DELphinus MOuvements GEStion

Octobre 2023

Cartographier le risque de captures de cétacés à partir des données d'effort de pêche et d'observation à la mer (package analytique R)

GOUVERNEMENT

FRANCE FILIÈRE PÊCHE

La Rochelle Université

cnrs

UBO Université de Bretagne Occidentale

Ifremer

COMITE NATIONAL DES PECHES CNPMEM

Durée du projet : 3 ans
Date de lancement : 01/03/2022
Date de fin : 30/06/2025
Coordinateurs de projet : Clara Ulrich, Pierre Petitgas, Jérôme Spitz, Marion PILLET.
Site web : https://delmoges.recherche.univ-lr.fr

Livrable

WP concerné : WP3
Responsables du WP : Dubroca Laurent, Authier Matthieu

Livrable L.3.3.1
Date de production : 27 octobre 2023

Titre : Cartographier le risque de captures de cétacés à partir des données d'effort de pêche et d'observation à la mer (package analytique R)

Auteurs : Authier Matthieu (La Rochelle Université), Brevet Mathieu (Ifremer), Dubroca Laurent (Ifremer).

Résumé

Depuis les années 1990, la France connaît régulièrement des épisodes de mortalités importantes de dauphins, qui entraînent des pics d'échouages sur le littoral Atlantique en hiver. Depuis 2016, les échouages de petits cétacés dans le golfe de Gascogne présentant des traces de capture, atteignent des niveaux inédits. Si les données scientifiques actuelles permettent d'évaluer globalement le risque induit par ces captures accidentelles pour la conservation de la population de dauphins communs, elles sont toutefois trop lacunaires pour comprendre les déterminants écosystémiques et halieutiques à l'origine de ces captures. En concertation avec l'Office français de la biodiversité, les professionnels de la pêche et l'Etat, l'Université la Rochelle-CNRS et l'Institut français de recherche pour l'exploitation de la mer (Ifremer) ont construit le projet Delmoges (Delphinus Mouvements Gestion). Il vise, dans un premier temps, à combler ces lacunes en allant chercher des nouvelles données sur les habitats des dauphins, sur leurs interactions trophiques dans l'écosystème et leurs interactions techniques avec les engins de pêche. Ensuite, le projet propose d'intégrer les connaissances sur l'ensemble du socio-écosystème pour envisager une diversité de scénarios de diminution des captures accidentelles incluant des solutions technologiques et, enfin, d'en évaluer les conséquences biologiques et socio-économiques.

Ce livrable est un package analytique permettant de mettre en œuvre un flux de travail reproductible afin d'estimer et de cartographier le risque de capture accidentelle d'espèces protégées, dont le dauphin commun (Delphinus delphis). Ce livrable sera amené à évoluer au cours des prochains mois en intégrant notamment d'autres fonctionnalités pour reproduire et pérenniser les analyses réalisées dans le WP3 à partir des données issues du SIH. Le package R analytique developpé s'appelle Pelarrp, où les initiales « RRP » signifient « Regularized Regression with Post-stratification ». Dans sa version initiale (v 0.1.0), le package inclut un ensemble de fonctions et de modèles statistiques pour analyser les données issues d'ObsMER.

Dissémination

Type de livrable : package analytique
Public : Oui
Lieux de stockage : Gitlab LRUniv

Consortium scientifique

La Rochelle Université
23 avenue Albert Einstein
BP 33060
17031 La Rochelle

https://www.univ-larochelle.fr/

Centre national de la recherche scientifique (CNRS)
3, rue Michel-Ange
75794 Paris cedex 16

https://www.cnrs.fr/fr

Institut Français pour l'Exploitation de la Mer (Ifremer)
1625 route de Sainte-Anne - CS 10070
29280 Plouzané

wwz.ifremer.fr/

Université de Bretagne Occidentale (UBO)
3 rue des Archives
CS93837
29238 Brest cedex 3

https://nouveau.univ-brest.fr/

Comité National des Pêches Maritimes et des Elevages Marins (CNPMEM)
134 avenue de Malakoff
75116 Paris

https://www.comite-peches.fr/

Table des matières

1 Contexte

1.1 CONTEXTE ENVIRONNEMENTAL ET SCIENTIFIQUE

Ce livrable vise à mettre en œuvre la reproductibilité d'une partie des analyses statistiques réalisées dans le cadre du WP3. La reproductibilité est importante pour garantir la qualité des résultats obtenus et permettre des ré-analyses ou de nouvelles analyses avec l'acquisition de nouvelles données.

1.2 ROLE DU LIVRABLE

Le livrable est un package à installer et utiliser avec le langage de programmation R (R Core Team 2022). Dans sa version 0.1.0, le package permet en outre de mettre en œuvre les analyses décrites dans Authier et al. (2021) et Rouby et al. (2022). Ce package est évolutif et intégrera de nouvelles fonctions d'ici la fin du projet Delmoges.

1.3 ACRONYMES ET ABREVIATIONS

RPP	Regularized Regression with Post-stratification[1]
SIH	Système d'Information Halieutique

[1] https://en.wikipedia.org/wiki/Multilevel_regression_with_poststratification

2 Installation et utilisation du package

Le package est compilé en un fichier zip à installer depuis une console R (R Core Team 2022). Il est hébergé à l'adresse suivante :

https://gitlab.univ-lr.fr/pelaverse

Néanmoins, étant encore en développement, le dépôt associé au package n'est pas public mais le deviendra à la version 1.0.0 du package, une fois d'autres fonctionnalités intégrées au packages.

Dans sa version 0.1.0, le package permet d'ajuster les modèles décrits par Authier et al. (2021) et Rouby et al. (2022). Ces modèles sont codés dans le langage de programmation probabilistique Stan (Carpenter et al. 2017 ; https://mc-stan.org/).

En outre, les fonctions incluses dans la version v.0.1.0 permettent de réaliser les analyses décrites, par exemple, dans la section 4 de cette saisine :

https://archimer.ifremer.fr/doc/00817/92863/99249.pdf

Le modèle utilisé dans cette saisine est disponible comme données, stockées sous la forme d'un texte à compiler en exécutable c++ avant ajustement sur des données empiriques. La confidentialité de ces dernières ne permet pas d'inclure un exemple reproductible dans le package.

Un exemple d'utilisation est :

```
> library(pelarrp)
>
> data("stanmodelcode")
> cat(stanmodelcode$gamma_multiyear)

data {
        int<lower = 1> n_op;
        int<lower = 1> n_week;
        int<lower = 1> n_zone;
        int<lower = 1> n_year;
  int<lower = 0, upper = 1> BYCATCH[n_op];
  vector<lower = 0.0>[n_op] DURATION;
  int<lower = 1, upper = n_week> WEEK[n_op];
  int<lower = 1, upper = n_zone> ZONE[n_op];
  int<lower = 1, upper = n_year> YEAR[n_op];
  int<lower = 1> n_maree;
  int<lower = 1> OP[n_maree];
  int<lower = 0> DaS[n_maree];
  int<lower = 1, upper = n_week> WEEK3[n_maree];
  int<lower = 1, upper = n_zone> ZONE3[n_maree];
  int<lower = 1, upper = n_year> YEAR3[n_maree];
  corr_matrix[n_week] R; // correlation matrix for intra-year level variations
  vector<lower = 0.0>[3] prior_scale_sigma;
  vector<lower = 0.0>[4] prior_scale_intercept;
  int<lower = 1> n_event;
  int<lower = 1> DOLPHIN[n_event];
  int<lower = 1, upper = n_year> YEAR4[n_event];
}

transformed data {
  int<lower = 0> shiftedOP[n_maree];
  for(i in 1:n_maree) {
    shiftedOP[i] = OP[i] - 1;
  }
}

parameters {
  vector[4] u_intercept;
  simplex[3] prop[3];
  vector<lower = 0.0>[3] sigma2;
  vector<lower = 0.0>[3] tau;
  vector[3] u_epsilon[n_week - 1];
  vector[n_week] u_alpha1[n_year, n_zone];
  vector[n_week] u_alpha2[n_year, n_zone];
  vector[n_week] u_alpha3[n_year, n_zone];
  vector[n_week] u_delta1[n_year];
  vector[n_week] u_delta2[n_year];
  vector[n_week] u_delta3[n_year];
  real<lower = 0.0> cv;
  cholesky_factor_corr[3] Omega[3];
}

transformed parameters {
  matrix[3, 3] CorMat[3];
  vector[3] rho[3];
  real shape = 1 / square(cv);
  vector[4] intercept;
  vector[3] sigma[3];
  vector[n_week] epsilon[3];
  vector[n_week] delta[3, n_year];
  vector[n_week] alpha[3, n_zone, n_year];
  vector[n_op] linpred[2];
  vector[n_maree] linpred3;
  // variance components
  sigma[1] = prior_scale_sigma[1] * sqrt(prop[1] * sigma2[1] / tau[1]); // 1-week,
2-zone, 3-year
  sigma[2] = prior_scale_sigma[2] * sqrt(prop[2] * sigma2[2] / tau[2]); // 1-week,
2-zone, 3-year
  sigma[3] = prior_scale_sigma[3] * sqrt(prop[3] * sigma2[3] / tau[3]); // 1-week,
2-zone, 3-year
  // intercepts
  intercept = prior_scale_intercept .* u_intercept;
  // random walk
  epsilon[1, 1] = intercept[1]; // logit scale
```

```
  epsilon[2, 1] = intercept[2]; // log scale
  epsilon[3, 1] = intercept[3]; // log scale
  for(t in 2:n_week) {
    epsilon[1, t] = epsilon[1, t - 1] + u_epsilon[t - 1, 1] * sigma[1, 1];
    epsilon[2, t] = epsilon[2, t - 1] + u_epsilon[t - 1, 2] * sigma[2, 1];
    epsilon[3, t] = epsilon[3, t - 1] + u_epsilon[t - 1, 3] * sigma[3, 1];
  }
  // correlated random effects: non centered parametrization
  // use cholesky factors Omega
  for(k in 1:n_year) {
    delta[1, k] = epsilon[1] + sigma[1, 3] * (Omega[3, 1, 1] * u_delta1[k]);
    delta[2, k] = epsilon[2] + sigma[2, 3] * (Omega[3, 2, 1] * u_delta1[k] + Omega[
3, 2, 2] * u_delta2[k]);
    delta[3, k] = epsilon[3] + sigma[3, 3] * (Omega[3, 3, 1] * u_delta1[k] + Omega[
3, 3, 2] * u_delta2[k] + Omega[3, 3, 3] * u_delta3[k]);
    for(j in 1:n_zone) {
      alpha[1, j, k] = delta[1, k] + sigma[1, 2] * (Omega[2, 1, 1] * u_alpha1[k, j]
);
      alpha[2, j, k] = delta[2, k] + sigma[2, 2] * (Omega[2, 2, 1] * u_alpha1[k, j]
+ Omega[2, 2, 2] * u_alpha2[k, j]);
      alpha[3, j, k] = delta[3, k] + sigma[3, 2] * (Omega[2, 3, 1] * u_alpha1[k, j]
+ Omega[2, 3, 2] * u_alpha2[k, j] + Omega[2, 3, 3] * u_alpha3[k, j]);
    }
  }
  // linear predictor
  for(i in 1:n_op) {
    linpred[1, i] = alpha[1, ZONE[i], YEAR[i], WEEK[i]];
    linpred[2, i] = exp(alpha[2, ZONE[i], YEAR[i], WEEK[i]]);
  }
  for(i in 1:n_maree) {
    linpred3[i] = alpha[3, ZONE3[i], YEAR3[i], WEEK3[i]] + log(DaS[i]);
  }
  for(m in 1:3) {
    CorMat[m] = multiply_lower_tri_self_transpose(Omega[m]);
    rho[m, 1] = CorMat[m, 1, 2];
    rho[m, 2] = CorMat[m, 1, 3];
    rho[m, 3] = CorMat[m, 2, 3];
  }
}

model {
  for(m in 1:3) {
    Omega[m] ~ lkj_corr_cholesky(1); // LKJ prior on the correlation matrix
  }
  cv ~ gamma(2.0, 5.0);
  u_intercept ~ normal(0.0, 1.0);
  sigma2 ~ gamma(0.5, 1.0);
  tau ~ gamma(1.0, 1.0);
  for(t in 1:(n_week - 1)) {
    u_epsilon[t] ~ multi_normal_cholesky(rep_vector(0.0, 3), Omega[1]);
  }
  for(k in 1:n_year) {
    u_delta1[k] ~ multi_normal(rep_vector(0.0, n_week), R);
    u_delta2[k] ~ multi_normal(rep_vector(0.0, n_week), R);
    u_delta3[k] ~ multi_normal(rep_vector(0.0, n_week), R);
    for(j in 1:n_zone) {
      u_alpha1[k, j] ~ multi_normal(rep_vector(0.0, n_week), R);
      u_alpha2[k, j] ~ multi_normal(rep_vector(0.0, n_week), R);
      u_alpha3[k, j] ~ multi_normal(rep_vector(0.0, n_week), R);
    }
  }
  target += bernoulli_logit_lpmf(BYCATCH| linpred[1]);
  for(i in 1:n_op) {
    target += gamma_lpdf(DURATION[i]| shape, shape / linpred[2, i]);
  }
  for(i in 1:n_maree) {
    target += poisson_log_lpmf(OP[i]| linpred3[i]) - log1m_exp(poisson_log_lpmf(0|
linpred3[i]));
  }
}

generated quantities {
  vector[n_op + n_maree] log_lik;
  real w[n_week, n_year, n_zone];
  for(i in 1:n_op) {
    log_lik[i] = bernoulli_logit_lpmf(BYCATCH[i]| linpred[1, i]) + gamma_lpdf(DURAT
ION[i]| shape, shape / linpred[2, i]);
```

```
  }
  for(i in 1:n_maree) {
    log_lik[n_op + i] = poisson_log_lpmf(OP[i]| linpred3[i]) - log1m_exp(poisson_lo
g_lpmf(0| linpred3[i]));
  }
  for(t in 1:n_week) {
    for(k in 1:n_year) {
      for(j in 1:n_zone) {
        w[t, k, j] = (exp(alpha[3, j, k, t]) / (1 - exp(-exp(alpha[3, j, k, t]))))
* inv_logit(alpha[1, j, k, t]);
      }
    }
  }
}
```

La commande ci-dessous permet de compiler ce modèle en exécutable c++ :

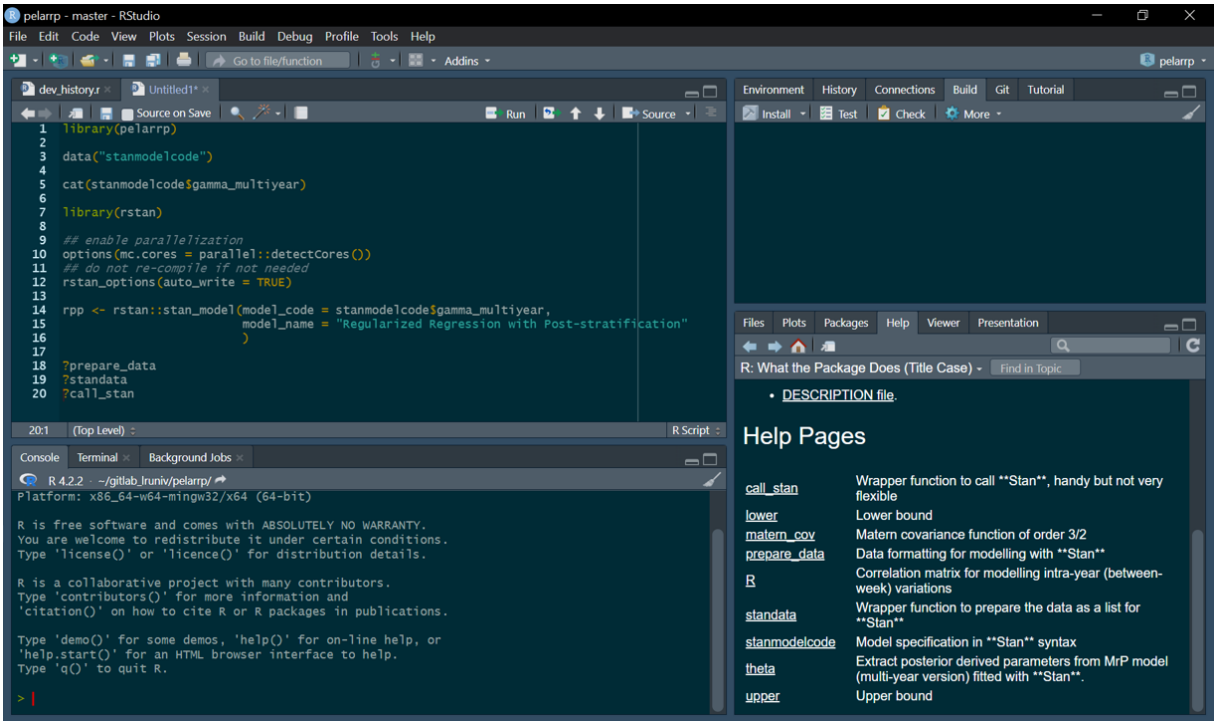
```
> library(rstan)
Le chargement a nécessité le package : StanHeaders

rstan version 2.26.23 (Stan version 2.26.1)
>
> ## enable parallelization
> options(mc.cores = parallel::detectCores())
> ## do not re-compile if not needed
> rstan_options(auto_write = TRUE)
>
> rpp <- rstan::stan_model(model_code = stanmodelcode$gamma_multiyear,
+                          model_name = "Regularized Regression with Post-stratific
ation"
+                          )
```

Le package inclut quelques fonctions pour faciliter la préparation des données (depuis un fichier extrait du SIH) :

```
> ?prepare_data
> ?standata
> ?call_stan
```

Le package pelarrp v.0.1.0 inclut actuellement 9 fonctions documentées :



3 Bibliographie

Authier, M., Rouby, E. & Macleod, K. (2021) Estimating Bycatch from Non-Representative Samples with (regularized) Multilevel Regression with Post-Stratification. Frontiers in Marine Science , Vol. 8, p. 719956. https://www.frontiersin.org/articles/10.3389/fmars.2021.719956/full

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P. & Riddell, A. (2017) Stan: A Probabilistic Programming Language. Journal of Statistical Software , Vol. 76, No. 1. https://www.jstatsoft.org/article/view/v076i01

R Core Team (2022). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.

Rouby, E., Dubroca, L., Cloâtre, T., Demanèche, S., Genu, M., Macleod, K., Peltier, H., Ridoux, V. & Authier, M. (2022) Estimating Bycatch from Non-Representative Samples (II): a Case Study on Common Dolphins in the Bay of Biscay. Frontiers in Marine Science , Vol. 8, No. 795942. https://www.frontiersin.org/articles/10.3389/fmars.2021.795942/full