

# Multiple detection functions in density surface models

David L Miller

October 4, 2018

CREEM, University of St Andrews, Scotland

## Abstract

We often want to combine data from multiple surveys into one spatial model. In this case we usually want to include multiple detection functions and use them with a single GAM. Here I show how to do that.

## 1 Introduction

Generally for a DSM we have:

$$\mathbb{E} \left[ n_i | \beta, \lambda, p(\hat{\theta}; \mathbf{z}_i) \right] = a_i p(\hat{\theta}; \mathbf{z}_i) \exp \left( \beta_0 + \sum_m f_m(x_{im}) \right),$$

where the number of individuals per segment (of area  $a_i$ ),  $n_i$ , follows some count distribution such as quasi-Poisson, Tweedie or negative binomial and we assume a log link. The  $f_m$  are smooth functions of environmental covariates,  $x_{im}$ , represented by a basis expansion (i.e.,  $f_m(x) = \sum_j \beta_j b_j(x)$  for some basis functions  $b_j$ ) penalized by a (sum of) quadratic penalty (or penalties);  $\beta_0$  is an intercept term, included in parameter vector  $\beta$ ;  $\lambda$  is a vector of smoothing parameters which control the wiggleness of the smooth components of the model.

Focussing on the  $p(\hat{\theta}; \mathbf{z}_i)$ , we assume here that there is one probability of detection for a given segment,  $i$ . We make no assumption about the source of the  $p(\hat{\theta}; \mathbf{z}_i)$  and, in fact we could have that  $p(\hat{\theta}_1; \mathbf{z}_1)$  is from one detection function and  $p(\hat{\theta}_2; \mathbf{z}_2)$  is from another where  $\hat{\theta} = (\hat{\theta}_1, \hat{\theta}_2)$ , provided that  $\mathbf{z}_1$  and  $\mathbf{z}_2$  are covariates from distinct segments). For clarity I adopt the following notation:  $p_i = p_k(\hat{\theta}_k; \mathbf{z}_i)$  is probability of detection for segment  $i$  from detection function  $k$  (with our “no overlap” rule in place still, so the detectability for segment  $i$  is only calculated by one detection function). Let’s say there are  $K$  detection functions total.

## 1.1 Variance estimation

The above only addresses mean effects, what about variance?

## 1.2 Delta method

Calculate  $CV(\hat{N}) = \sqrt{CV_{\text{GAM}}(\hat{N})^2 + \sum_k CV(\hat{p}_k)^2}$ . There, the  $\text{Var}_{\text{GAM}}(\hat{N})$  (hence  $CV_{\text{GAM}}(\hat{N})$ ) is calculated using the usual GAM estimator (see the varprop paper for details).

## 1.3 Variance via variance propagation

Thinking about the variance propagation method of Bravington, Miller and Hedley (2018)<sup>1</sup>, we need not only the  $p_k(\hat{\boldsymbol{\theta}}; \mathbf{z}_i)$ s but also their derivatives wrt  $\hat{\boldsymbol{\theta}}$  and the Hessian corresponding to the detection functions. Following from the varprop paper, we fit the following model to estimate variance:

$$\log \mathbb{E}[n_i | \boldsymbol{\beta}, \boldsymbol{\lambda}, \hat{p}_i] = \log a_i \hat{p}_i + X_i \boldsymbol{\beta} + \kappa_i \boldsymbol{\delta},$$

defining the vectors  $\boldsymbol{\delta} \triangleq \hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0$  and  $\kappa_i \triangleq \left. \frac{d \log p(\boldsymbol{\theta}, z_i)}{d \boldsymbol{\theta}} \right|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0}$ . Extending this to our case of multiple detection functions, the definition of  $\boldsymbol{\delta}$  follows simply and  $\kappa_i \triangleq \left. \frac{d \log p_k(\boldsymbol{\theta}_k, z_i)}{d \boldsymbol{\theta}_k} \right|_{\boldsymbol{\theta}_k=\boldsymbol{\theta}_{0,k}}$ . We then also need to form the covariance matrix for the detection functions, I think we have to assume no covariance between the detection functions, so we have:

$$\mathbf{V}_{\boldsymbol{\theta}} = \begin{pmatrix} \mathbf{V}_{\boldsymbol{\theta}_1} & 0 & \dots & 0 \\ 0 & \mathbf{V}_{\boldsymbol{\theta}_2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \mathbf{V}_{\boldsymbol{\theta}_K} \end{pmatrix}.$$

finish this thought...

## 2 Implementation

To implement this in `dsm` you need to be able to identify each observation as being from one detection function. You also need to be able to know which segments relate to a given detection function. This leads to some implementation differences in `dsm`.

**Detection function:** One detection function for each data subset (e.g., per cruise etc).

---

<sup>1</sup><https://arxiv.org/abs/1807.07996>

**Observation table:** The observation table is unchanged (it seems easiest to concatenate the tables used to fit the detection function) *but* you must ensure that the object IDs (column `object`) are unique and match those used to fit the detection functions.

**Segment table:** Additional column `ddfobj`, which refers to which detection function is used for each set of segments.

The call to `dsm` now lets you use a list of detection functions, the order of the detection functions in the list relates to the numbering in the `ddfobj` column in the segment table.

### 3 Some special cases and changes to `dsm`

This approach allows us to do some other stuff that we couldn't do before in `dsm`.

#### 3.1 Strip transects

A new function is used to construct dummy detection functions for use with `dsm`. This takes the object IDs, group sizes, truncation and transect type and constructs a model object that can be used with `dsm`. This replaces the `strip.width` way of specifying strip transects. This also works with whatever one might call the point transect equivalent ("circle transects"?).

#### 3.2 Changes

- No longer need to supply the `transect=` argument, this is figured out from the detection functions. This means you can mix points and line and strips and circles.
- No `strip.width` argument, as this is handled by dummy detection functions.
- No `transect` argument as this can always be determined from the detection functions (dummy or not).