MAMBO sim framework tests

Simon Rolph

2023-08-18

This is a testbench for setting up a simulation framework for testing integrated models.

Requirements

```
#devtools::install_github("ropensci/NLMR")
#install.packages("virtualspecies")

#remotes::install_github("IIASA/ibis.iSDM") #
#remotes::install_github("IIASA/ibis.iSDM@dev") #development branch

#install.packages("INLA",repos=c(getOption("repos"),INLA="https://inla.r-inla-download.org/R/
stable"), dep=TRUE)
#install.packages("inlabru")



library(NLMR)
library(virtualspecies)
```

```
## Warning: package 'virtualspecies' was built under R version 4.2.3
```

```
## Loading required package: raster
```

```
## Warning: package 'raster' was built under R version 4.2.3
```

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

```
## Warning: package 'sp' was built under R version 4.2.3
```

```
## The legacy packages maptools, rgdal, and rgeos, underpinning the sp package,
## which was just loaded, will retire in October 2023.
## Please refer to R-spatial evolution reports for details, especially
## https://r-spatial.org/r/2023/05/15/evolution4.html.
## It may be desirable to make the sf package available;
## package maintainers should consider adding sf to Suggests:.
## The sp package is now running under evolution status 2
##      (status 2 uses the sf package in place of rgdal)
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.2.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:raster':
##
##     intersect, select, union
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
```

```
##
## Attaching package: 'tidyr'
```

```
## The following object is masked from 'package:raster':
##
##     extract
```

```
library(ibis.iSDM)
library(INLA)
```

```
## Warning: package 'INLA' was built under R version 4.2.3
```

```
## Loading required package: Matrix
```

```
## Warning: package 'Matrix' was built under R version 4.2.3
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loading required package: foreach
```

```
## Loading required package: parallel
## This is INLA_23.04.24 built 2023-04-24 19:15:35 UTC.
##  - See www.r-inla.org/contact-us for how to get help.
```

```
library(inlabru)
```

```
## Warning: package 'inlabru' was built under R version 4.2.3
```

```
library(xgboost)
```

```
## Warning: package 'xgboost' was built under R version 4.2.3
```

```
##
## Attaching package: 'xgboost'
```

```
## The following object is masked from 'package:dplyr':
##
##     slice
```

```
library(terra)
```

```
## Warning: package 'terra' was built under R version 4.2.3
```

```
## terra 1.7.39
```

```
##
## Attaching package: 'terra'
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
library(uuid)
library(assertthat)
```

```
##
## Attaching package: 'assertthat'
```

```
## The following object is masked from 'package:terra':
##
##     noNA
```

```
library(sf)
```

```
## Warning: package 'sf' was built under R version 4.2.3
## Linking to GEOS 3.9.3, GDAL 3.5.2, PROJ 8.2.1; sf_use_s2() is FALSE
```

```
sessionInfo()
```

```
## R version 4.2.2 (2022-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 17763)
## 
## Matrix products: default
## 
## locale:
## [1] LC_COLLATE=English_United Kingdom.1252 
## [2] LC_CTYPE=English_United Kingdom.1252   
## [3] LC_MONETARY=English_United Kingdom.1252
## [4] LC_NUMERIC=C                           
## [5] LC_TIME=English_United Kingdom.1252    
## 
## attached base packages:
## [1] parallel  stats     graphics  grDevices utils     datasets  methods  
## [8] base     
## 
## other attached packages:
##  [1] sf_1.0-14            assertthat_0.2.1     uuid_1.1-0          
##  [4] terra_1.7-39         xgboost_1.7.5.1      inlabru_2.8.0       
##  [7] INLA_23.04.24        foreach_1.5.2        Matrix_1.6-1        
## [10] ibis.iSDM_0.0.8      tidyr_1.3.0          dplyr_1.1.2         
## [13] virtualspecies_1.5.1 raster_3.6-23        sp_2.0-0            
## [16] NLMR_1.1.1          
## 
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.0   xfun_0.40          bslib_0.5.1        purrr_1.0.2       
##  [5] splines_4.2.2      lattice_0.20-45    vctrs_0.6.3        generics_0.1.3    
##  [9] htmltools_0.5.6    yaml_2.3.7         utf8_1.2.3         rlang_1.1.1       
## [13] e1071_1.7-13       jquerylib_0.1.4    pillar_1.9.0       glue_1.6.2        
## [17] DBI_1.1.3          lifecycle_1.0.3    codetools_0.2-18   evaluate_0.21     
## [21] knitr_1.43         fastmap_1.1.1      class_7.3-20       fansi_1.0.4       
## [25] proto_1.0.0        Rcpp_1.0.11        KernSmooth_2.23-20 classInt_0.4-9    
## [29] cachem_1.0.8       jsonlite_1.8.7     digest_0.6.33      grid_4.2.2        
## [33] cli_3.6.1          tools_4.2.2        magrittr_2.0.3     sass_0.4.7        
## [37] proxy_0.4-27       tibble_3.2.1       pkgconfig_2.0.3    data.table_1.14.8 
## [41] rmarkdown_2.24     rstudioapi_0.15.0  iterators_1.0.14   R6_2.5.1          
## [45] units_0.8-3        compiler_4.2.2
```

```
set.seed(42)
```

Environment

Generate a random landscape using NLMR

```
#create x environmental variables variables
env_variables <- terra::rast(
  list(
    env1 = terra::rast(NLMR::nlm_mpd(ncol = 1000,nrow = 1000,roughness = 0.4)),
    env2 = terra::rast(NLMR::nlm_mpd(ncol = 1000,nrow = 1000,roughness = 0.5)),
    env3 = terra::rast(NLMR::nlm_mpd(ncol = 1000,nrow = 1000,roughness = 0.6)),
    env4 = terra::rast(NLMR::nlm_mpd(ncol = 1000,nrow = 1000,roughness = 0.8))
  )
)
```

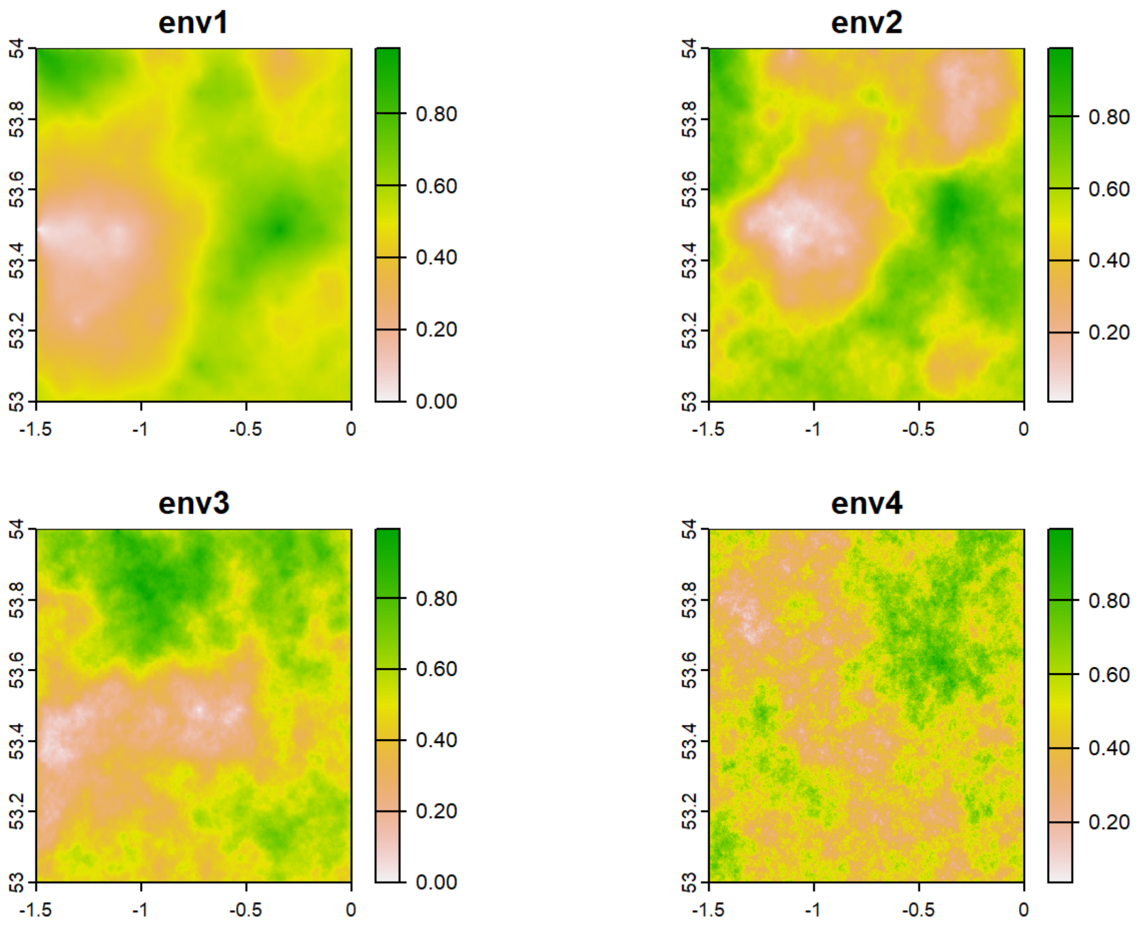
```
## Warning in NLMR::nlm_mpd(ncol = 1000, nrow = 1000, roughness = 0.4): nlm_mpd
## changes the dimensions of the RasterLayer if even ncols/nrows are choosen.
```

```
## Warning in NLMR::nlm_mpd(ncol = 1000, nrow = 1000, roughness = 0.5): nlm_mpd
## changes the dimensions of the RasterLayer if even ncols/nrows are choosen.
```

```
## Warning in NLMR::nlm_mpd(ncol = 1000, nrow = 1000, roughness = 0.6): nlm_mpd
## changes the dimensions of the RasterLayer if even ncols/nrows are choosen.
```

```
## Warning in NLMR::nlm_mpd(ncol = 1000, nrow = 1000, roughness = 0.8): nlm_mpd
## changes the dimensions of the RasterLayer if even ncols/nrows are choosen.
```

```
# create target raster with the correct extent/crs etc.
s <- rast(nrows=nrow(env_variables), ncols=ncol(env_variables), xmin=-1.5, xmax=0, ymin=53, y
max=54,nlyrs=length(names(env_variables)),names = names(env_variables),crs = "EPSG:4326")

#put the values in the new raster
values(s) <- values(env_variables)
env_variables <- s

#visualise
plot(env_variables)
```



Species

Generate some virtual species using virtual species

```
#generate species using virtual species package and the simualted landscapes
virtual_species1 <- virtualspecies::generateRandomSp(raster::raster(env_variables),species.pr
evalence = 0.4)
```

```
##  - Determining species' response to predictor variables
```

```
##  - Calculating species suitability
```

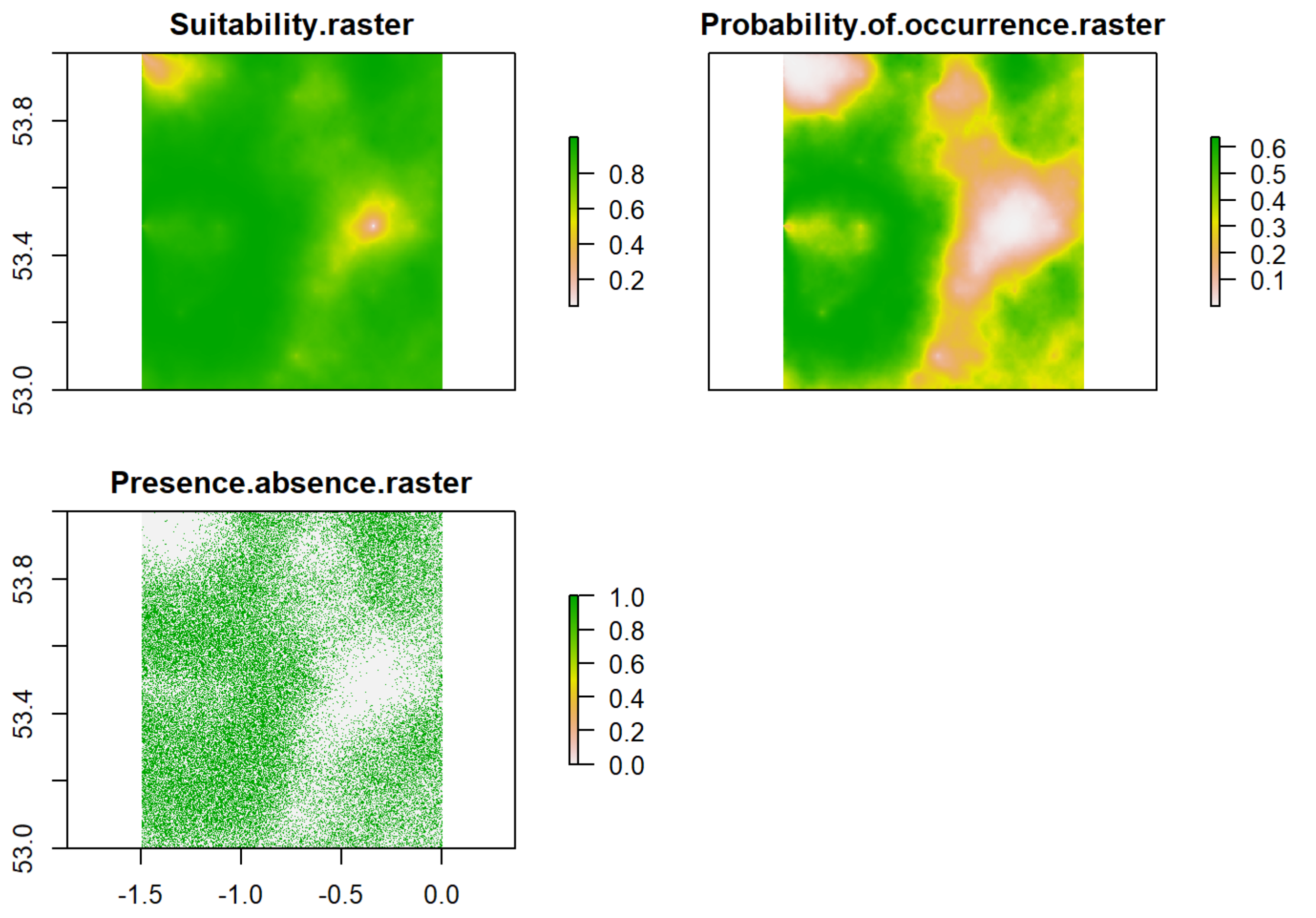
```
## Generating virtual species environmental suitability...
```

```
##  - The response to each variable was rescaled between 0 and 1. To
##             disable, set argument rescale.each.response = FALSE
```

```
##  - The final environmental suitability was rescaled between 0 and 1.
##             To disable, set argument rescale = FALSE
```

```
##  - Converting into Presence - Absence
```

```
##   --- Determing beta automatically according to alpha and species.prevalence
##    Logistic conversion finished:
##
## - beta = 0.9443359375
## - alpha = -0.1
## - species prevalence =0.3986488991494
```



```
virtual_species2 <- virtualspecies::generateRandomSp(raster::raster(env_variables),species.pr
evalence = 0.1)
```

```
##  - Determining species' response to predictor variables
```

```
##  - Calculating species suitability
```

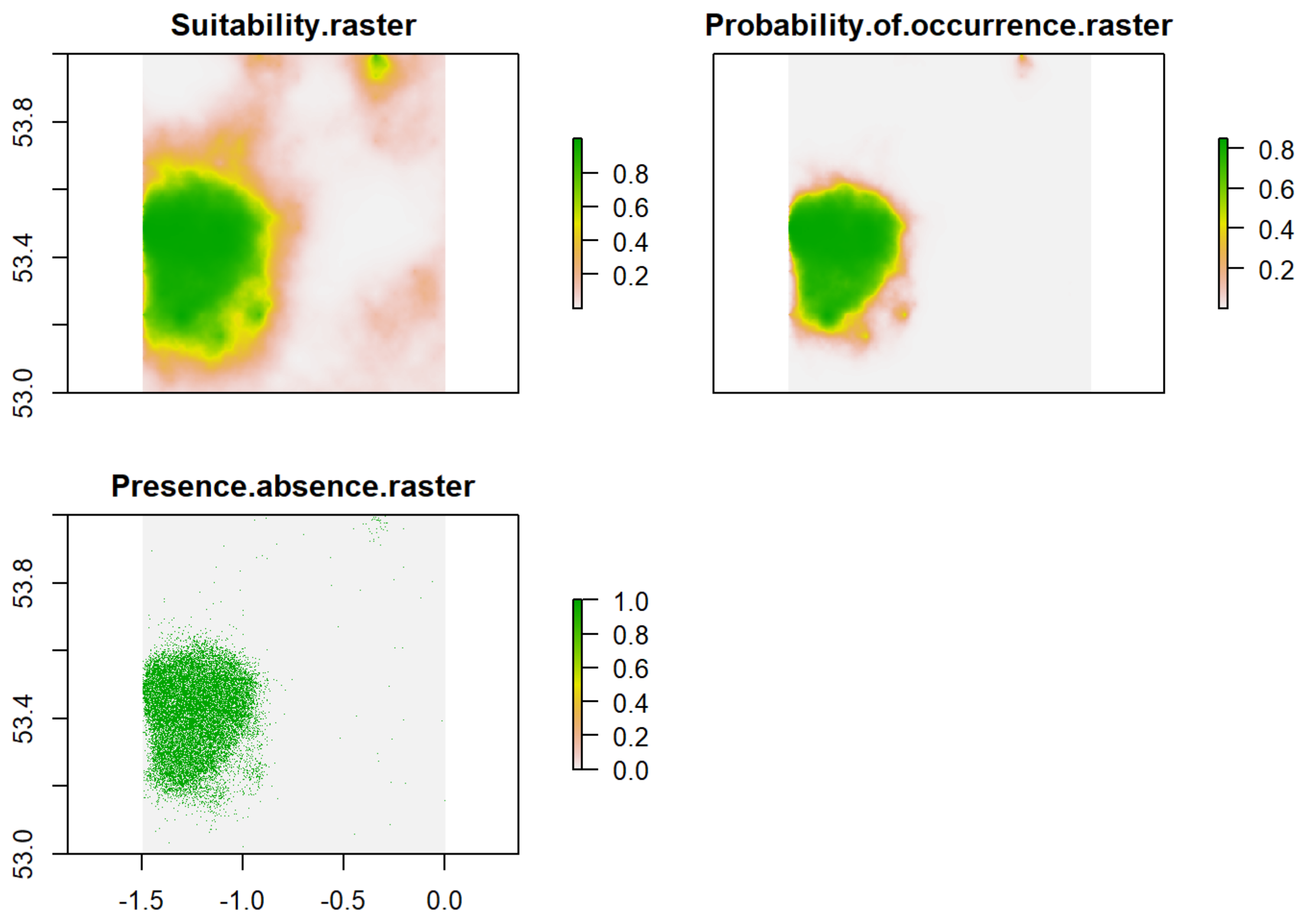
```
## Generating virtual species environmental suitability...
```

```
##  - The response to each variable was rescaled between 0 and 1. To
##             disable, set argument rescale.each.response = FALSE
```

```
##  - The final environmental suitability was rescaled between 0 and 1.
##             To disable, set argument rescale = FALSE
```

```
##  - Converting into Presence - Absence
##    --- Determing beta automatically according to alpha and species.prevalence
```

```
##    Logistic conversion finished:
##
## - beta = 0.828125
## - alpha = -0.1
## - species prevalence =0.101547994440887
```



```
virtual_species3 <- virtualspecies::generateRandomSp(raster::raster(env_variables),species.pr
evalence = 0.5)
```

```
##   - Determining species' response to predictor variables
```

```
##   - Calculating species suitability
```

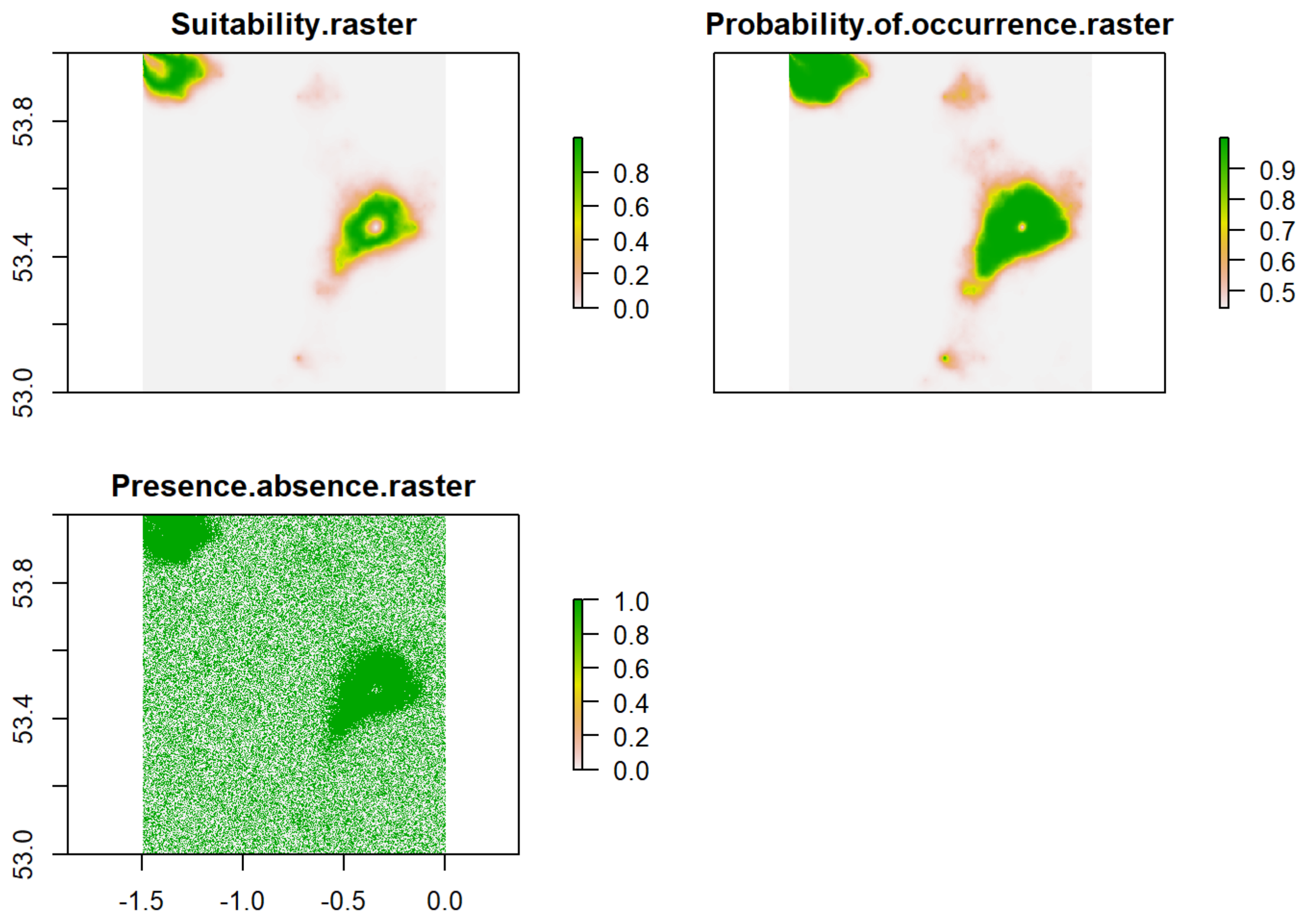
```
## Generating virtual species environmental suitability...
```

```
##   - The response to each variable was rescaled between 0 and 1. To
##             disable, set argument rescale.each.response = FALSE
```

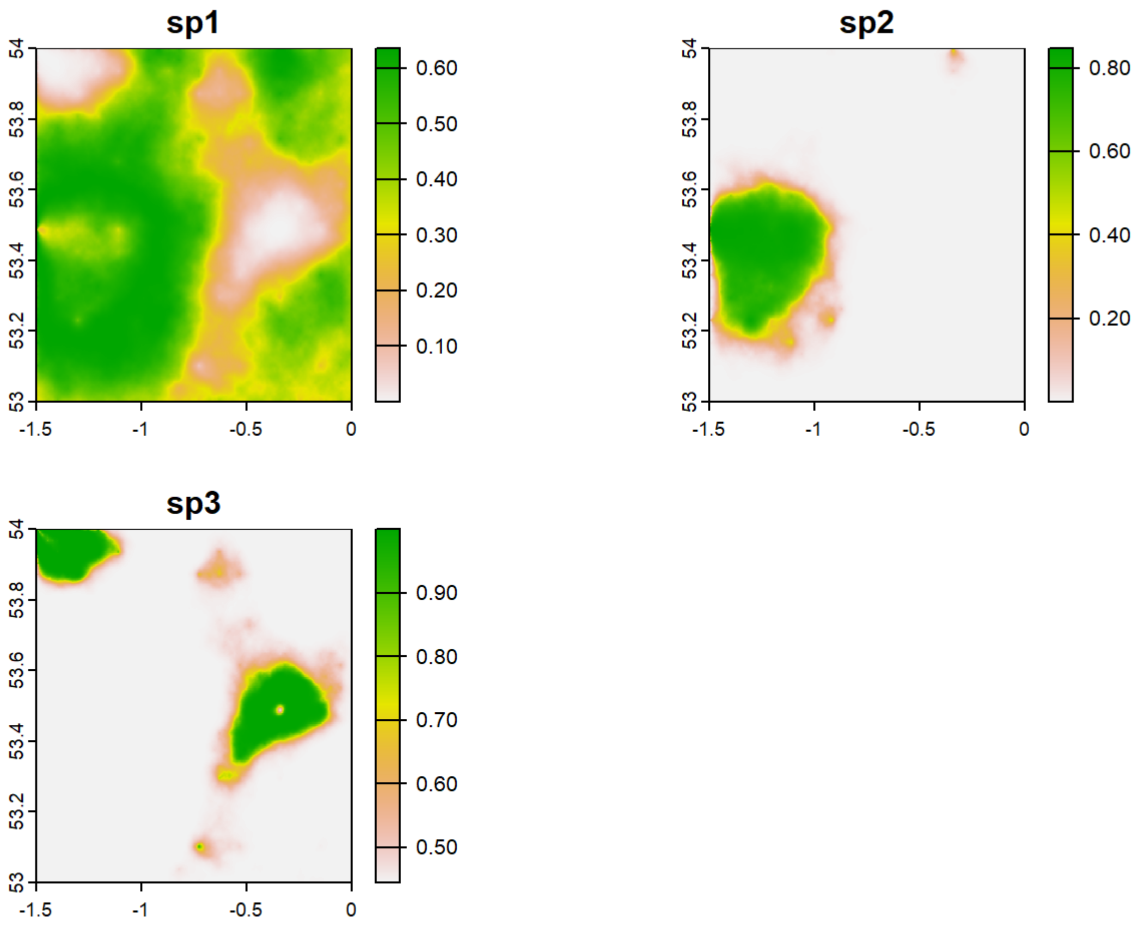
```
##   - The final environmental suitability was rescaled between 0 and 1.
##             To disable, set argument rescale = FALSE
##  - Converting into Presence - Absence
```

```
##    --- Determing beta automatically according to alpha and species.prevalence
```

```
##    Logistic conversion finished:
##
## - beta = 0.02197265625
## - alpha = -0.1
## - species prevalence =0.499692886079272
```



```
#compile into one raster
species_occ_true <- rast(
  list(
    sp1 = rast(virtual_species1$probability.of.occurrence),
    sp2 = rast(virtual_species2$probability.of.occurrence),
    sp3 = rast(virtual_species3$probability.of.occurrence))
)

#visualise
plot(species_occ_true)
```



Sample

Sample the data with three different ways

- Presence-only citizen science data
- Presence-absence timed counts
- Automated trap data represented as presence-absence with replicates (eg. multiple sampling events but in the same location)

```
get_samples <- function(env_var,sp_occ,n_deployments,n_samples){
  deployment_locations <- spatSample(env_var,
                                     size = n_deployments,
                                     as.points=T,
                                     xy=T,
                                     replace=T)

  #sample the species
  deployments <- terra::extract(sp_occ,deployment_locations,xy=T)

  #go long data frame
  deployments <- pivot_longer(deployments,cols = names(sp_occ),names_to="species",values_to
="sp_occ")

  names(deployments)[1] <- "deployment_id"

  #repeat for n_samples
  samples <- do.call("rbind", replicate(n_samples, deployments, simplify = FALSE))

  #turn to T/F detection
  samples$detect_test <- runif(nrow(samples))
  samples$detected <- samples$detect_test<samples$sp_occ

  #add a sample ID
  samples$sample_id <- 1:nrow(samples)

  samples %>% arrange(deployment_id)
}

#unstructured sampling
cit_sci_samples <- get_samples(env_variables,species_occ_true,1000,1)
cit_sci_samples
```

```
## # A tibble: 3,000 x 8
##    deployment_id        x     y species   sp_occ detect_test detected sample_id
##            <dbl>    <dbl> <dbl> <chr>      <dbl>       <dbl> <lgl>        <int>
##  1             1 -1.19     53.7 sp1     0.625         0.793  FALSE            1
##  2             1 -1.19     53.7 sp2     0.0484        0.297  FALSE            2
##  3             1 -1.19     53.7 sp3     0.445         0.346  TRUE             3
##  4             2 -0.724    53.1 sp1     0.0895        0.364  FALSE            4
##  5             2 -0.724    53.1 sp2     0.000265      0.481  FALSE            5
##  6             2 -0.724    53.1 sp3     0.825         0.979  FALSE            6
##  7             3 -0.789    53.8 sp1     0.356         0.495  FALSE            7
##  8             3 -0.789    53.8 sp2     0.000391      0.491  FALSE            8
##  9             3 -0.789    53.8 sp3     0.446         0.0276 TRUE             9
## 10             4 -0.00826  53.3 sp1     0.411         0.757  FALSE           10
## # i 2,990 more rows
```

```
#timed count sampling
timed_count_samples <- get_samples(env_variables,species_occ_true,100,1)
timed_count_samples
```

```
## # A tibble: 300 x 8
##    deployment_id       x     y species   sp_occ detect_test detected sample_id
##            <dbl>   <dbl> <dbl> <chr>      <dbl>       <dbl> <lgl>        <int>
##  1             1 -0.0878  53.2 sp1     0.355          0.864 FALSE            1
##  2             1 -0.0878  53.2 sp2     0.000390       0.268 FALSE            2
##  3             1 -0.0878  53.2 sp3     0.446          0.552 FALSE            3
##  4             2 -1.07    53.0 sp1     0.412          0.528 FALSE            4
##  5             2 -1.07    53.0 sp2     0.000488       0.293 FALSE            5
##  6             2 -1.07    53.0 sp3     0.445          0.761 FALSE            6
##  7             3 -0.995   53.1 sp1     0.599          0.103 TRUE             7
##  8             3 -0.995   53.1 sp2     0.00979        0.991 FALSE            8
##  9             3 -0.995   53.1 sp3     0.445          0.640 FALSE            9
## 10             4 -0.253   53.3 sp1     0.441          0.485 FALSE           10
## # i 290 more rows
```

```
#AMI trap
ami_trap_samples <- get_samples(env_variables,species_occ_true,5,50)
ami_trap_samples
```

```
## # A tibble: 750 x 8
##    deployment_id      x     y species   sp_occ detect_test detected sample_id
##            <dbl>  <dbl> <dbl> <chr>      <dbl>       <dbl> <lgl>        <int>
##  1             1 -0.687  53.5 sp1     0.449          0.514 FALSE            1
##  2             1 -0.687  53.5 sp2     0.000606       0.851 FALSE            2
##  3             1 -0.687  53.5 sp3     0.445          0.667 FALSE            3
##  4             1 -0.687  53.5 sp1     0.449          0.812 FALSE           16
##  5             1 -0.687  53.5 sp2     0.000606       0.836 FALSE           17
##  6             1 -0.687  53.5 sp3     0.445          0.626 FALSE           18
##  7             1 -0.687  53.5 sp1     0.449          0.629 FALSE           31
##  8             1 -0.687  53.5 sp2     0.000606       0.689 FALSE           32
##  9             1 -0.687  53.5 sp3     0.445          0.791 FALSE           33
## 10             1 -0.687  53.5 sp1     0.449          0.718 FALSE           46
## # i 740 more rows
```

Modelling

Modelling using ibis.iSDM with the inlabru engine

```
#create a background raster layer that just has value 1 for all cells
background <- setValues(env_variables,1)[[1]]
```

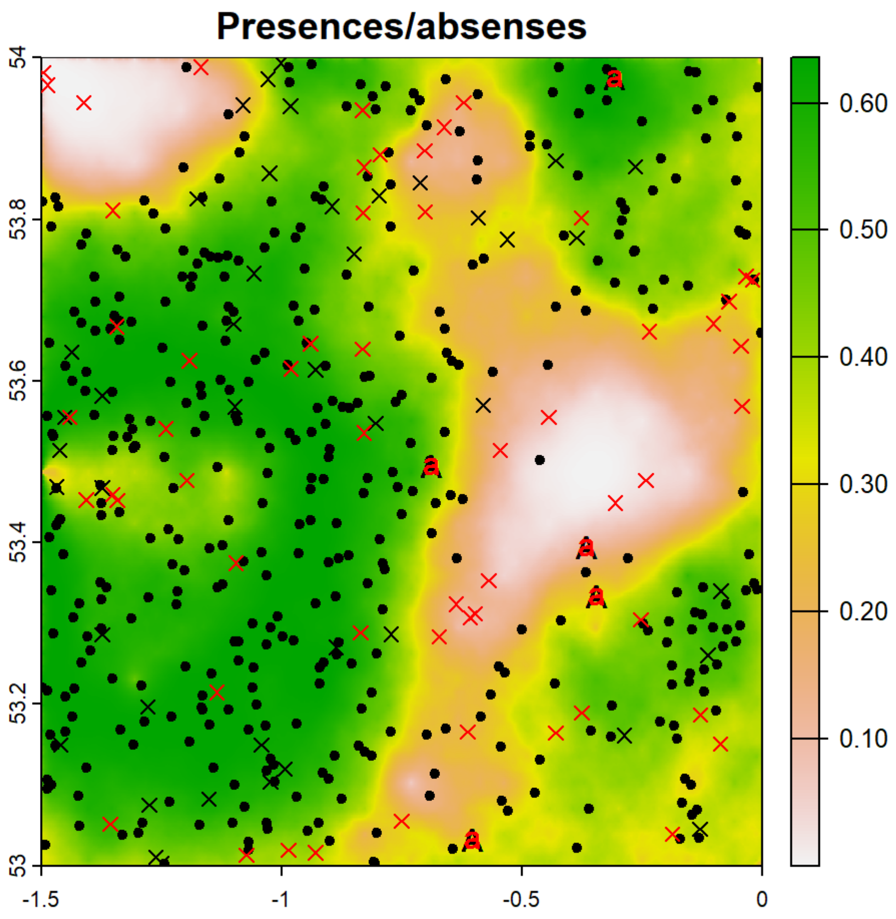
Reformat the data

Extract the right data and add crs etc.

In the plot:

- Back solid circles - citizen science PO data
- Black ‘X’ - presence timed counts
- Black ‘A’ - presence automated trap
- Red ‘X’ - absence from timed count
- Red ‘a’ - absence from automated trap

```
sp1_po <- cit_sci_samples %>%
  filter(detected==T,
         species == "sp1") %>%
  st_as_sf(coords =c("x","y"),crs = 4326)%>%
  mutate(detected = as.numeric(detected))

sp1_pa_ami <- ami_trap_samples %>%
  filter(species == "sp1") %>%
  st_as_sf(coords =c("x","y"),crs = 4326)%>%
  mutate(detected = as.numeric(detected))

sp1_pa_timed <- timed_count_samples %>%
  filter(species == "sp1") %>%
  st_as_sf(coords =c("x","y"),crs = 4326)%>%
  mutate(detected = as.numeric(detected))

#plot presences
plot(terra::rast(virtual_species1$probability.of.occurrence),main = "Presences/absenses")
plot(sp1_po$geometry,add=T,pch=20)
sp1_pa_ami %>% filter(detected==1) %>% pull(geometry) %>% plot(add=T,pch="A")
sp1_pa_timed %>% filter(detected==1) %>% pull(geometry) %>% plot(add=T,pch=4)
sp1_pa_ami %>% filter(detected==0) %>% pull(geometry) %>% plot(add=T,pch="a",col="red")
sp1_pa_timed %>% filter(detected==0) %>% pull(geometry) %>% plot(add=T,pch=4,col="red")
```



```
sp1_map <- recordPlot()

head(sp1_po)
```

```
## Simple feature collection with 6 features and 6 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -1.47973 ymin: 53.0045 xmax: -0.8085586 ymax: 53.79129
## Geodetic CRS:  WGS 84
## # A tibble: 6 x 7
##   deployment_id species sp_occ detect_test detected sample_id
##           <dbl> <chr>    <dbl>       <dbl>    <dbl>     <int>
## 1             8 sp1      0.604       0.419        1        22
## 2             9 sp1      0.585       0.535        1        25
## 3            10 sp1      0.632       0.570        1        28
## 4            12 sp1      0.387       0.195        1        34
## 5            14 sp1      0.636       0.287        1        40
## 6            19 sp1      0.303       0.116        1        55
## # i 1 more variable: geometry <POINT [°]>
```

```
head(sp1_pa_ami)
```

```
## Simple feature collection with 6 features and 6 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -0.6869369 ymin: 53.496 xmax: -0.6869369 ymax: 53.496
## Geodetic CRS:  WGS 84
## # A tibble: 6 x 7
##   deployment_id species sp_occ detect_test detected sample_id
##           <dbl> <chr>    <dbl>       <dbl>    <dbl>     <int>
## 1             1 sp1      0.449       0.514        0         1
## 2             1 sp1      0.449       0.812        0        16
## 3             1 sp1      0.449       0.629        0        31
## 4             1 sp1      0.449       0.718        0        46
## 5             1 sp1      0.449       0.903        0        61
## 6             1 sp1      0.449       0.362        1        76
## # i 1 more variable: geometry <POINT [°]>
```

```
head(sp1_pa_timed)
```

```
## Simple feature collection with 6 features and 6 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -1.263514 ymin: 53.01051 xmax: -0.08783784 ymax: 53.88539
## Geodetic CRS:  WGS 84
## # A tibble: 6 x 7
##   deployment_id species sp_occ detect_test detected sample_id
##           <dbl> <chr>    <dbl>       <dbl>    <dbl>     <int>
## 1             1 sp1      0.355       0.864        0         1
## 2             2 sp1      0.412       0.528        0         4
## 3             3 sp1      0.599       0.103        1         7
## 4             4 sp1      0.441       0.485        0        10
## 5             5 sp1      0.440       0.364        1        13
## 6             6 sp1      0.148       0.729        0        16
## # i 1 more variable: geometry <POINT [°]>
```

Fit models (with no integration)

```
#psabs_settings <- pseudoabs_settings(background=background,nrpoints=nrow(sp1_po))#not used
mod_po <- distribution(background) %>%
  add_predictors(env = env_variables,
                 transform = "scale",
                 derivates = "none") %>%
  add_biodiversity_poipo(sp1_po,
                         name = "Citizen sci po data",
                         field_occurrence = "detected") %>%
  engine_inlabru() %>%
  train(runname =  "sp1_po",
        verbose = T,
        aggregate_observations = F)
```

```
## [32m[Setup] 2023-08-18 16:19:24 | Creating distribution object...[39m
```

```
## [32m[Setup] 2023-08-18 16:19:24 | Adding predictors...[39m
```

```
## [32m[Setup] 2023-08-18 16:19:24 | Transforming predictors...[39m
```

```
## [32m[Setup] 2023-08-18 16:19:25 | Adding poipo dataset...[39m
```

```
## [32m[Estimation] 2023-08-18 16:19:25 | Collecting input parameters.[39m
```

```
## [32m[Estimation] 2023-08-18 16:19:26 | Adding engine-specific parameters.[39m
```

```
## [32m[Estimation] 2023-08-18 16:19:30 | Engine setup.[39m
```

```
## [32m[Estimation] 2023-08-18 16:19:30 | Starting fitting.[39m
```

```
## iinla: Iteration 1 [max:10]
```

```
## iinla: Iteration 2 [max:10]
```

```
## iinla: Max deviation from previous: 0.00118% of SD, and line search is inactive [stop if:
<10% and line search inactive]
```

```
## iinla: Convergence criterion met, running final INLA integration with known theta mode.
```

```
## iinla: Iteration 3 [max:10]
```

```
## [32m[Estimation] 2023-08-18 16:19:57 | Starting prediction.[39m
```

```
## [32m[Done] 2023-08-18 16:20:21 | Completed after 55.8 secs[39m
```

```
mod_pa_ami <- distribution(background) %>%
  add_predictors(env = env_variables,
                 transform = "scale",
                 derivates = "none") %>%
  add_biodiversity_poipa(sp1_pa_ami,
                         name = "AMI trap data",
                         field_occurrence = "detected") %>%
  engine_inlabru() %>%
  train(runname =  "sp1_pa_ami",
        verbose = T,
        aggregate_observations = F)
```

```
## [32m[Setup] 2023-08-18 16:20:21 | Creating distribution object...[39m
```

```
## [32m[Setup] 2023-08-18 16:20:21 | Adding predictors...[39m
```

```
## [32m[Setup] 2023-08-18 16:20:21 | Transforming predictors...[39m
```

```
## [32m[Setup] 2023-08-18 16:20:22 | Adding poipa dataset...[39m
```

```
## [32m[Estimation] 2023-08-18 16:20:22 | Collecting input parameters.[39m
```

```
## [32m[Estimation] 2023-08-18 16:20:23 | Adding engine-specific parameters.[39m
```

```
## [32m[Estimation] 2023-08-18 16:20:28 | Engine setup.[39m
```

```
## [32m[Estimation] 2023-08-18 16:20:28 | Starting fitting.[39m
```

```
## iinla: Iteration 1 [max:10]
```

```
## iinla: Iteration 2 [max:10]
```

```
## iinla: Max deviation from previous: 2.89e-08% of SD, and line search is inactive [stop if:
<10% and line search inactive]
```

```
## iinla: Convergence criterion met, running final INLA integration with known theta mode.
```

```
## iinla: Iteration 3 [max:10]
```

```
## [32m[Estimation] 2023-08-18 16:20:55 | Starting prediction.[39m
```

```
## [32m[Done] 2023-08-18 16:21:10 | Completed after 48.71 secs[39m
```

```
mod_pa_timed <- distribution(background) %>%
  add_predictors(env = env_variables,
                 transform = "scale",
                 derivates = "none") %>%
  add_biodiversity_poipa(sp1_pa_timed,
                         name = "Timed count data",
                         field_occurrence = "detected") %>%
  engine_inlabru() %>%
  train(runname =  "sp1_pa_timed",
        verbose = T,
        aggregate_observations = F)
```

```
## [32m[Setup] 2023-08-18 16:21:11 | Creating distribution object...[39m
```

```
## [32m[Setup] 2023-08-18 16:21:11 | Adding predictors...[39m
```

```
## [32m[Setup] 2023-08-18 16:21:11 | Transforming predictors...[39m
```

```
## [32m[Setup] 2023-08-18 16:21:11 | Adding poipa dataset...[39m
```

```
## [32m[Estimation] 2023-08-18 16:21:11 | Collecting input parameters.[39m
```

```
## [32m[Estimation] 2023-08-18 16:21:12 | Adding engine-specific parameters.[39m
```

```
## [32m[Estimation] 2023-08-18 16:21:17 | Engine setup.[39m
```

```
## [32m[Estimation] 2023-08-18 16:21:17 | Starting fitting.[39m
```

```
## iinla: Iteration 1 [max:10]
```

```
## iinla: Iteration 2 [max:10]
```

```
## iinla: Max deviation from previous: 3.14e-10% of SD, and line search is inactive [stop if:
<10% and line search inactive]
```

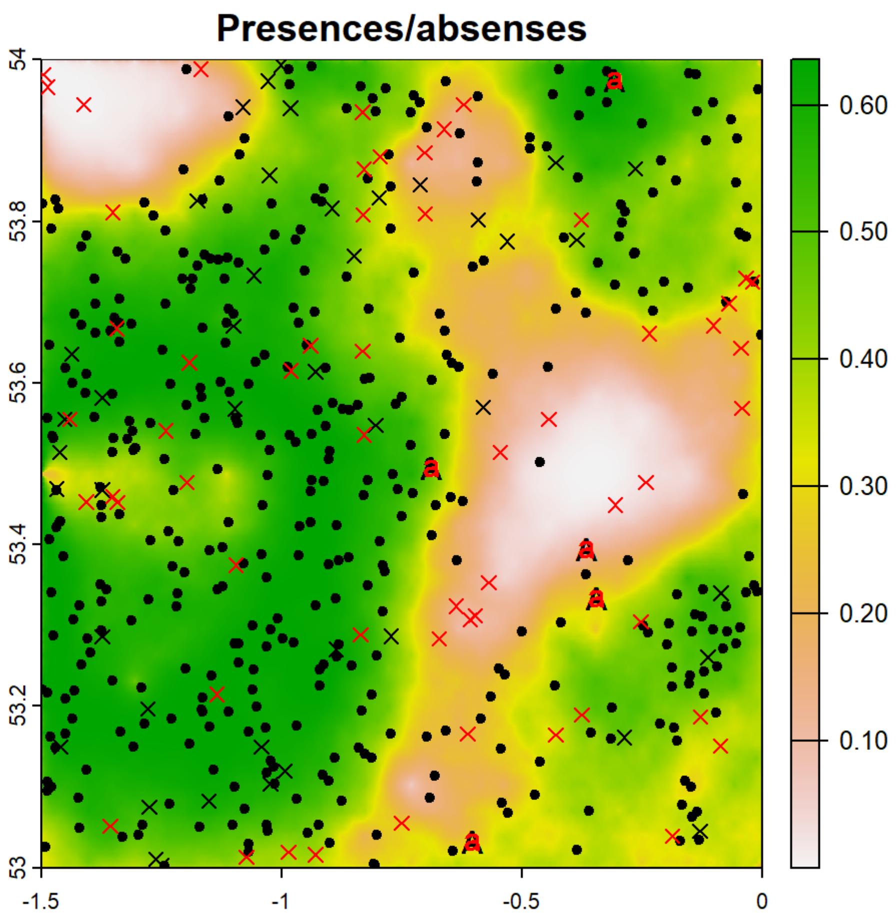
```
## iinla: Convergence criterion met, running final INLA integration with known theta mode.
```

```
## iinla: Iteration 3 [max:10]
```

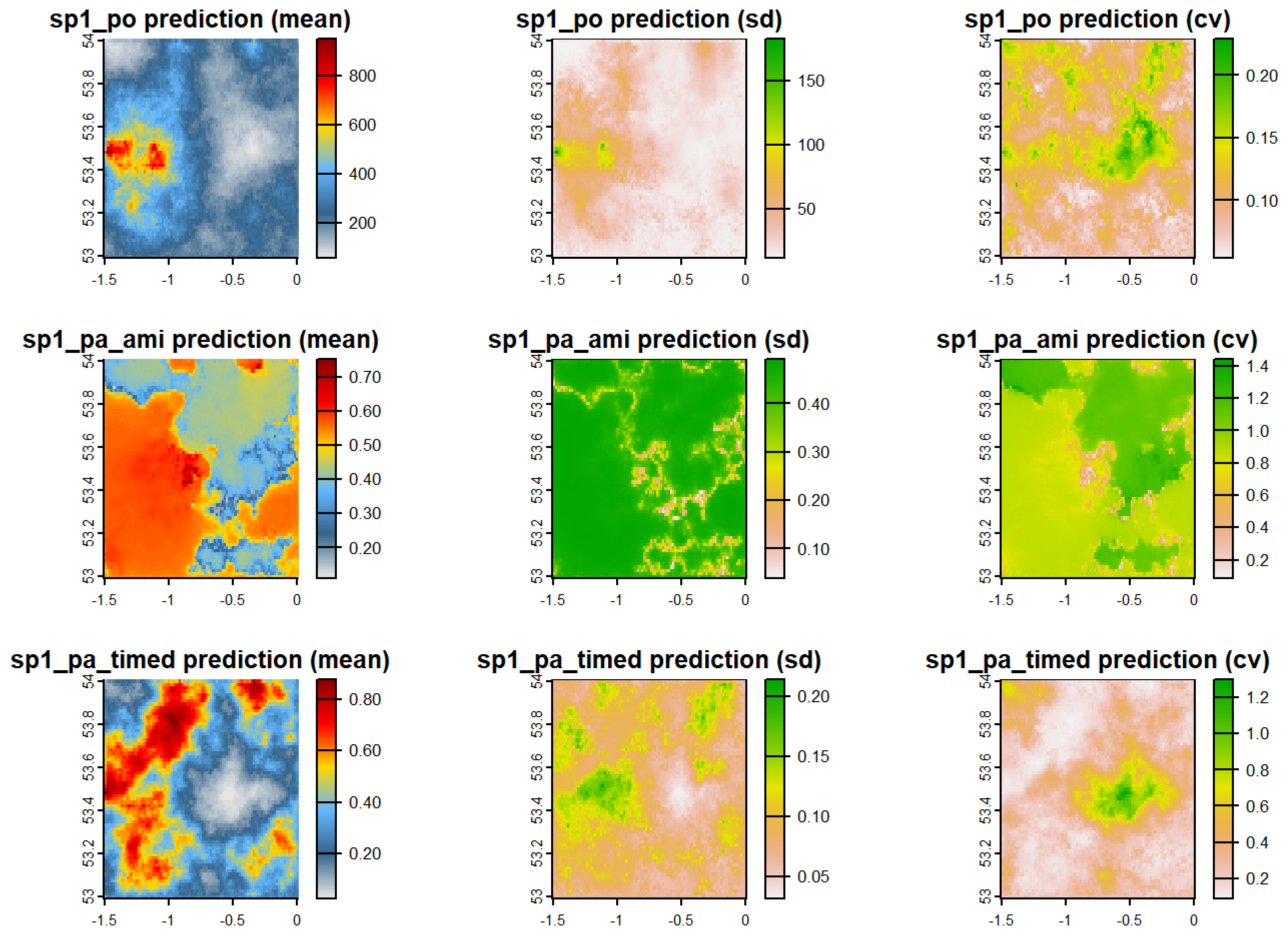
```
## [32m[Estimation] 2023-08-18 16:21:44 | Starting prediction.[39m
```

```
## [32m[Done] 2023-08-18 16:21:57 | Completed after 45.69 secs[39m
```

```
#species distribution and samples
sp1_map
```



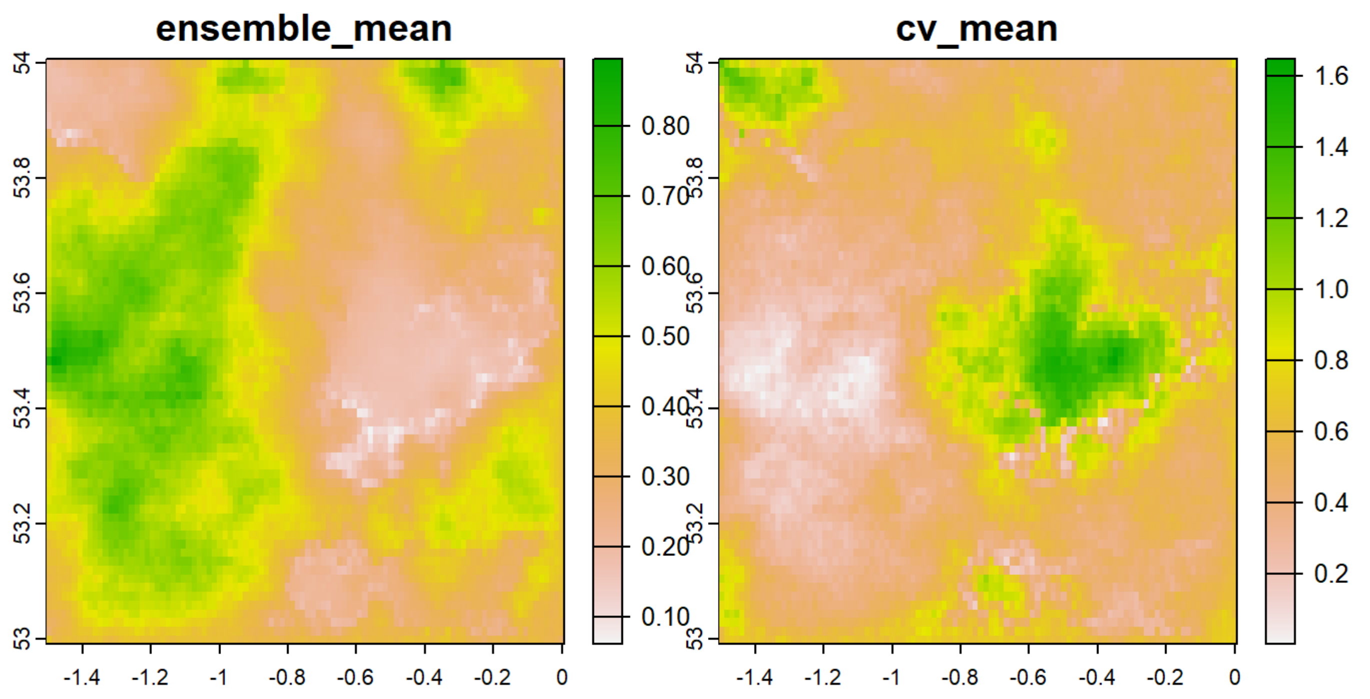
```
# mean and variance of each model
par(mfrow = c(3, 3))
plot(mod_po)
plot(mod_po$fits$prediction$sd,main ="sp1_po prediction (sd)")
plot(mod_po$fits$prediction$cv,main ="sp1_po prediction (cv)")

plot(mod_pa_ami)
plot(mod_pa_ami$fits$prediction$sd,main ="sp1_pa_ami prediction (sd)")
plot(mod_pa_ami$fits$prediction$cv,main ="sp1_pa_ami prediction (cv)")

plot(mod_pa_timed)
plot(mod_pa_timed$fits$prediction$sd,main ="sp1_pa_timed prediction (sd)")
plot(mod_pa_timed$fits$prediction$cv,main ="sp1_pa_timed prediction (cv)")
```



```
par(mfrow = c(1, 1))
```

Make an ensemble model

```
e <- ensemble(mod_po, mod_pa_ami,mod_pa_timed, method = "mean", normalize = TRUE)
plot(e)
```



Fit integrated models

Combinations of 2 data types + all 3 data types

(these all seem to be much worse…)

```
#po+pa_timed
mod_po_pa_timed <- distribution(background) %>%
  add_predictors(env = env_variables,
                 transform = "scale",
                 derivates = "none") %>%
  add_biodiversity_poipo(sp1_po,
                         name = "Citizen sci po data",
                         field_occurrence = "detected") %>%
  add_biodiversity_poipa(sp1_pa_timed,
                           name = "Timed count data",
                           field_occurrence = "detected") %>%
  engine_inlabru() %>%
  train(runname =  "sp1_po_pa_timed",
        verbose = T,
        aggregate_observations = F)
```

```
## [32m[Setup] 2023-08-18 16:22:00 | Creating distribution object...[39m
```

```
## [32m[Setup] 2023-08-18 16:22:00 | Adding predictors...[39m
```

```
## [32m[Setup] 2023-08-18 16:22:00 | Transforming predictors...[39m
## [32m[Setup] 2023-08-18 16:22:01 | Adding poipo dataset...[39m
```

```
## [32m[Setup] 2023-08-18 16:22:01 | Adding poipa dataset...[39m
```

```
## [32m[Estimation] 2023-08-18 16:22:01 | Collecting input parameters.[39m
```

```
## [32m[Estimation] 2023-08-18 16:22:02 | Adding engine-specific parameters.[39m
```

```
## [32m[Estimation] 2023-08-18 16:22:06 | Engine setup.[39m
```

```
## [32m[Estimation] 2023-08-18 16:22:07 | Starting fitting.[39m
```

```
## iinla: Iteration 1 [max:10]
```

```
## iinla: Iteration 2 [max:10]
```

```
## iinla: Max deviation from previous: 0.000603% of SD, and line search is inactive [stop if:
<10% and line search inactive]
```

```
## iinla: Convergence criterion met, running final INLA integration with known theta mode.
```

```
## iinla: Iteration 3 [max:10]
```

```
## [32m[Estimation] 2023-08-18 16:22:43 | Starting prediction.[39m
```

```
## [32m[Done] 2023-08-18 16:23:03 | Completed after 1.03 mins[39m
```

```
#po+pa_ami
mod_po_pa_ami <- distribution(background) %>%
  add_predictors(env = env_variables,
                 transform = "scale",
                 derivates = "none") %>%
  add_biodiversity_poipo(sp1_po,
                       name = "Citizen sci po data",
                       field_occurrence = "detected") %>%
  add_biodiversity_poipa(sp1_pa_ami,
                         name = "AMI trap pa data",
                         field_occurrence = "detected") %>%
  add_biodiversity_poipa(sp1_pa_timed,
                         name = "Timed count data",
                         field_occurrence = "detected") %>%

  engine_inlabru() %>%
  train(runname =  "sp1_po_pa_ami",
        verbose = T,
        aggregate_observations = F)
```

```
## [32m[Setup] 2023-08-18 16:23:03 | Creating distribution object...[39m
```

```
## [32m[Setup] 2023-08-18 16:23:03 | Adding predictors...[39m
```

```
## [32m[Setup] 2023-08-18 16:23:03 | Transforming predictors...[39m
```

```
## [32m[Setup] 2023-08-18 16:23:04 | Adding poipo dataset...[39m
```

```
## [32m[Setup] 2023-08-18 16:23:04 | Adding poipa dataset...[39m
## [32m[Setup] 2023-08-18 16:23:04 | Adding poipa dataset...[39m
```

```
## [32m[Estimation] 2023-08-18 16:23:04 | Collecting input parameters.[39m
```

```
## [32m[Estimation] 2023-08-18 16:23:04 | Adding engine-specific parameters.[39m
```

```
## [32m[Estimation] 2023-08-18 16:23:09 | Engine setup.[39m
```

```
## [32m[Estimation] 2023-08-18 16:23:09 | Starting fitting.[39m
```

```
## iinla: Iteration 1 [max:10]
```

```
## iinla: Iteration 2 [max:10]
```

```
## iinla: Max deviation from previous: 5.54e-06% of SD, and line search is inactive [stop if:
<10% and line search inactive]
```

```
## iinla: Convergence criterion met, running final INLA integration with known theta mode.
```

```
## iinla: Iteration 3 [max:10]
```

```
## [32m[Estimation] 2023-08-18 16:23:43 | Starting prediction.[39m
```

```
## [32m[Done] 2023-08-18 16:24:01 | Completed after 57.28 secs[39m
```

```
#pa+pa
mod_pa_pa <- distribution(background) %>%
  add_predictors(env = env_variables,
                 transform = "scale",
                 derivates = "none") %>%
  add_biodiversity_poipa(sp1_pa_ami,
                         name = "AMI trap pa data",
                         field_occurrence = "detected") %>%
  add_biodiversity_poipa(sp1_pa_timed,
                         name = "Timed count data",
                         field_occurrence = "detected") %>%
  engine_inlabru() %>%
  train(runname =  "sp1_pa_pa",
        verbose = T,
        aggregate_observations = F)
```

```
## [32m[Setup] 2023-08-18 16:24:01 | Creating distribution object...[39m
```

```
## [32m[Setup] 2023-08-18 16:24:01 | Adding predictors...[39m
```

```
## [32m[Setup] 2023-08-18 16:24:01 | Transforming predictors...[39m
```

```
## [32m[Setup] 2023-08-18 16:24:02 | Adding poipa dataset...[39m
## [32m[Setup] 2023-08-18 16:24:02 | Adding poipa dataset...[39m
```

```
## [32m[Estimation] 2023-08-18 16:24:02 | Collecting input parameters.[39m
```

```
## [32m[Estimation] 2023-08-18 16:24:03 | Adding engine-specific parameters.[39m
```

```
## [32m[Estimation] 2023-08-18 16:24:07 | Engine setup.[39m
```

```
## [32m[Estimation] 2023-08-18 16:24:07 | Starting fitting.[39m
```

```
## iinla: Iteration 1 [max:10]
```

```
## iinla: Iteration 2 [max:10]
## iinla: Max deviation from previous: 2.69e-05% of SD, and line search is inactive [stop if:
<10% and line search inactive]
```

```
## iinla: Convergence criterion met, running final INLA integration with known theta mode.
```

```
## iinla: Iteration 3 [max:10]
```

```
## [32m[Estimation] 2023-08-18 16:24:36 | Starting prediction.[39m
```

```
## [32m[Done] 2023-08-18 16:24:49 | Completed after 46.74 secs[39m
```

```
#all
mod_all <- distribution(background) %>%
  add_predictors(env = env_variables,
                 transform = "scale",
                 derivates = "none") %>%
  add_biodiversity_poipo(sp1_po,
                       name = "Citizen sci po data",
                       field_occurrence = "detected") %>%
  add_biodiversity_poipa(sp1_pa_ami,
                         name = "AMI trap pa data",
                         field_occurrence = "detected") %>%
  add_biodiversity_poipa(sp1_pa_timed,
                         name = "Timed count data",
                         field_occurrence = "detected") %>%
  engine_inlabru() %>%
  train(runname =  "sp1_po_pa_ami_timed",
        verbose = T,
        aggregate_observations = F)
```

```
## [32m[Setup] 2023-08-18 16:24:49 | Creating distribution object...[39m
```

```
## [32m[Setup] 2023-08-18 16:24:49 | Adding predictors...[39m
```

```
## [32m[Setup] 2023-08-18 16:24:49 | Transforming predictors...[39m
```

```
## [32m[Setup] 2023-08-18 16:24:50 | Adding poipo dataset...[39m
```

```
## [32m[Setup] 2023-08-18 16:24:50 | Adding poipa dataset...[39m
## [32m[Setup] 2023-08-18 16:24:50 | Adding poipa dataset...[39m
```

```
## [32m[Estimation] 2023-08-18 16:24:50 | Collecting input parameters.[39m
```

```
## [32m[Estimation] 2023-08-18 16:24:50 | Adding engine-specific parameters.[39m
## [32m[Estimation] 2023-08-18 16:24:54 | Engine setup.[39m
```

```
## [32m[Estimation] 2023-08-18 16:24:54 | Starting fitting.[39m
```

```
## iinla: Iteration 1 [max:10]
```

```
## iinla: Iteration 2 [max:10]
```

```
## iinla: Max deviation from previous: 5.54e-06% of SD, and line search is inactive [stop if:
<10% and line search inactive]
```

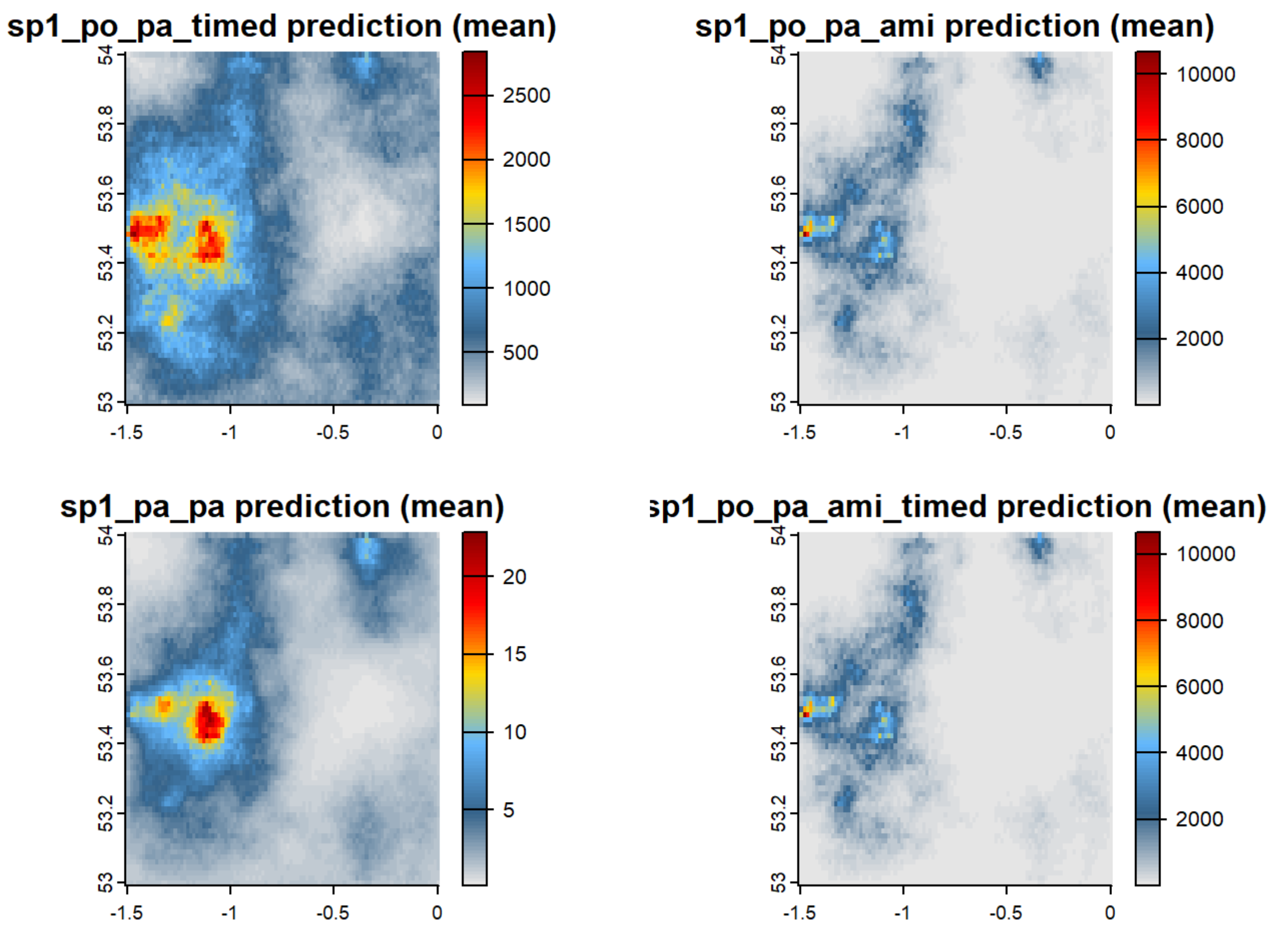
```
## iinla: Convergence criterion met, running final INLA integration with known theta mode.
```

```
## iinla: Iteration 3 [max:10]
```

```
## [32m[Estimation] 2023-08-18 16:25:29 | Starting prediction.[39m
```

```
## [32m[Done] 2023-08-18 16:25:48 | Completed after 58.56 secs[39m
```

```r
#compare integrated models
par(mfrow = c(2, 2))
plot(mod_po_pa_timed)
plot(mod_po_pa_ami)
plot(mod_pa_pa)
plot(mod_all)
```


sp1_po_pa_timed prediction (mean)
sp1_po_pa_ami prediction (mean)
sp1_pa_pa prediction (mean)
sp1_po_pa_ami_timed prediction (mean)
54
53.8
53.6
53.4
53.2
53
-1.5
-1
-0.5
0
2500
2000
1500
1000
500
10000
8000
6000
4000
2000
20
15
10
5

```
par(mfrow = c(1, 1))
```