Analyse de données fonctionnelles avec PenFoFPLS

Jeu de données : CanadianWeather

Description :

"**dailyAv**" un tableau tridimensionnel c(365, 35, 3) résumant les données recueillies dans 35 stations météorologiques différentes au Canada sur les points suivants : [,,1] = [,, 'Temperature.C'] : température quotidienne moyenne pour chaque jour de l'année [,,2] = [,, 'Precipitation.mm'] : précipitations quotidiennes moyennes pour chaque jour de l'année, arrondies à 0,1 mm. [,,3] = [,, 'log10precip'] : logarithme en base 10 de Precipitation.mm après avoir remplacé les 27 zéros par 0,05 mm (Ramsay et Silverman 2006, p. 248).

"**place**" Noms des 35 stations météorologiques différentes au Canada dont les données sont résumées dans "dailyAv". La longueur de ces noms varie entre 6 et 11 caractères. En revanche, daily[["lieu"]] comporte tous 11 caractères, les noms comportant moins de caractères étant prolongés par des blancs à la fin.

"**province**" nom de la province canadienne contenant chaque lieu.

"**coordonnées**" matrice numérique indiquant la latitude nord et la longitude ouest de chaque lieu.

"**région**" Laquelle des 4 zones climatiques contient chaque lieu : Atlantique, Pacifique, Continental, Arctique.

"**monthlyTemp**" Matrice de dimensions (12, 35) donnant la température moyenne en degrés Celsius pour chaque mois de l'année.

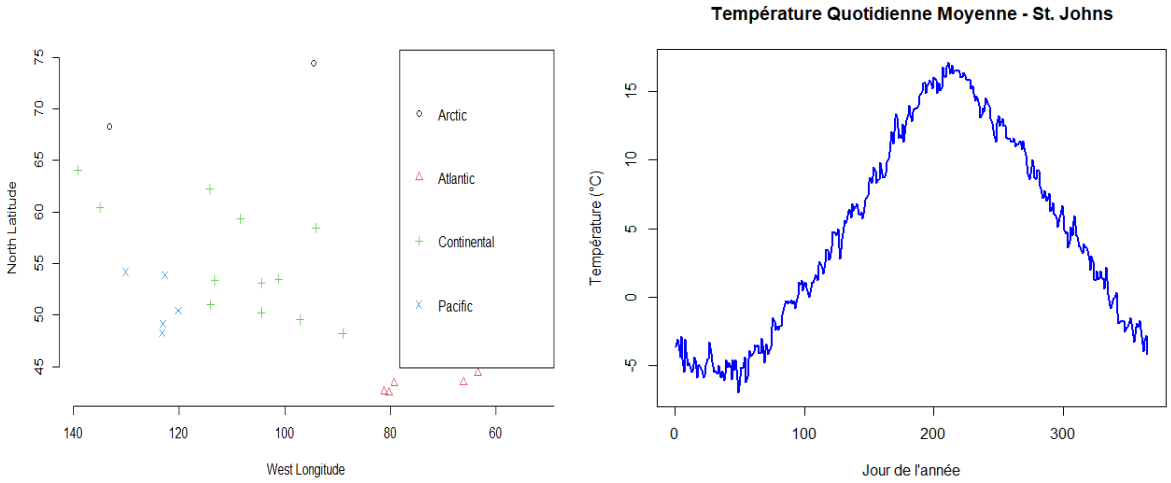
"**monthlyPrecip**" Matrice de dimensions (12, 35) donnant la moyenne des précipitations quotidiennes en millimètres pour chaque mois de l'année.

"**geogindex**" Ordre des stations météorologiques de l'est à l'ouest et au nord.

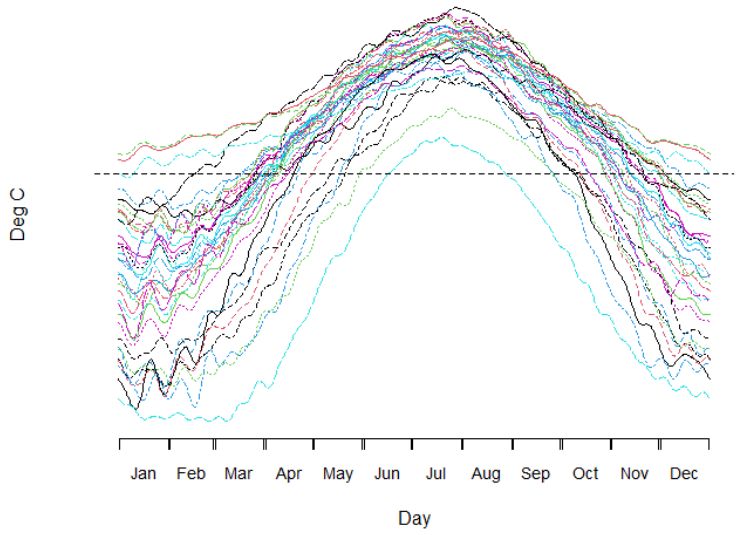
Quand dit-on que des données sont fonctionnelles ?

Des données sont fonctionnelles lorsque l'unité de mesure est une fonction plutôt qu'une valeur unique ou un vecteur. En d'autres termes, au lieu d'avoir des mesures individuelles à des points spécifiques, chaque observation représente une courbe, une trajectoire ou une fonction continue qui évolue généralement en fonction du temps, de l'espace ou d'une autre dimension.

Visualisation pour comprendre les données



On peut voir ici une répartition des stations météorologiques en fonction des longitudes et des latitudes. On rappelle que les stations sont au nombre de 35 sur la fig1. La deuxième figure, montre la variation de la température selon les jours de l'année à la station St Johns.



On observe que toutes les stations enregistrent des pics de températures de juillet à aout. Cela est du au fait que l'été est la saison la plus chaude au Canada. On observe aussi des stations qui relèvent des températures plus élevées que d'autres.

FPCA avec penFoFPLS

Nous avons choisi la température comme variable dépendante et la précipitation comme variable indépendante. Notre objectif principal est d'évaluer la performance du modèle dans sa capacité à généraliser les relations entre température et précipitation. On va donc analyser comment le modèle se comporte lorsqu'il est confronté à de nouvelles données.

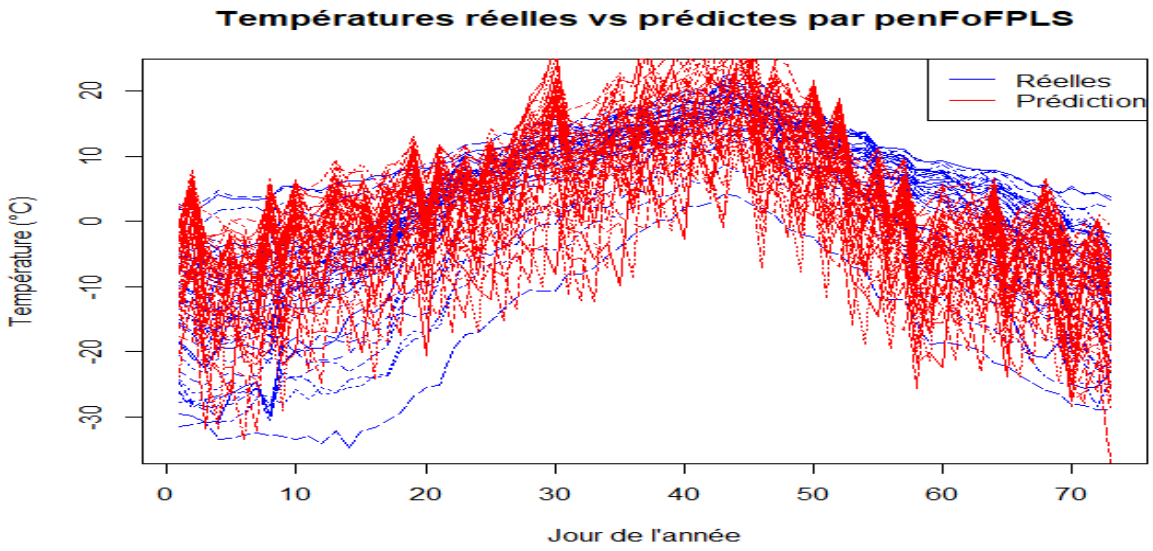
Pour ce faire, le jeu de données a été divisés en données d'entrainement et de test. Pour évaluer la performance du modèle à généraliser nous avons utilisé RMSE.

Première Partie

Choix des paramètres et ajustement avec ffpls_bs

```
argvals_X <- seq(0, 1, length.out = ncol(X_train))
argvals_Y <- seq(0, 1, length.out = ncol(Y_train))

basisobj_X <- fda::create.bspline.basis(rangeval = range(argvals_X), nbasis =
20)
basisobj_Y <- fda::create.bspline.basis(rangeval = range(argvals_Y), nbasis =
20)

ncomp <- 3
penalty_X <- 0
penalty_Y <- 0


model <- penFoFPLS::ffpls_bs(X = X_train,
                             Y = Y_train,
                             center = TRUE,
                             argvals_X = argvals_X,
                             argvals_Y = argvals_Y,
                             ncomp = ncomp,
                             basisobj_X = basisobj_X,
                             basisobj_Y = basisobj_Y,
                             penalty_X = penalty_X,
                             penalty_Y = penalty_Y)


# Faire des prédictions sur l'ensemble de test
prediction_penFo <- predict.ffpls_bs(model, newdata = X_test)
```

Visualisation



Evaluation avec RMSE

Pour évaluer la capacité du modèle à généraliser, nous avions utilisé le package caret qui permet d'utiliser plusieurs évaluateurs comme R2, MAE, RMSE, …

```
RMSE      MAE
10.19586 7.762459
```

Le RMSE (Root Mean Squared Error) mesure la Moyenne de l'erreur quadratique entre les valeurs prédites par le modèle et les valeurs réelles.

Le MAE (Mean Absolute Error) mesure la moyenne des valeurs absolues des différences entre les prédictions et les observations.

Deuxième partie

Nous allons appliquer le modèle dans une validation croisée k-fold. Cette technique va nous permettre de mieux évaluer la performance du modèle. Elle consiste à entrainer le modèle k fois, chaque fois en utilisant k-1 plis comme un ensemble d'entrainement et le pli restant comme ensemble de test. Pour chaque itération on évaluera la performance du modèle.

```
num_folds <- 5

folds <- createFolds(seq_len(nrow(CanadianWeather$dailyAv[, ,
"Temperature.C"])), k = num_folds, list = TRUE)

accuracy_scores <- numeric(num_folds)

for (fold in seq_along(folds)) {
  # Extract training and testing indices for the fold
  train_indices <- unlist(folds[-fold])
```

```
  test_indices <- unlist(folds[fold])

  # Extract the functional data for training and testing
  X_train_cv <- CanadianWeather$dailyAv[, , "Precipitation.mm"][train_indices,
]
  Y_train_cv <- CanadianWeather$dailyAv[, , "Temperature.C"][train_indices, ]

  X_test_cv <- CanadianWeather$dailyAv[, , "Precipitation.mm"][test_indices, ]
  Y_test_cv <- CanadianWeather$dailyAv[, , "Temperature.C"][test_indices, ]

  fdobj_cv <- penFoFPLS::ffpls_bs(X = X_train_cv,
                                  Y = Y_train_cv,
                                  center = TRUE,
                                  argvals_X = argvals_X,
                                  argvals_Y = argvals_Y,
                                  ncomp = ncomp,
                                  basisobj_X = basisobj_X,
                                  basisobj_Y = basisobj_Y,
                                  penalty_X = penalty_X,
                                  penalty_Y = penalty_Y)

  predictions <- predict.ffpls_bs(fdobj_cv, X_test_cv)

  accuracy <- RMSE(predictions[, , 1], Y_test_cv)

  # Store the accuracy for this fold
  accuracy_scores[fold] <- accuracy
}


print(accuracy_scores)
```

On obtient les RMSE de chaque pli :

```
10.200783 10.087635 10.146325 10.108767 9.780664
Une performance moyenne de 10.06483
```