/gdal
##  GDAL binary built with GEOS: TRUE
##  Loaded PROJ.4 runtime: Rel. 4.9.3, 15 August 2016, [PJ_VERSION: 493]
##  Path to PROJ.4 shared files: C:/Users/louise/Documents/R/win-library/3.6/rgdal/pr
##  Linking to sp version: 1.3-1
```

Load the models and prediction grid:

```
load("dsms.RData")
load("dsms-xy.RData")
load("predgrid.RData")
```

Estimation of variance

Depending on the model response (count or Horvitz-Thompson), we can use either `dsm.var.prop` or `dsm.var.gam`, respectively. `dsm_nb_xy_ms` doesn't include any covariates at the observer level in the detection function, so we can use `dsm.var.gam` to estimate the uncertainty.

```
# need to remove the NAs as we did when plotting
predgrid_var <- predgrid[!is.na(predgrid$Depth), ]
# now estimate variance
var_nb_xy_ms <- dsm.var.gam(dsm_nb_xy_ms, predgrid_var,
                            off.set=predgrid_var$off.set)
```

To summarise the results of this variance estimate:

```
summary(var_nb_xy_ms)
```

```
## Summary of uncertainty in a density surface model calculated
##  analytically for GAM, with delta method
##
## Approximate asymptotic confidence interval:
##     2.5%     Mean    97.5%
## 1123.709 1589.216 2247.565
```

```
## (Using log-Normal approximation)
##
## Point estimate                  : 1589.216
## CV of detection function        : 0.06670757
## CV from GAM                     : 0.1653
## Total standard error            : 283.2538
## Total coefficient of variation : 0.1782
```

Try this out for some of the other models you've saved. Remember the rule when there are covariates in the detection function model:

- use `dsm.var.prop` if there are covariates in the detection function that vary at the level of the segment (like Beaufort) and
- use `dsm.var.gam` if there are individual-level covariates (like observer) in the detection function

If there are no covariates in the detection function, use `dsm.var.gam` since there's no covariance between the detection function and the spatial model.

Summarise multiple models

We can again summarise all the models, as we did with the DSMs and detection functions, now including the variance:

```
# This function harvests relevant statistics
summarize_dsm_var <- function(model, predgrid) {

  summ <- summary(model)
  vp <- summary(dsm.var.gam(model, predgrid, off.set=predgrid$off.set))
  unconditional.cv.square <- vp$cv^2
  asymp.ci.c.term <- exp(1.96*sqrt(log(1+unconditional.cv.square)))
  asymp.tot <- c(vp$pred.est / asymp.ci.c.term,
                 vp$pred.est,
                 vp$pred.est * asymp.ci.c.term)

  data.frame(response = model$family$family,
             terms    = paste(rownames(summ$s.table), collapse=", "),
             AIC      = AIC(model),
             REML     = model$gcv.ubre,
             "Dev_exp" = paste0(round(summ$dev.expl*100,2),"%"),
             "low_CI" = round(asymp.tot[1],2),
             "Nhat" = round(asymp.tot[2],2),
             "up_CI" = round(asymp.tot[3],2)
             )
}
```

```
# make a list of models (add more here!)
model_list <- list(dsm_nb_xy, dsm_nb_x_y, dsm_nb_xy_ms, dsm_nb_x_y_ms)
# give the list names for the models, so we can identify them later
```

```
names(model_list) <- c("dsm_nb_xy", "dsm_nb_x_y", "dsm_nb_xy_ms", "dsm_nb_x_y_ms")
# Apply the summary function to list of models
per_model_var <- ldply(model_list, summarize_dsm_var, predgrid=predgrid_var)
```

```
kable(per_model_var, digits=1, booktabs=TRUE, escape=TRUE,
      caption = "Model performance: bivariate vs univariate spatial smooths without a
```

Table 1: Model performance: bivariate vs univariate spatial smooths without and with environmental covariates.

.id	response	terms	AIC	REML	Dev_exp	low_CI
dsm_nb_xy	Negative Binomial(0.105)	s(x,y)	775.3	392.6	40.63%	1135.5
dsm_nb_x_y	Negative Binomial(0.085)	s(x), s(y)	789.8	395.9	31.27%	1091.2
dsm_nb_xy_ms	Negative Binomial(0.108)	s(x,y), s(Depth)	758.1	384.8	37.65%	1123.7
dsm_nb_x_y_ms	Negative Binomial(0.098)	s(y), s(Depth)	762.6	386.1	35.88%	1193.0

```
#  kable_styling(latex_options="scale_down")
```

Plotting

We can plot a map of the coefficient of variation, but we first need to estimate the variance per prediction cell, rather than over the whole area. This calculation takes a while!

```
# use the split function to make each row of the predictiond data.frame into
# an element of a list
predgrid_var_split <- split(predgrid_var, 1:nrow(predgrid_var))
var_split_nb_xy_ms <- dsm.var.gam(dsm_nb_xy_ms, predgrid_var_split,
                                  off.set=predgrid_var$off.set)
```

Now we have the per-cell coefficients of variation, we assign that to a column of the prediction grid data and plot it as usual:

```
predgrid_var_map <- predgrid_var
cv <- sqrt(var_split_nb_xy_ms$pred.var)/unlist(var_split_nb_xy_ms$pred)
predgrid_var_map$CV <- cv
p <- ggplot(predgrid_var_map) +
       geom_tile(aes(x=x, y=y, fill=CV, width=10*1000, height=10*1000)) +
       scale_fill_viridis() +
       coord_equal() +
       geom_point(aes(x=x,y=y, size=count),
                  data=dsm_nb_xy_ms$data[dsm_nb_xy_ms$data$count>0,])
print(p)
```

Note that here we overplot the segments where sperm whales were observed (and scale the size of the point according to the number observed), using `geom_point()`.

We can also overplot the effort, which can be a useful way to see what the cause of uncertainty is. Though it may not only be caused by lack of effort but also covariate

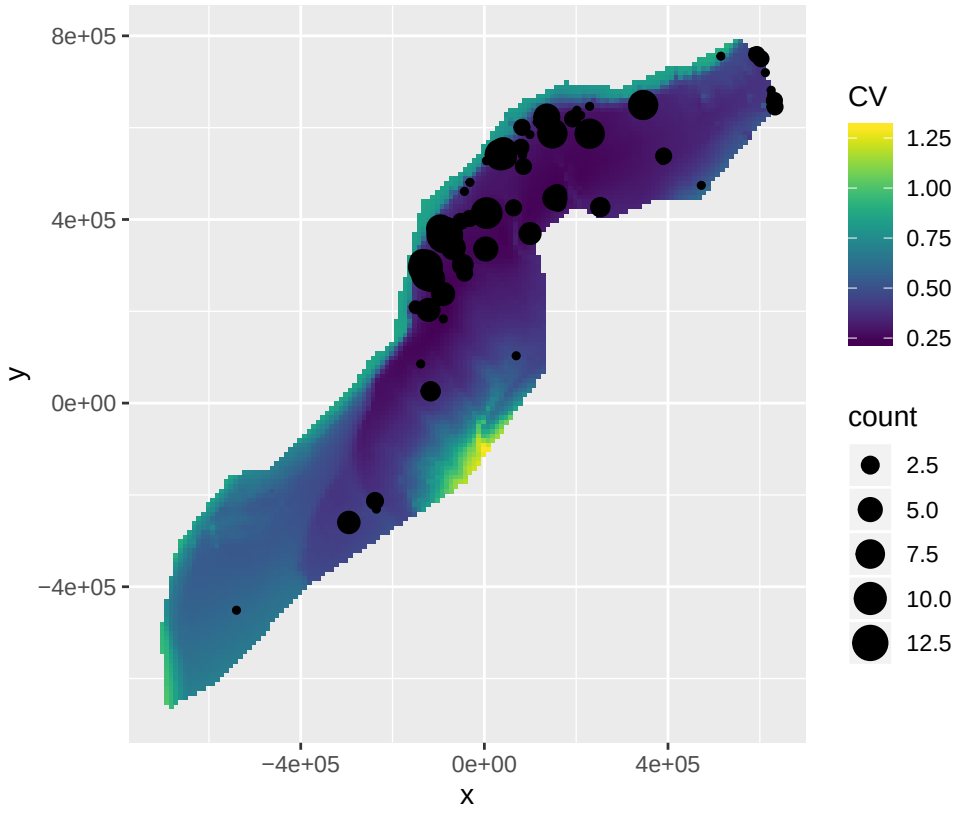

Figure 1: Uncertainty (CV) in prediction surface from bivariate spatial smooth with environmental covariates. Sightings overlaid.

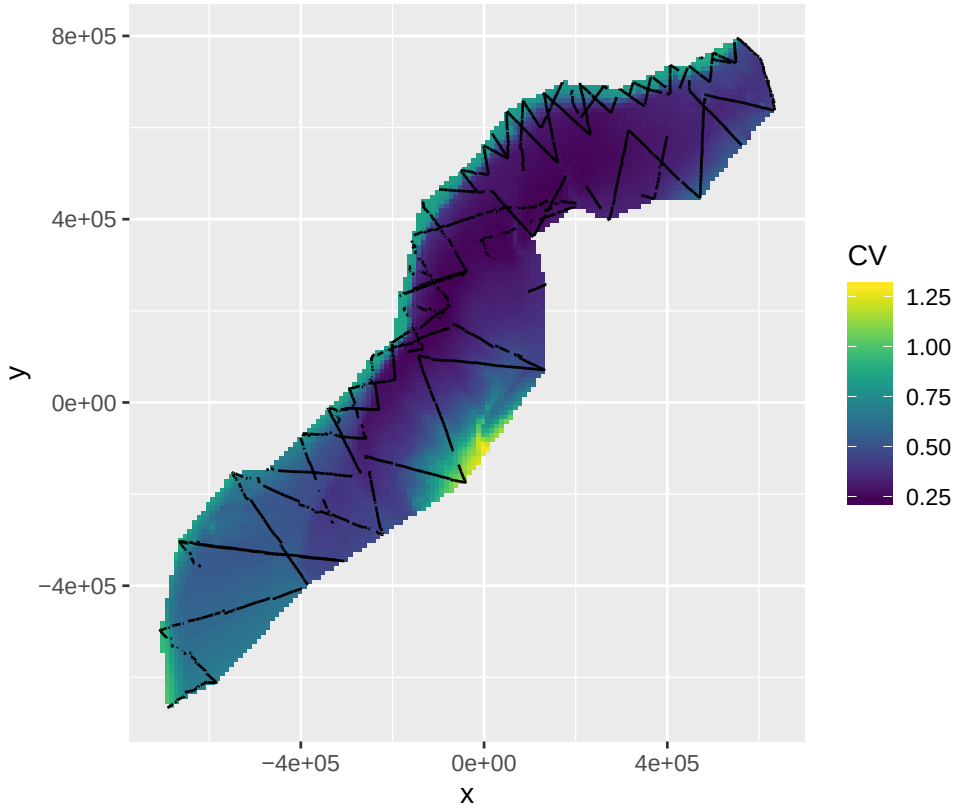

Figure 2: Uncertainty (CV) in prediction surface from bivariate spatial smooth with environmental covariates. Effort overlaid.

coverage, this can be useful to see.

First we need to load the segment data from the database `gdb`:

```
tracks <- readOGR("Analysis.gdb", "Segments")
```

```
## OGR data source with driver: OpenFileGDB
## Source: "C:\workshops\2019\Advanced DS\Practicals\Analysis.gdb", layer: "Segments"
## with 949 features
## It has 8 fields
```

```
tracks <- fortify(tracks)
```

We can then just add this to the plot object we have built so far (with `+`), but this looks a bit messy with the observations, so let's start from scratch:

```
p <- ggplot(predgrid_var_map) +
       geom_tile(aes(x=x, y=y, fill=CV, width=10*1000, height=10*1000)) +
       scale_fill_viridis() +
       coord_equal() +
       geom_path(aes(x=long, y=lat, group=group), data=tracks)
print(p)
```

Try this with the other models you fitted and see what the differences are between the maps of coefficient of variation.

Save the uncertainty maps to raster files

As with the predictions, we'd like to save our uncertainty estimates to a raster layer so we can plot them in ArcGIS. Again, this involves a bit of messing about with the data format before we can save.

```
# setup the storage for the cvs
cv_raster <- raster(predictorStack)
# we removed the NA values to make the predictions and the raster needs them
# so make a vector of NAs, and insert the CV values...
cv_na <- rep(NA, nrow(predgrid))
cv_na[!is.na(predgrid$Depth)] <- cv
# put the values in, making sure they are numeric first
cv_raster <- setValues(cv_raster, cv_na)
# name the new, last, layer in the stack
names(cv_raster) <- "CV_nb_xy"
```

We can then save that object to disk as a raster file:

```
writeRaster(cv_raster, "cv_raster.img", datatype="FLT4S", overwrite=TRUE)
```

Extra credit

The functions `dsm.var.prop` and `dsm.var.gam` can accept arbitrary splits in the data, not just whole areas or cells. Make a `list` with two elements: one a `data.frame` of all the cells with $y > 0$ and one with $y \leq 0$. Estimate the variance for these regions. Note that you'll need to sum the offsets for each area to get the correct value to supply to `off.set=...`.