Readme

##### Github file structure #####

* readme
* images
* input data
  + speciesList.csv
  + speciesObs.csv
  + rasterstack with datalayers of choice, saved as geotiff. Or use the scripts below to create the stack from biooracler.
* results
* scripts
  + readme
    - Download data from GBIF and pseudoabsences
    - Download and formatting of environmental data layers
    - Model test and projection
    - Inventory of potential data layers (add as PDF)
  + script: Download data from GBIF
    - * gbif.download.2025.r

Prepare a request to GBIF

* + - * gbif.download.check.Mars2025.r

Opens download from GBIF and filters data

* + script: Download and formatting of environmental data layers
    - * *access.biooracler.r*
      * *focal.interpolation.r*
      * (Another version of this, using a different set of layers is:

Prepare environmental layers.HAV2022.r)

* + script: model test and projection
    - * invasive.data.masterscript.HAV2025.r

Loops over species to:

extract environmental data

filter observations and prepare tables

prepare CV setup

identify important variables and plot

train Random Forest models, in CV and with full dataset.

plot performance parameters and prepare table

predict and plot occurrence using environmental rasters

Prepares and plots map of average occurrence

Prepares combined map with occurrence and traffic

* + - * SEanalytics.functions2025.r # function library

#####

Download data from GBIF and pseudoabsences

**Data download**

The occurrances are downloaded using the script *gbif.download.2025.r*. The species to download (or possibly another taxonomuc level) are defined in a ,csv file. The link to this file is set as variable file\_url. Taxon keys are subsequently used to download occurrences (occ\_download). A link to the data for download is sent to your email.

Downloaded data files may be stored in files like this. In some cases species data is downloaded at different instances, thus there may be more than one file to process at the later stage

example of filenames:

* GBIF alla arter 0429558-210914110416597.csv
* GBIF download.2.dec 13/0213464-220831081235567.csv

**Data filtration**

Data filtration takes place in the script ”*gbif.download.check.Mars 2025.r*” , first by excluding erroneous coordinates with the function clean\_coordinates() and thereafter by inspecting observations field ”basisOfRecord” and excluding the following classes: "MACHINE\_OBSERVATION", "PRESERVED\_SPECIMEN", "FOSSIL\_SPECIMEN", "MATERIAL\_CITATION" , "MATERIAL\_SAMPLE" , "LIVING\_SPECIMEN"). The following classes were included: "HUMAN\_OBSERVATION", "OCCURRENCE", "OBSERVATION".

For each species the cleaned data are saved as.csv file with the following columns:

"gbifID","occurrenceID","species", "occurrenceStatus", "decimalLongitude", "decimalLatitude", "coordinateUncertaintyInMeters", "depth", "depthAccuracy","eventDate"

For each species a csv file is generated with the following format

write.csv(temp,file = paste(speciespath,s,".csv", sep =""), row.names = F)

In addition to the files, the script creates plots of finds on a world map, partly for each species individually and partly a plot with all species in different colors to give an image of where in the world finds of invasive species have been made.

The last step is to create pseudoabsences for presence-absence modeling. A plot of all downloaded observations (testplot.cleanput.jpg) shows that observations are unevenly distributed globally and that one can assume that there is a greater chance that someone found a species in an area where other species are reported.

From the collected downloaded, cleaned, data a set of *e.g*. 10,000 observations is sampled which are used as pseudoabsences under the assumption that the distribution of all findings in "cleanput" can be considered a reasonable model for "sampling effort".

Of the selected findings, many will, in later stages, turn out to be at points where environmental data is missing.

The script contains a number of options to do this, which need to be selected by editing the script. One may for example define an “excludebox”, an area in which pseudoabsences are not generated. The rationale is that we do not want pseudoabsences in an area where we expect the speces to be absent due to not yet colonizing. There are also options to group species to create different sets of pseudoabsences for e.g. bentic and pelagic species. This is still under development and may generate strange results. (e.g. by using only benthic absences for a bentic species may trick the algoritm to “believe” that the species is pelagic.)

Information about which files are to be used as input in models for different species is gathered in the file "data.table.csv" (or derivates thereof) which has the following format:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **species** | **present.data** | **absence.data** | **pseudoabsence.data** | **n.present** | **n.absent** | **comment on Taxon** |
| Potamopyrgus antipodarum | Potamopyrgus antipodarum.csv | Potamopyrgus antipodarum.csv | pseudoabsences.marine.excludeboxbackground.species.csv | 1620 | 0 | names match |
| Crepidula fornicata | Crepidula fornicata.csv | Crepidula fornicata.csv | pseudoabsences.marine.excludeboxbackground.species.csv | 4007 | 25345 | names match |
| Leptosiphonia brodiei | Leptosiphonia brodiei.csv | Leptosiphonia brodiei.csv | pseudoabsences.marine.excludeboxbackground.species.csv | 208 | 0 | Leptosiphonia fibrillosa |

Note that the downloaded data may use a different taxon name than the one you used. This will be indicated in the column “comment on taxon”. The script that loads data into the model can use different files for positive and negative findings is a inherited function from the previous project. In this project, positive and negative findings are read from the same file and the same pseudoabsences are used for all species, but it is possible, for example, to enter negative findings manually or have different "background models" for different groups of organisms by choosing different pseudoabsence files.

The selection of points in the pseudoabsence model has not been filtered to remove duplicates with the same coordinates before sampling. This means that areas with many duplicates may be overrepresented. Filtering of duplicates was not done until the last run, otherwise it would of course have been done here as well.

The "model" used to handle differences in sampling effort is one of the things that can be further developed in the next project.

Note that the current model does not implement thinning in the preprocessing, which may result in some bias. Implementing a thinning step is on the bucket list.

## Download and formatting of environmental data layers

Environmental layers were downloaded with the script *access.biooracler.r*. Varibles to usedwere specified in the file “dataset och variabler 2025.xlsx".

As the datalayers have an approximate coastline, some pixels will lack a value despite actually containing a small portion of sea. This will in turn make some species observations appear to be on land. To deal with this the script “*focal.interpolation.r*” is used to expand the area with values to the most nearby pixels without values.

After that a rasterstack is built from the interpolated layers. The procedure was repeated tor a number of climate scenarios, creating one stack per scenario.

This step may be omitted if a rasterstack with datalayers already exists.

Model test and projection

The subsequent modelling steps were performed using the script invasive.data.masterscript.with CV.2025.r with help functions library script Seanalytics.functions2025.r

The script is not yet ready to execute in one step. Mark the part of the script that you want to run and execute. It may be a good idea to prepare individual scripts for each of the steps.

The species to run are defined in the file data.table. However, efter data extraction and filtration it may turn out that sone speces has too few observation to build the cross validation scheme (and results would make no sense). Those variables are stored in the variable “exclude”. In theory the subsewuent steps woud then loop over species not listen in “exclude”. However, in reality the script crashes occationally and it has been necessary to manually define which species to excude in the loop, or restart the loop, excluding the species already processed.

The script includes the following steps

* Filepaths, filenames, an suffixes
* Read present, absent and pseudoabsence points and extract environmental data
* Preparing iterations
* Estimate weight of predictors with MCMC algortihm
* Make plots of variables’ weight and correlation with species observations
* Train random forest models
* Extract C50 rules
* Calculate and plot ROC curves from cross validation
* Spatial prediction of species presence and plotting individual maps
* Plot map stacks with average probability
* Plot maps that combine cumulative average probability for all species with traffic layers

*Note that the MCMC step depend on functions that were made available by the author. The varable selection step is not essential for the random forest model (but there is a option to use selected variables). The algorithm is computationally expensive and may be omitted. There are other tree based algoritmns that does a similar job.*

**Filepaths, filenames, and suffixes**

For the most important folders, different variants may be created with different "suffixes" for results with different settings or variables (e.g. climate scenarios)

### Read present, absent and pseudoabsence points and extract environmental data

The filtered data from GBIF is loaded. Duplicates are then filtered out if they have the same coordinates and occurrence status. Next, environmental data is extracted from the raster stack that has the "correct" suffix. Observations without complete environmental data are filtered out. In the loop, a table is also created with statistics for each species on the number of positive and negative findings, including pseudoabsences. Species with fewer than 5 unique, complete, findings are filtered out.

### **Preparing iterations**

The analysis is performed as 5 replicates of a 5x5 cross-validation. For each replicate data are permuted. Then, positive and negative findings are sampled, individually, to belong to one of five possible (other values ​​at the CV level are possible) sets. When an observation is assigned interation i it means that i it will be part of the test set during the i’th iteration of the cross-validation

Since many points are close to each other and would lead to overestimation of predictive power if one allowed nearby points to be included in both the test and training sets, the coordinates may, for example, be rounded as follows:

lonmin <- 10\*floor(my.data$Lon/10)

lonmax <- 10\*ceiling(my.data$Lon/10)

latmin <- 10\*floor(2\*my.data$Lat/10)/2

latmax <- 10\*ceiling(2\*my.data$Lat/10)/2

and each observation is given an observation area whose name is given by the above. When you then sample data for training and test sets, observation areas are sampled, not individual observations. For some species, all findings ended up in a few observation areas.

In order to carry out a 5x cross-validation, there must be findings in at least 5 different areas, otherwise the algorithm crashes. Since a meaningful estimate of predictive power, and also confidence in the maps, is doubtful if the positive findings come from fewer than five areas, such species are excluded.

The size of the rasters and the very principle of grouping findings that may be considered too close to each other for meaningful ROC analysis can be debated.

Information about the iterations is saved in a .rda file and is then used partly in the variable selection algorithm and partly when the Random Forest model is trained for cross-validation.

### **Estimate weight of predictors with MCMC algorithm**

The weight of the variables may be estimated with MCMC feature selection according to the same principle as in earlier model trials by Bergkvist et al (2020). Depending on the number of variables it may, or may not, be possible to buid a Random Forest model with only significant variables, but MCMC could nevertheless be used to get a measure of the usefulness of the variables. The MCMC algorithm is run 25 times with different parts of the dataset to get a feel for how sensitive the weight of the variables (RI index) is to the selection of data. However, as the method is implemented, nearby points may be included in the same dataset and the weight of the variables may be affected by overtraining. This should be handled in the next project, for example by filtering out nearby finds.

The MCMC algorithm is time-consuming and the 25 iterations are carried out on different nodes with the program package parallel{}.

**Make plots of variables’ weight and correlation with species observations**

In part, a plot is made per species that shows the weight of the variables (RI index). A couple of random variables have been added to the model to prevent it from crashing. In the plot, a horizontal line has been inserted corresponding to the RI index that the model considers to be "better than chance". In addition, plots are made for each variable where x is the measured value of the predictor variable divided into 10 equally sized "bins" and the axis is the proportion of positive findings given that x lies in this bin.

**Train random forest models**

A model is trained for each sub-dataset according to the file created under “prepare iterations”. For these models, only the "probability that the observation has the status present" is saved for each observation. In addition, a model is trained with the entire input data that is saved and used for spatial prediction.

In this implementation, iterations of Random Forests are run sequentially. The time gain of running in parallel on different nodes has not been that great, as the number of variables is small, and it has not been worth the time to modify the script for parallel training of models. However, this could be done easily in the future.

There is an option to run the random forest model with selected variables. In this case the model will use those variables that were deemed significant in the same iteration and repeat… This may still be sensitive to overfitting though it there is correlation between points in the training set.

### Extract C50 rules

Test of a simple algorithm to create simple "rules" that describe the relationships between predictor variables and occurrence status. The output of this is saved as a text file for each species. The algorithm can be seen as an attempt to explain what happens inside the "decision trees" even if it is a different decision tree algorithm. The function has been tested without any optimization whatsoever, but it might be interesting to compare with the plots made above and as a "demo" of a track to follow up in future projects.

The C50 rules are not used anywhere else in the script.

### 

**Calculate and plot ROC curves from cross validation**

ROC curves are calculated based on the Random Forest simulations above. The curves are based on the probability predicted for each observation when it constituted the test data, and thus with a model not based on points in this rectangle.

Since the Random Forest algorithm was run in 5 repetitions of a 5x cross-validation, 5 ROC curves with different AUCs were obtained. All five curves are shown as a line in the plot and the average curve as a green polygon. How much difference there is between the AUC values ​​from different repetitions gives an indication of how sensitive the model is to the data.

**Spatial prediction of species presence and plotting individual maps (Sweden and global)**

The Random Forest model is also trained with the entire dataset is used for prediction with a raster stack corresponding to the data used to train the model. The results are first saved as .rda files and, in a later step, as GeoTIFF. There is room here to make the script more uniform and skip the first step.

(it is also possible to make predictions with all 25 models from the cross validation, make a stack of the predicted rasters and then make an average map. This way it is also possible to make maps showing the min and max probability, or the SD of the prediction at each pixel, as a measure of uncertainty-

### **Plot map stacks with average probability**

For each experiment, a raster stack is made with predicted probability from all species for which the model succeeded in creating such a map. An average value is calculated with the mean() function. There are also maps with logarithmic probability and weighted logarithmic probability. This is a legacy from the previous project by Bergkvist et al (2020).

### **Plot maps that combine cumulative average probability for all species with traffic layers**

Raster layers with traffic data from 2016 that were used in the previous project (Bergkvist et al 2020) are read in and then the raster map corresponding to average probability is cropped and resampled to the same extent and resolution as the traffic map. The area being analyzed is given by the extent of the map used by Bergkvist et al (2020).

A composite plot is made with basically the same code as in the previous project. The plot has four panels with

* Average probability for all species given type of model (with or without chlorophyll and without chlorophyll but only points where chlorophyll data is available)
* Traffic data from 2016. The difference to the previous analysis is that the ceiling for traffic intensity was set to 1000 instead of 10000 (values ​​above the ceiling are set to the ceiling). The reason for truncating high values ​​is that otherwise you don't see the traffic lanes in the Gulf of Bothnia, which are much less intensive than the major waterways in the southern Baltic Sea.
* A map where the traffic data is superimposed on the model results with a different color.
* A map where traffic data and model results have been added. During the addition, a scale parameter is used which is set arbitrarily so that the traffic lane will appear.