Create virtual species with sdmvspecies

April 26, 2014

Chapter 1

Introduction

Simulations of virtual species are more and more pupulation in testing the effects of different aspects of modellling and sampling strategy on performance fo species distribuiton models (SDMs). This vignette illustrates how to create virtual species using sdmvspecies. If you are using a recent version of R , you can install SDMvspecies with this:

```
install.packages(c("raster", "sdmvspecies"))
```

This work still progress, Suggestions are welcomed.

Chapter 2

Select a method

There are many method which can create virtual species. Choice one that you think is suitable for your study.

Niche synthese method

This method bases on paper "Assessing habitat-suitability models with a virtual species" (Hirzel et al., 2001). In this method, the virtual species is generated by creating a simulated ecological niche in an multi-dimensional space, for i th-dimensional space, the ecological niche is calculate as W_i * H_i, W_i is the weight of this space and the H_i is the virtual species's niche suitability index (H ϵ [0,1]). The finally total niche suitability index is built as summarised in:

$$H = \frac{1}{\sum_{i=1}^{n} W_i} \sum_{i=1}^{n} H_i W_i$$

Which H is the niche suitability of that cell, H_i is the value of the i th partial niche coefficient and the W_i is weight of this partial niche suitability. Habitat suitability is composed of many weighted average of partial niche suitability (H_i). Partial niche suitability (H_i) is project from environment by a function. We call this type of function "Niche resposne function", and it projects environment variable to niche suitability. There are five kinds of "Niche response function", and we will explain them latter. Let's do a simple example:

```
> # load the sdmvspecies library
> library("sdmvspecies")
> # load parallel library for use mutilcore CPU
> library("parallel")
> # find package's location
> package.dir <- system.file(package="sdmvspecies")
> # let see where is our sdmvspecies is installed in
> package.dir
```

```
[1] "/usr/local/lib/R/site-library/sdmvspecies"
```

```
> # find env dir under the package's location
> env.dir <- paste(package.dir, "/external/env/", sep="")
> # let see env dir
> env.dir
```

```
[1] "/usr/local/lib/R/site-library/sdmvspecies/external/env/"
```

```
> # get the environment raster file
> env.files <- list.files(env.dir, pattern="*.bil$", full.names=TRUE)
> # see the file list
> env.files
```

```
 [1] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio10.bil"
 [2] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio11.bil"
 [3] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio12.bil"
 [4] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio13.bil"
 [5] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio14.bil"
 [6] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio15.bil"
 [7] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio16.bil"
 [8] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio17.bil"
 [9] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio18.bil"
[10] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio19.bil"
[11] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio1.bil"
[12] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio2.bil"
[13] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio3.bil"
[14] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio4.bil"
[15] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio5.bil"
[16] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio6.bil"
[17] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio7.bil"
[18] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio8.bil"
[19] "/usr/local/lib/R/site-library/sdmvspecies/external/env//bio9.bil"
```

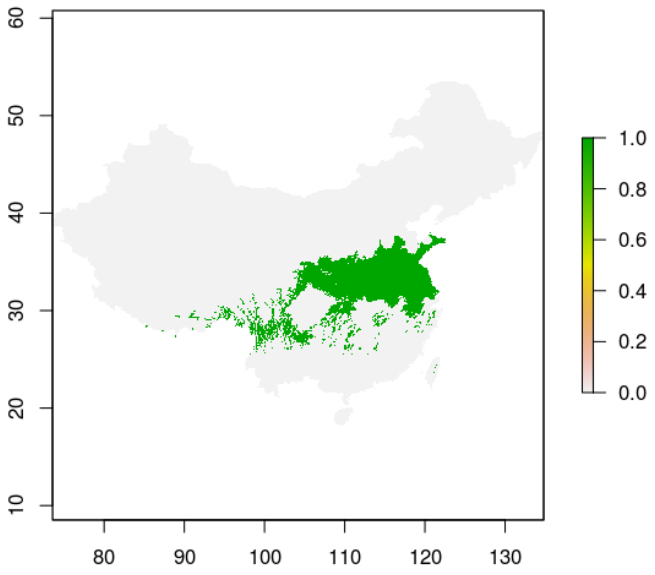
```
> # put the environment file in a raster stack,
> # which require all the environment should have same resolution and extend
> env.stack <- stack(env.files)
> # let see the env.stack var
> env.stack
```

```
class       : RasterStack
dimensions  : 453, 735, 332955, 19  (nrow, ncol, ncell, nlayers)
resolution  : 0.08333334, 0.08333334  (x, y)
extent      : 73.5577, 134.8077, 15.78, 53.53  (xmin, xmax, ymin, ymax)
coord. ref. : +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0
names       : bio10, bio11, bio12, bio13, bio14, bio15, bio16, bio17, bio18, bio19,  bio1,
min values  :   -24,  -291,    12,     2,     0,    17,     5,     0,     5,     0,  -138,
max values  :   319,   219,  4199,   973,   202,   154,  2501,   714,  2501,   971,   258,
```

```
> # here let's configure the environment response function and weight
> config <- list(
+     c("bio1","1",2),
+     c("bio14", "2", 2),
+     c("bio5", "3", 1),
+     c("bio11", "4", 2),
+     c("bio16", "5", 1)
+     )
> # call the niche synthsis method
> species.raster <- nicheSynthese(env.stack, config)
> # let see the result raster,
> # you should noticed that it's continue value map not distributin map
> species.raster

class       : RasterLayer
dimensions  : 453, 735, 332955  (nrow, ncol, ncell)
resolution  : 0.08333334, 0.08333334  (x, y)
extent      : 73.5577, 134.8077, 15.78, 53.53  (xmin, xmax, ymin, ymax)
coord. ref. : +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0
data source : in memory
names       : bio1
values      : 1.833303, 17.66818  (min, max)

> # write the map to file, so you can use it latter in GIS software
> # or further analysis.
> #
> #writeRaster(species.raster, "synthese.img", "HFA", overwrite=TRUE)
>
> # to make binary distribution map, you should chosee a threshold to make map
> # see the map then to decide the threshold to binary
> plot(species.raster)
> # choice threshold, here we choice 4
> threshold <- 14
> # make binary map
> distribution.map <- species.raster > threshold
> # plot the map out
> plot(distribution.map)
```

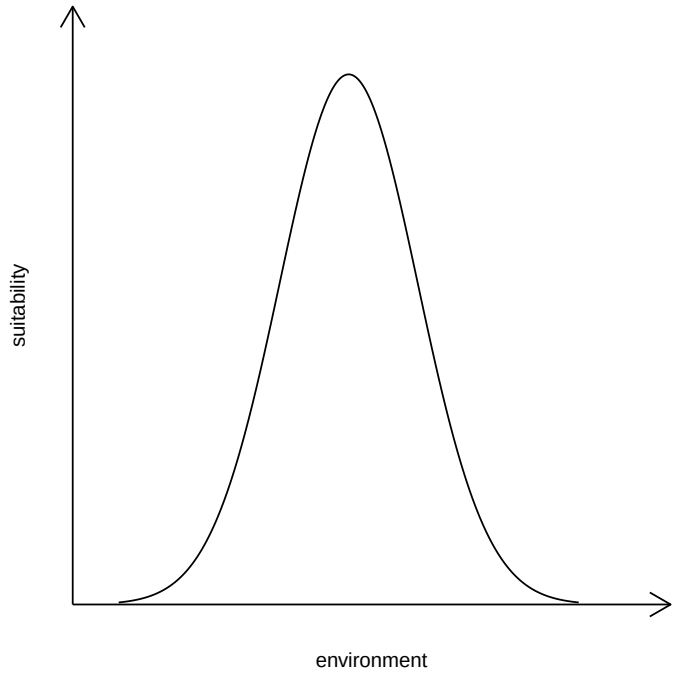


Configure response function

The hardest part of this method is the config. Here we explain that, in this method. You need configure a function: how the the species habit suitability index response the environment variable. There are serveral response function currently that you can used:

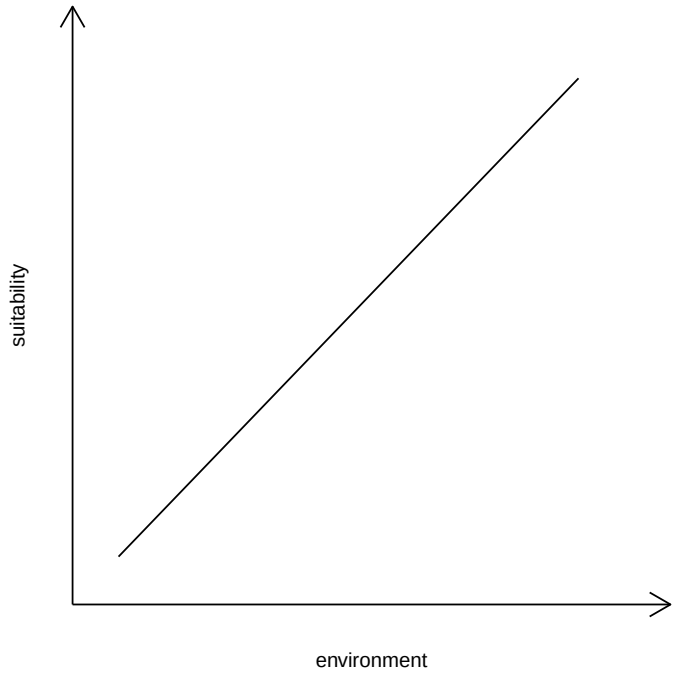
Bell-shaped function

In this method, species habit suitablity are bell-shaped function, see Figure:



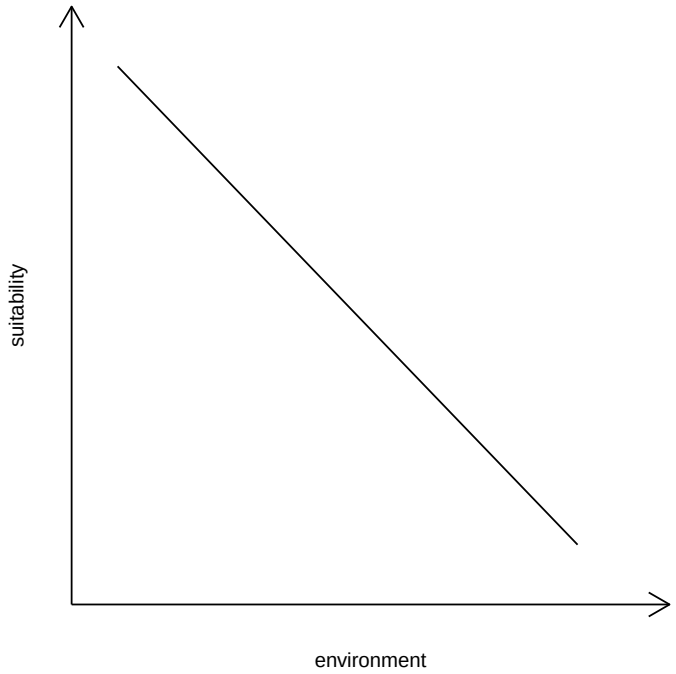
Linear increase function

In this method, species habit suitability are increas with the environment varability.



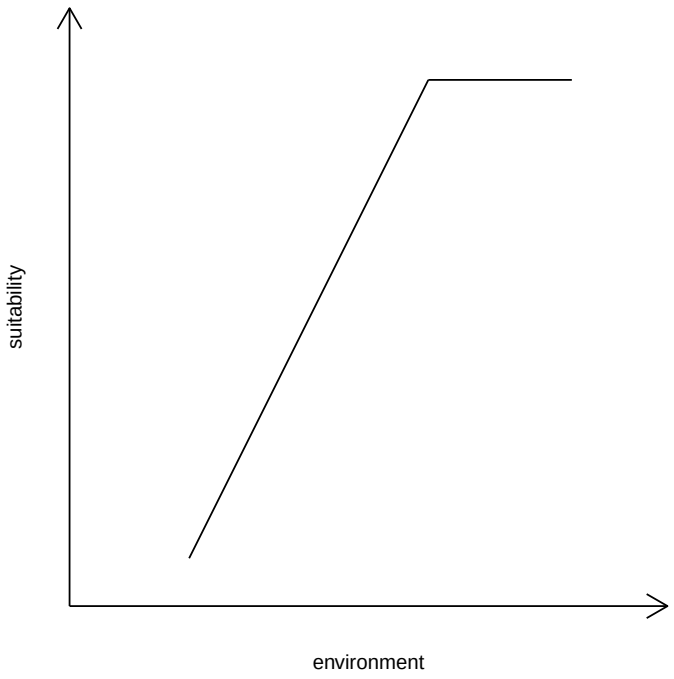
Linear decrease function

In this method, species habit suitability are decrease with the environment

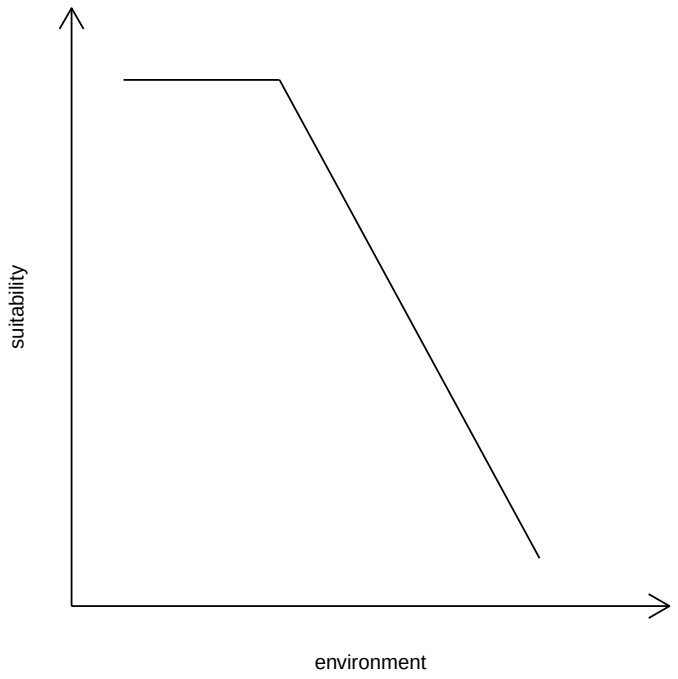


variably.

Truncated linear (increasing) function



Truncated linear (decreasing) function



Write configure code

Response function we retruduce abover, can be use in source with certain code:

- Bell-shaped function: 1
- Linear increase function: 2
- Linear decrease function: 3
- Truncated linear (increasing) function: 4
- Truncated linear (decreasing) function: 5

We see the same code used before, We change the code format for more easy to understand what we do.

```
> config <- list(
+     c("bio1","1",2),
+     c("bio14", "2", 2),
+     c("bio5", "3", 1),
+     c("bio11", "4", 2),
```

```
+     c("bio16", "5", 1)
+     )
```

`config` is a list in R , it contain servel basic element like this: `c("bio11", "4", 2)`, which `"bio11"` is the raster layer name, which always be your env name, indecate which environment should be used, `"4"` indecate what response function we should used (Truncated linear (increasing) function), `2` is the weight.

pick mean method

this method is first development by Jiménez-Valverde and Lobo (2007). Original they first use PCA (Pricipal Component Analysis) to make two main environment to present local environment. In order to extend this method, we don't limit the environment number and the way that variable generate or pick up. This method will consider the points fall in mean±SD are suitable for virutal species occurence in one variable level, only the points fall in all variabes will conside as real occurence.

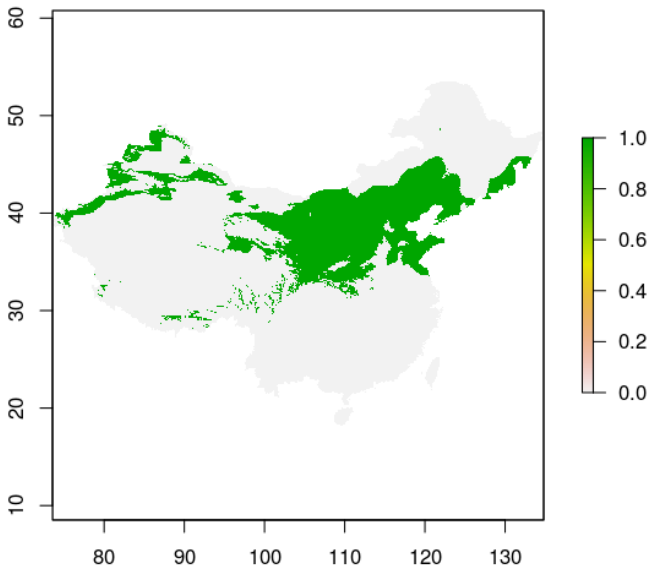
```
> # load the sdmvspecies library
> library("sdmvspecies")
> # find package's location
> package.dir <- system.file(package="sdmvspecies")
> # let see where is our sdmvspecies is installed in
> package.dir

[1] "/usr/local/lib/R/site-library/sdmvspecies"

> # find env dir under the package's location
> env.dir <- paste(package.dir, "/external/env/", sep="")
> # let see env dir
> env.dir

[1] "/usr/local/lib/R/site-library/sdmvspecies/external/env/"

> # get the environment raster file
> file.name <- files <- c("bio1.bil", "bio12.bil", "bio7.bil", "bio5.bil")
> files <- paste(env.dir, file.name, sep="")
> # make raster stack
> env.stack <- stack(files)
> # run pick mean
> species.raster <- pickMean(env.stack)
> # plot map
> plot(species.raster)
```



Pick median method

This method is first development by Lobo and Tognelli (2011).

Original they first use PCA (Pricipal Component Analysis) to choose environment variable from candidate. Here we don't decicus how to choose environment by using PCA or other way choose environment variable. This method will calculate the quartiles of each variables. The points fall in the two central quartiles of variable will conside as suit for species occurence at this variables level. The cells fall in all of variables' central quartiles will conside of species really occurence.

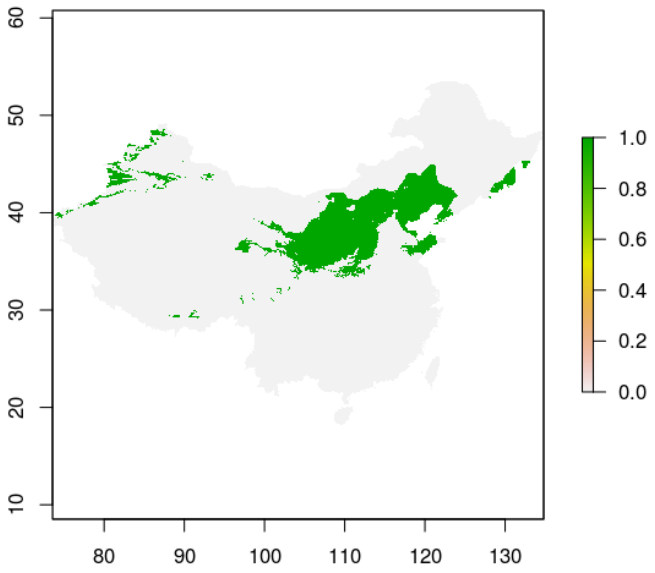
```
> # load the sdmvspecies library
> library("sdmvspecies")
> # load parrllel to use mutilcore CPU
> library("parallel")
> # find package's location
> package.dir <- system.file(package="sdmvspecies")
> # let see where is our sdmvspecies is installed in
> package.dir
```

```
[1] "/usr/local/lib/R/site-library/sdmvspecies"
```

```
> # find env dir under the package's location
> env.dir <- paste(package.dir, "/external/env/", sep="")
> # let see env dir
> env.dir

[1] "/usr/local/lib/R/site-library/sdmvspecies/external/env/"

> # get the environment raster file
> file.name <- files <- c("bio1.bil", "bio12.bil", "bio7.bil", "bio5.bil")
> files <- paste(env.dir, file.name, sep="")
> # make raster stack
> env.stack <- stack(files)
> # run pick mean
> species.raster <- pickMedian(env.stack)
> # plot map
> plot(species.raster)
```



Artificial Bell-shaped response method

This method is first development by Varela et al. (2014). User need to specfic the mean and starnard devation of normal curves (bell-shaped) for each variable.

The species' overall climatic suitability was defined as the multiplication of each variables's suitability. Then User choose a threshold to make distribution map.

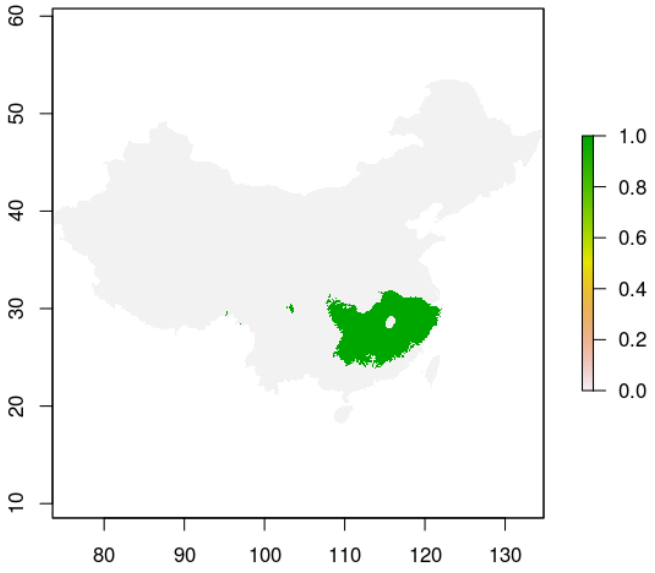
```
> # load the sdmvspecies library
> library("sdmvspecies")
> # load parrllel to use mutilcore CPU
> #library("parallel")
> # find package's location
> package.dir <- system.file(package="sdmvspecies")
> # let see where is our sdmvspecies is installed in
> package.dir

[1] "/usr/local/lib/R/site-library/sdmvspecies"

> # find env dir under the package's location
> env.dir <- paste(package.dir, "/external/env/", sep="")
> # let see env dir
> env.dir

[1] "/usr/local/lib/R/site-library/sdmvspecies/external/env/"

> # get the environment raster file
> file.name <- files <- c("bio1.bil", "bio12.bil", "bio7.bil", "bio5.bil")
> files <- paste(env.dir, file.name, sep="")
> # make raster stack
> env.stack <- stack(files)
> # config
> config <- list(c("bio1",150, 50), c("bio12", 2000, 500), c("bio7", 400, 100), c("bio5", 30
> # run pick mean
> species.raster <- artificialBellResponse(env.stack, config)
> # plot map
> plot(species.raster)
> # species distribution map
> species.distribution.raster <- species.raster > 0.2
> # plot map
> plot(species.distribution.raster)
```


60
50
40
30
20
10
80
90
100
110
120
130
1.0
0.8
0.6
0.4
0.2
0.0

Part I

References

Bibliography

AH Hirzel, V Helfer, and F Metral. Assessing habitat-suitability models with a virtual species. *Ecological modelling*, 145(2):111–121, 2001.

Alberto Jiménez-Valverde and Jorge M Lobo. Threshold criteria for conversion of probability of species presence to either–or presence–absence. *Acta Oecologica*, 31(3):361–369, 2007.

Jorge M Lobo and Marcelo F Tognelli. Exploring the effects of quantity and location of pseudo-absences and sampling biases on the performance of distribution models with limited point occurrence data. *Journal for Nature Conservation*, 19(1):1–7, 2011.

Sara Varela, Robert P. Anderson, Raúl García-Valdés, and Federico Fernández-González. Environmental filters reduce the effects of sampling bias and improve predictions of ecological niche models. *Ecography*, 2014. ISSN 1600-0587. doi: 10.1111/j.1600-0587.2013.00441.x. URL `http://dx.doi.org/10.1111/j.1600-0587.2013.00441.x`.