Utilisation de la programmation dynamique pour la segmentation de series temporelles

Aline Canard*

20 mars 2019

Abstract

De nombreux domaines scientifiques et techniques ont besoin de représenter des paramètres qui évoluent au cours du temps et de les modéliser. Il s'agit de pouvoir en extraire différentes séquences de fonctionnement et pour chacun des régimes définis y proposer un modèle de régression. Ce document présente les résultats d'une segmentation par programmation dynamique et propose de modéliser chaque régime par une régression polynomiale.

*Master2 SAAD, Université de Caen Normandie

1 Introduction

Les séries temporelles interviennent dans de nombreux domaines tels que la finance, l'économie, ingénierie, la bioinformatique pour représenter les variations d'une mesure au cours du temps. Le domaine d'étude de ce type de données se nomme "analyse fonctionnelle" [RS05, RS02, FV06]. La philosophie repose sur l'idée de ne pas traiter les données comme des observations stockées dans un vecteur mais comme des valeurs (discrètes) d'une fonction. Les individus sont donc considérés non pas comme des vecteurs mais comme des fonctions. Des techniques de modélisation vont permettre de donner une représentation synthétique de tels phénomènes. Ce document présentera une de ces méthodes de segmentation de série temporelle : la régression par morceaux basée sur un modèle polynomial hétéroscédastique [CSGA09b, CSGA09a, CN19]. La méthode utilisée est la programmation dynamique. Les paramètres du modèle seront estimés par maximum de vraisemblance. En général, cette méthode est plutôt utilisée dans le cas de changement abruptes de régimes.

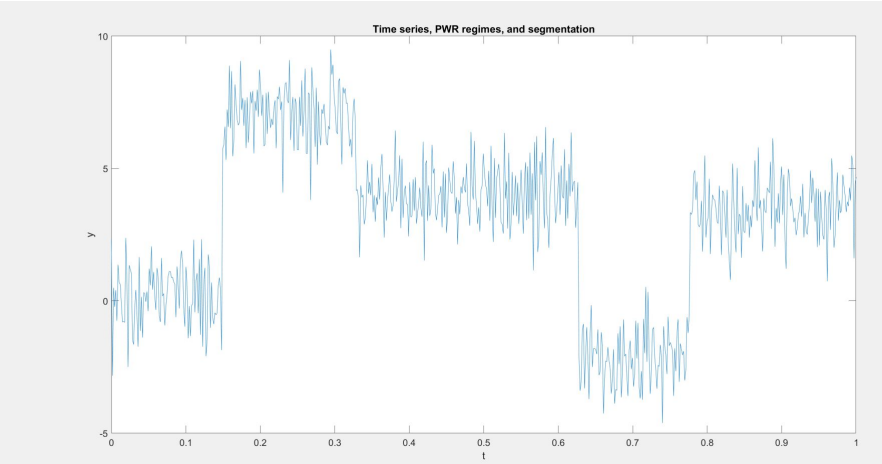

FIGURE 1 – Exemple de série temporelle à segmenter

Ce document a pour but de présenter la méthode mathématique et le code R réalisé pour effectuer la segmentation de telle série temporelle. Il est basé sur la publication : [RS05, RS02, FV06].

Dans une première partie, le modèle sera spécifié, et cela permettra de justifier le choix de la programmation dynamique comme méthode de segmentation. La deuxième partie s'attachera à expliquer les différentes fonction du code R et leur organisation. Enfin les repré-

sentation graphiques de la segmentation réalisée avec l'algorithme R sur un exemple de série temporelle seront présentées en fin de document.

2 Modèle de mélange pour des données fonctionnelles

2.1 Définition du modèle

Soit $\mathbf{x} = (\mathcal{X}_1, ..., \mathcal{X}_n)$, une observation d'une série temporelle où les x_i sont observés au temps t_i. Le modèle de régression polynomiale par morceaux suppose que la série temporelle soit composée de K régimes sur K intervalles dont les indices de bornes peuvent être notés : $\boldsymbol{\gamma} = (\gamma_1, ..., \gamma_{K+1})$ avec $\gamma_1 = 0$ et $\gamma_{K+1} = n$. Cela définit une partition de la série temporelle en K segments polynomiaux $(\mathbf{x}_1, ..., \mathbf{x}_K)$, de longueur $n_1, ..., n_K$ où $\mathbf{x}_K = \{\mathcal{X}_i | i \in I_K\}$ est l'ensemble des éléments d'un segment k dont les indices sont : $I_K = (\boldsymbol{\gamma}_K, \boldsymbol{\gamma}_{K+1}]$. Les modèles standards de régression polynomiale sont des modèles homoscédastiques puisqu'ils supposent que les différents modèles de régression polynomiale ont la même de bruit. Dans notre cas, nous considérons le cadre plus général d'un modèle hétéroscédastique ; ce qui permet d'avoir une variance de bruit entre les différents modèles de régression polynomiale.On peut le définir ainsi :

$$\forall i = 1, ..n, \mathbf{x}_i = \boldsymbol{\beta}_k^T \mathbf{r}_i + \sigma_k \epsilon_i \ ; \ \epsilon_i \sim \mathcal{N}(0, 1), \tag{1}$$

où k est tel que $i \in I_k, \boldsymbol{\beta}_k$ est le vecteur des coefficients de la $(p+1)^{eme}$ dimension d'un polynôme de degré p associé au k^{eme} segment, avec $k \in \{1, .., K\}$, $\mathbf{r}_i = (1, t_i, t_i^2, ..., t_i^p)^T$ est le vecteur de monômes de dimension $(p+1)$ associé au paramètre $\boldsymbol{\beta}_k$, et les ϵ_i sont des variables aléatoires, indépendantes et distribuées selon une loi normale. Elles représentent un bruit additionnel dans chaque segment k.

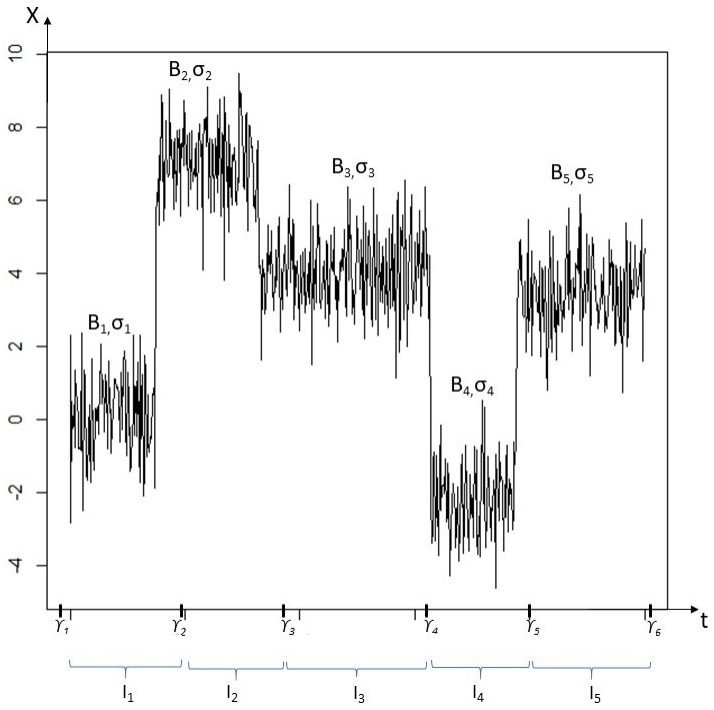

FIGURE 2 – Série temporelle avec les notations

L'objectif de la segmentation sera de positionner les $\boldsymbol{\gamma}_k$ de façon optimale.

$$\text{A l'instant } t, \text{ on peut écrire : } \mathbf{x}_t = \begin{cases} \boldsymbol{\beta}_1^T \mathbf{r}_t + \sigma_1 \epsilon_t, \; si \; \mathbf{x}_t \in I_1 \\ \boldsymbol{\beta}_2^T \mathbf{r}_t + \sigma_2 \epsilon_t, \; si \; \mathbf{x}_t \in I_2 \qquad \text{où } \epsilon_t \; \sim \mathcal{N}(0,1) \\ ... \\ \boldsymbol{\beta}_K^T \mathbf{r}_t + \sigma_K \epsilon_t, \; si \; \mathbf{x}_t \in I_K \end{cases}$$

2.2 Estimation des paramètres du modèle par maximum de vraisemblance

Avec ce modèle, les paramètres peuvent s'écrire de la forme : $(\boldsymbol{\psi}, \boldsymbol{\gamma})$ où $\boldsymbol{\psi} = (\boldsymbol{\beta}_1, ..., \boldsymbol{\beta}_k, \boldsymbol{\sigma}_1^2, .., \boldsymbol{\sigma}_k^2)$ est l"ensemble des coefficients polynomiaux et des variances de bruit, et $\boldsymbol{\gamma} = (\gamma_1, ..., \gamma_{K+1})$ l'ensemble des points de rupture. L'estimation des paramètres se fera par maximum de vraisemblance. L'indépendance conditionnelle des données est supposée. Ainsi, d'après le modèle défini dans l'équation (Equation1), on peut montrer que dans chaque segment k, $\mathbf{x}_i$ a une distribution gaussienne de moyenne $\boldsymbol{\beta}_k^T \mathbf{r}_i$ et de variance σ_k^2 et, par conséquent, que la log-

vraisemblance du vecteur paramètre $(\boldsymbol{\psi}, \boldsymbol{\gamma})$ caractérisant le modèle de régression polynomiale par morceaux est la somme des log-vraisemblances locales sur les K segments comme suit :

$$\begin{aligned}
L(\boldsymbol{\psi}, \boldsymbol{\gamma}, \mathbf{x}) &= \prod_{k=1}^{K} p(\mathbf{x}_k, \boldsymbol{\psi}, \boldsymbol{\gamma}) \\
\ln(L(\boldsymbol{\psi}, \boldsymbol{\gamma}, \mathbf{x})) &= \sum_{k=1}^{K} ln(p(\mathbf{x}_k, \boldsymbol{\psi}, \boldsymbol{\gamma})) \\
&= \sum_{k=1}^{K} ln(\prod_{t=\gamma_K+1}^{\gamma_{K+1}} \mathcal{N}(\mathbf{x}_t, \boldsymbol{\beta}_k^T \mathbf{r}_i, \sigma_k^2)) \\
&= \sum_{k=1}^{K} ln(\prod_{t \in I_k} \mathcal{N}(\mathbf{x}_t, \boldsymbol{\beta}_k^T \mathbf{r}_i, \sigma_k^2)) \\
&= \sum_{k=1}^{K} \sum_{t \in I_k} ln(\mathcal{N}(\mathbf{x}_t, \boldsymbol{\beta}_k^T \mathbf{r}_i, \sigma_k^2)) \\
&= \sum_{k=1}^{K} \sum_{t \in I_k} ((-\frac{1}{2}))(\frac{\mathbf{x}_t - \boldsymbol{\beta}_k^T \mathbf{r}_i}{\sigma_k})^2 + \ln(\sigma_k^2)) + Constante
\end{aligned} \tag{2}$$

Maximiser la log-vraisemblance revient à minimiser le critère ci-dessous selon les paramètres $\boldsymbol{\psi}$ et $\boldsymbol{\gamma}$:

$$J(\boldsymbol{\psi}, \boldsymbol{\gamma}) = \sum_{k=1}^{K} \left[\frac{1}{\sigma_k^2} \sum_{t \in I_k} (\mathbf{x}_t - \boldsymbol{\beta}_k^T \mathbf{r}_i)^2 + n_k \ln(\sigma_k^2) \right] \tag{3}$$

où n_k est le nombre d'éléments dans le segment k.

Vue la forme de cette log-vraisemblance (en log de somme), il n'est pas possible de trouver une solution analytique. Par contre, comme le critère J est additif sur les K segments, on peut utiliser l'algorithme de Fisher (Fisher, 1958 ; Lechevallier, 1990), procédé de programmation dynamique (Bellman, 1961 ; Brailovsky et Kempner, 1992) pour réaliser la minimisation globale. Cet algorithme est récursif et alterne entre l'optimisation des γ et des $\boldsymbol{\beta}$ et $\boldsymbol{\sigma}$. Ce procédé dynamique a une complexité en temps en $O(Kp^2n^2)$; ce qui peut être coûteux en temps machine dans le cadre de jeux de données volumineux.

2.3 Approximation et segmentation d'une série temporelle par la méthode de régression par morceaux

Une fois les paramètres estimés, la segmentation d'une série temporelle, qui peut être représentée par le vecteur des classes : $\widehat{\mathbf{z}} = (\widehat{z}_1, ..., \widehat{z}_n)$, où $\widehat{z}_i \in \{1, ..K\}$, peut être déduit en posant $\widehat{z}_i = k$ si $i \in (\widehat{\boldsymbol{\gamma}}_K, \widehat{\boldsymbol{\gamma}}_{K+1}]$; les paramètres $(\widehat{\boldsymbol{\psi}}, \widehat{\boldsymbol{\gamma}})$ ayant été obtenus par le procédé de programmation dynamique. Puis une approximation de la série temporelle est donnée par $\widehat{\mathbf{x}}_i = \sum_{k=1}^{K} \widehat{z}_{ik} \widehat{\boldsymbol{\beta}}_k^T \mathbf{r}_i$ où $\widehat{z}_{ik} = 1$ si $\widehat{z}_i = k$ et $\widehat{z}_{ik} = 0$ sinon. La formulation vectorielle de la série temporelle approchée $\widehat{\mathbf{x}}$ peut s'écrire :

$$\widehat{\mathbf{x}} = \sum_{k=1}^{K} \widehat{\mathbf{z}}_k \mathbf{T} \widehat{\boldsymbol{\beta}}_k \tag{4}$$

où $\widehat{\mathbf{z}}_k$ est une matrice diagonale dont les éléments diagonaux sont $(\widehat{z}_1, ..., \widehat{z}_n)$ et

$$\mathbf{T} = \begin{pmatrix} 1 & \mathbf{t}_1 & \mathbf{t}_1^2 & \cdots & \mathbf{t}_1^p \\ 1 & \mathbf{t}_2 & \mathbf{t}_2^2 & \cdots & \mathbf{t}_2^p \\ \vdots & \vdots & \vdots & \vdots & \\ 1 & \mathbf{t}_n & \mathbf{t}_n^2 & \cdots & \mathbf{t}_n^p \end{pmatrix}$$

est la matrice de régression de dimension $[n \times (p+1)]$.

3 Description de l'algorithmique de segmentation

Le programme complet de segmentation d'une série temporelle selon le modèle de régression polynomiale par morceaux par la méthode de programmation dynamique a été implémenté en R. La programmation dynamique Le principe de cette méthode consiste à décomposer le problème en sous-problèmes et à établir une relation de récurrence avec les solutions optimales des sous-problème. L'algorithme proposé par Lechevalier en 1990 est basé sur l'approche de Fisher. Le principe est de maximiser dans ce cas la somme des log-vraisemblances des K sous-modèles de régression relatifs aux K classes ou intervalles, qui est un critère additif sur les K intervalles. Le programme en R contient 6 fonctions. La manière dont ces fonctions s'imbriquent est présentée via la carte mentale ci-dessous (Fig3) :

3.1 Carte mentale de l'ensemble du programme

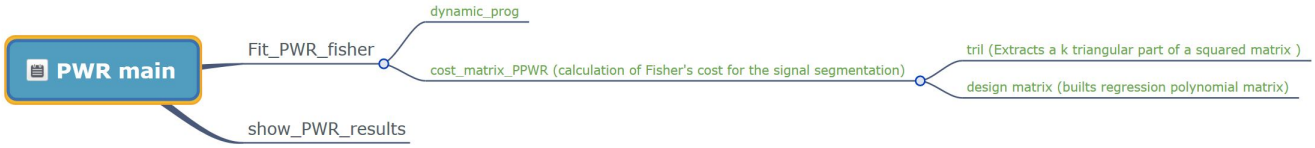

FIGURE 3 – Carte mentale de l'algorithme

3.2 Description de la fonction : designmatrix

Cette fonction, permet, à partir d'un vecteur $\mathbf{r}_i$ et d'un degré de polynôme p, de construire la matrice de régression polynomiale de degré p :

$$\mathbf{T} = \begin{pmatrix} 1 & \mathbf{t}_1 & \mathbf{t}_1^2 & \cdots & \mathbf{t}_1^p \\ 1 & \mathbf{t}_2 & \mathbf{t}_2^2 & \cdots & \mathbf{t}_2^p \\ \vdots & \vdots & \vdots & \vdots & \\ 1 & \mathbf{t}_n & \mathbf{t}_n^2 & \cdots & \mathbf{t}_n^p \end{pmatrix}$$

3.3 Description de la fonction : tril

Cette fonction a pour but de récupérer un k-triangule d'une matrice. k permet de déterminer à partir de quelle diagonale on fait l'extraction, et selon le signe de k, si on récupère les valeurs au dessus de cette kème diagonale ou en dessous. Cette fonction existe dans Matlab, mais il a fallu la concevoir pour R.

3.4 Description de la fonction : cost-matrix-PPWR

Cette fonction a pour objectif de calculer la matrice coût de Fisher pour la segmentation du signal sur un intervalle $[a, b]$. Elle est définie comme suit :

$$C(a,b) = \sum_{t=a}^{t=b} [ln(\sigma^2) + (\mathbf{x}_t - mu)^2/\sigma^2], \tag{5}$$

avec mu $= \boldsymbol{\beta}^T \mathbf{r}_t$: un polynôme d'ordre p. Ici $\boldsymbol{\beta}$ se calcule pour chaque couple (a,b).

La fonction prend en entrée la variable d'intérêt de la série temporelle, le vecteur temps, l'ordre du polynôme de régression et Lmin : le nombre de points minimum dans un segment

(par défaut Lmin = 1) et ressort la matrice de coût, de dimension $n \times n$. La matrice de coût a au préalable été initialisée à l'infini.

Le code R a été amélioré par rapport à une limitation existant dans le code Matlab :une condition a été ajoutée sur la boucle de parcours de la matrice pour éviter d'en sortir.

3.5 Description de la fonction : dynamic-prog

Cette fonction réalise l'algorithme de programmation dynamique. Elle cherche la partition qui minimise le critère $J(\boldsymbol{\psi}, \boldsymbol{\gamma})$. Elle prend comme entrée la matrice de coût sur l'intervalle $[a, b]$. Cette fonction ressort le critère $J(\boldsymbol{\psi}, \boldsymbol{\gamma})$ (vecteur de dimension $K \times 1$ contenant le critère pour chaque valeur du nombre d'intervalles k qui varie de 1 jusqu'à K et le temps de changement (partition) estimés (estimée) pour chaque valeur de k. C'est une matrice de dimension $Kmax \times Kmax$.

Une fois la partition estimée, les coefficients de régression associés à chaque segment ainsi que la variance du bruit sur chaque segment seront calculés par la méthode du maximum de vraisemblance dans la fonction suivante.

3.6 Description de la fonction : fit-PWR-fisher

Une fois la partition estimée par la fonction précédente : "*dynamic_prog*", cette fonction calcule les coefficients de régression associés à chaque segment ainsi que la variance du bruit sur chaque segment par la méthode du maximum de vraisemblance.

3.7 Description de la fonction : show-PWR-results

Cette fonction réalise la représentation graphique de la série initiale, de sa segmentation et des courbes de régression pour chaque régime. Deux graphiques seront générés par le code, dans des fenêtres externes.

3.8 Description de la fonction : main-PWR-fisher

Cette dernière fonction (le programme principal) permet d'appeler les bibliothèques R nécessaires, définit les variables K et p, va récupérer le jeu de données et enfin source toutes les fonctions précédemment décrites pour faire tourner le programme. Ainsi 2 courbes s'affichent dans 2 fenêtres :

- la première (Fig4) présente la série temporelle initiale, et les polynômes qui l'approchent pour chaque régime. Chacune des courbes des polynômes a sa propre couleur, et la partie correspondant au régime associé est présentée en trait plus épais (en pointillé pour le reste).
- la seconde (Fig5) représente la série temporelle initiale, et la segmentation optimale avec en rouge la courbe de régression pour chaque régime.

Les figures ci-dessous présentent les sorties du code R pour un exemple de série temporelle.

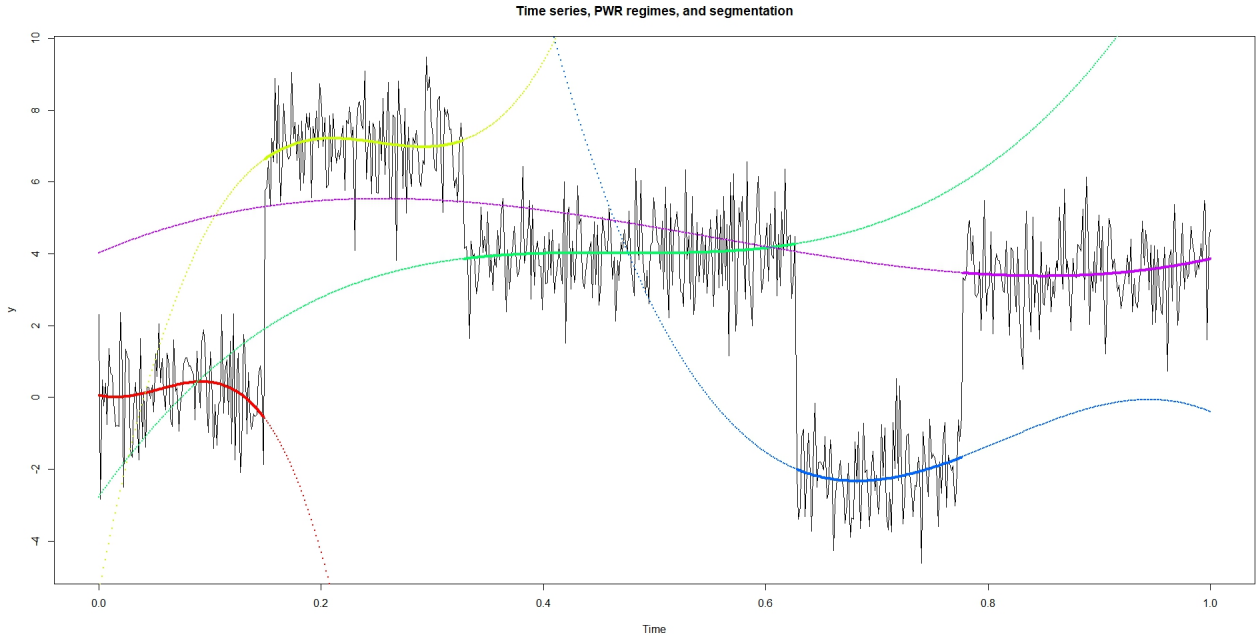

FIGURE 4 – Série temporelle et fonctions de régressions polynomiales

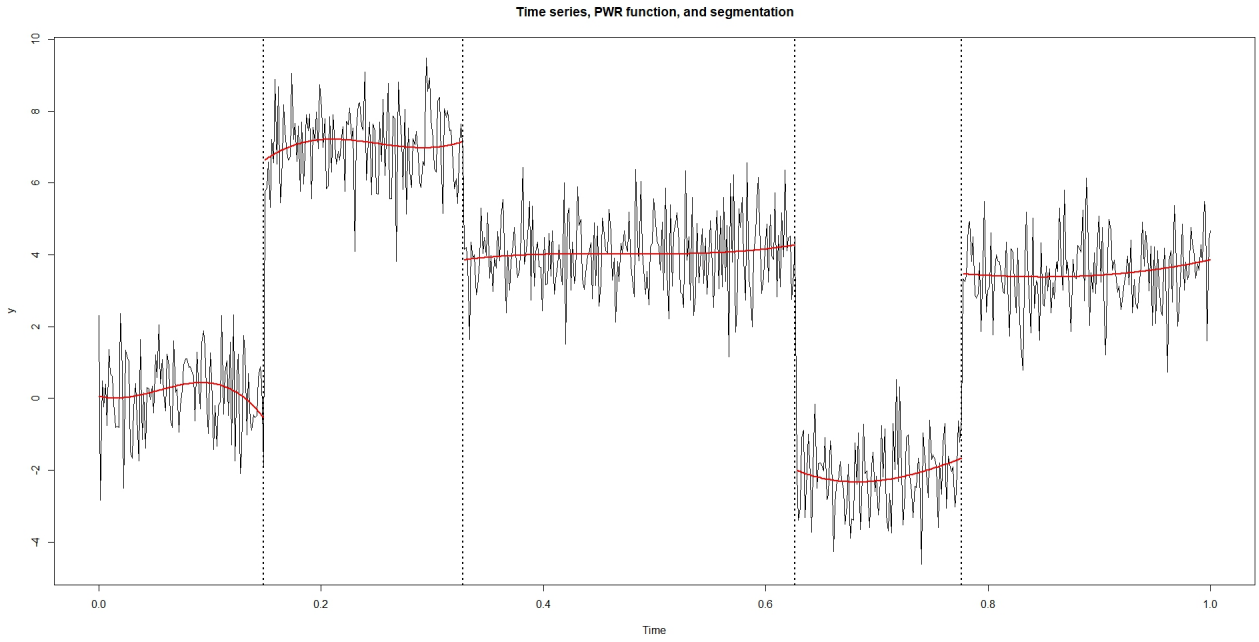

FIGURE 5 – Série temporelle segmentée et approximée par les régressions polynomiales

4 Conclusion

Cette méthode de segmentation a donc permis de retrouver les différents régimes de la série temporelle et les coefficients de régression pour chaque région. Elle peut fonctionner pour n'importe quel degré de polynôme à partir de $p = 0$. Par rapport au code Matlab, il a fallu rajouter une fonction d'extraction d'un triangle dans une matrice (qui n'existe pas en R), et séparer le cas p=0 car R considère qu'une matrice de dimension 1 est un vecteur et ne permet pas de faire des calculs matriciels. En terme de vitesse d'exécution des programmes, Matlab est plus de 3 fois plus rapide, avec 8.63 secondes contres 29.1 secondes sous R. Enfin, les représentations graphiques et les résultats calculatoires des 2 programmes (R et Matlab) sont exactement les mêmes.

Références

[CN19] Faicel Chamroukhi and Hien D. Nguyen. Model-based clustering and classification of functional data. *Wiley Interdisciplinary Reviews : Data Mining and Knowledge Discovery*, to appear 2019. DOI : 10.1002/widm.1298.

[CSGA09a] F. Chamroukhi, A. Samé, G. Govaert, and P. Aknin. A regression model with a hidden logistic process for feature extraction from time series. In *International Joint Conference on Neural Networks (IJCNN)*, pages 489–496, Atlanta, GA, June 2009.

[CSGA09b] F. Chamroukhi, A. Samé, G. Govaert, and P. Aknin. Time series modeling by a regression approach based on a latent process. *Neural Networks*, 22(5-6) :593–602, 2009.

[FV06] Frédéric Ferraty and Philippe Vieu. *Nonparametric functional data analysis : theory and practice*. Springer series in statistics, 2006.

[RS02] J. O. Ramsay and B. W. Silverman. *Applied Functional Data Analysis : Methods and Case Studies*. Springer Series in Statistics. Springer, 2002.

[RS05] J. O. Ramsay and B. W. Silverman. *Functional Data Analysis*. Springer Series in Statistics. Springer, New York, June 2005.