INSTITUT NATIONAL
DES SCIENCES
APPLIQUÉES
ROUEN NORMANDIE

https://fr.tradingview.com/symbols/TVC-CAC40/

LESOURD Nathan

LOUKIANENKO Tristan

à l'attention de M. Hérault.

Projet TSA - 2021/2022

ITI3

Table des matières

1. Étude bibliographique sur le modèle

a. Présentation

Le filtre de Wiener est un filtre linéaire, causal et à réponse impulsionnelle finie (RIF, c'est-à-dire que sa réponse est seulement basée sur les valeurs du signal en entrée). On utilise ce filtre pour faire de la restauration de signal, c'est-à-dire retrouver le signal d'entrée $x(t)$ bruité par le bruit $b(t)$.

b. Histoire

Le filtre tient son nom du mathématicien américain Norbert Wiener, il a publié ses travaux sur ce filtre en 1949 mais il travaillait déjà dessus dans les années 40. Bien que ce soit le nom de Wiener qui ait été retenu pour ce filtre, on sait que le mathématicien russe Andreï Kolmogorov travaillait sur les mêmes types de filtres à la même époque.

c. Fonctionnement

Soit l'observation $y[n] = x[n] + b[n]$, on voit que le bruit est considéré comme additif.
Soit $h[n]$, la réponse impulsionnelle finie du filtre de Wiener, égale à h_n entre 0 et N-1 et nulle ailleurs (filtre RIF d'ordre N-1).

Si on considère $\widehat{x}[n]$ la sortie du filtre, donc une estimation de $x[n]$, telle que $\widehat{x}[n] = \sum_{i=0}^{N-1} h[i] * y[n-i]$, alors le but du filtre de Wiener est de minimiser $\xi^2 = E[|e[n]|^2]$, où $e[n]$ est l'erreur d'estimation telle que $e[n] = x[n] - \widehat{x}[n]$.

Pour cela, on dispose des équations de Wiener-Hopf:
pour tout $k \in [0, N-1]$, on a:

$$R_{XY}[k] = \sum_{i=0}^{N-1} h[i] * R_{YY}[k-i]$$

On peut alors obtenir l'erreur quadratique minimale $\xi^2_{min} = R_X[0] - \sum_{i=0}^{N-1} h[i] * r_{XY}[i]$.

De toutes ces équations, nous pouvons déterminer $h[n]$ ce qui nous permet de définir notre filtre.

d. Utilisation du filtre de nos jours

Le filtre de Wiener est encore beaucoup utilisé de nos jours. Encore très récemment il a été utilisé pour cartographier le rayonnement de fond cosmique (grâce à des réseaux de neurones) ou encore en mathématiques sur des graphes couplé à des algorithmes d'approximation polynomiale.

2. Étude bibliographique sur l'application

a. Présentation de l'indice boursier

Le CAC 40 voit le jour le 31 décembre 1987 et s'impose aujourd'hui comme le principal indice boursier de la place de Paris. C'est un panier composé de 40 valeurs de sociétés françaises. Ces entreprises sont sélectionnées par un comité d'expert de l'Euronext parmi 100 entreprises dont les échanges de titres sont les plus importants. L'Euronext est la principale place boursière en Europe avec plus de 1900 émetteurs représentant une capitalisation boursière totale de 6 400 milliards d'euros.
L'objectif de cette sélection n'est pas de prendre les 40 entreprises les plus puissantes et les plus riches en France. Le comité sélectionne les entreprises afin d'obtenir un panier représentatif des différents secteurs d'activité en France.
L'acronyme CAC signifie Cotation Assisté en Continu et 40 pour la sélection de 40 entreprises parmi les 100 entreprises françaises ayant les capitalisations les plus importantes. Cela veut dire que la valeur de l'indice est actualisée en continu, à vrai dire toutes les 15 secondes de 9h du matin à 17h30 le soir. Cela veut dire aussi que les entreprises composant ce panier sont amenées à changer. En effet, en cas de crise d'une entreprise ou au contraire à l'essor d'une autre, le comité de l'Euronext peut être amené à changer la constitution du panier.

b. Intérêts de cet indice

Mais en quoi cet indice est-il pertinent? De par sa constitution, il offre une représentativité de la santé de l'économie française. Cette représentativité est d'autant plus forte que tous les secteurs d'activités sont pris en compte de manière égale. De plus, du fait de l'internationalisation des entreprises françaises, le CAC 40 s'avère être un indice ayant non plus une échelle nationale mais européenne voire mondiale.

c. Travaux d'estimation du CAC 40

Après plusieurs recherches, nous n'avons pas trouvé de travaux utilisant le filtre RIF pour la prédiction d'indice boursier. En revanche, d'autres méthodes plus perfectionnées ont été mises au point en utilisant le filtre de Kalman ou encore en prenant en compte la stochastique de la bourse. Pour cette étude, nous avons fait le choix de nous restreindre à un filtre à réponse impulsionnelle ne s'intéressant pas au caractère aléatoire des crises boursières. Le filtre RIF est un modèle à boîte noire, c'est-à-dire qu'on ne connaît pas les règles de fonctionnement de la bourse. Voyons les résultats que ce modèle peut nous apporter et ses limites.

3. Présentation du jeu de données choisi

Dans le contexte de ce projet, nous avons récupéré des jeux de données du CAC 40. Pour étudier notre modèle et pouvoir comparer les résultats, nous avons sélectionné trois temporalités différentes : au jour, à la semaine et au mois. Pour ces trois différentes temporalités nous avons retenu à chaque fois une période d'environ 1 an, c'est-à-dire les données de l'année passée. Ces données sont d'actualités car elles sont les dernières valeurs du CAC 40 à compter du 20/05/2022, date de téléchargement des datasets.

De plus, nous avons fait le choix de retenir le cours d'ouverture, c'est-à-dire la valeur de l'indice à 9h du matin. Ce choix reste arbitraire et n'a que peu d'importance dans l'étude de la prédiction du CAC40.

a. Cours d'ouverture du CAC 40 : Quotidien

source: https://fr.finance.yahoo.com/
format: csv
temporalité: quotidien
date: 02/09/2021 → 19/05/2022 (259 jours)
taille échantillon: 259
min: 5963
max: 7376
mean: 6711

b. Cours d'ouverture du CAC 40 : Hebdomadaire

source: https://fr.finance.yahoo.com/
format: csv
temporalité: hebdomadaire
date : 06/05/2021→ 19/05/2022 (54 semaines)
taille échantillon: 54
min: 6062
max: 7219
mean: 6693

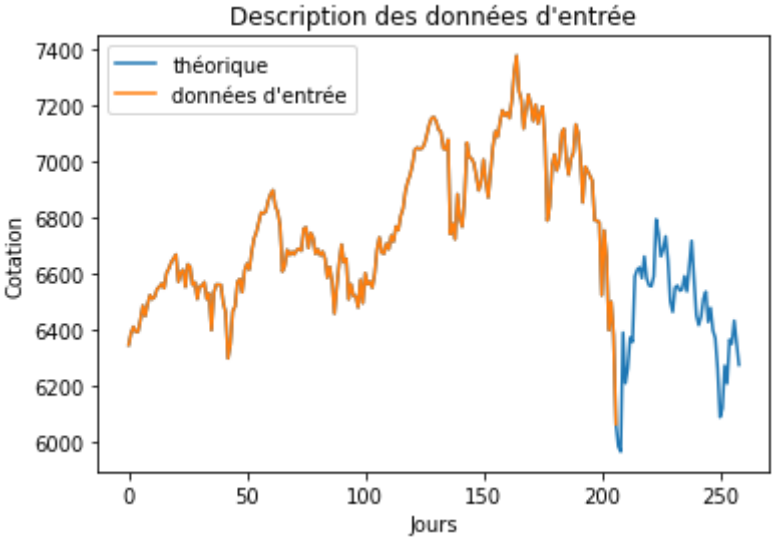
c. Cours d'ouverture du CAC 40 : Mensuel

source: https://fr.finance.yahoo.com/
format: csv
temporalité: mois
durée: 19/04/2021→ 19/05/2022 (13 mois)
taille échantillon: 13
min: 6273
max: 7153
mean: 6648

4. Explication de l'implémentation

a. Partition du jeu de données

Tout d'abord, nous avons divisé nos différents jeux de données: nous avons gardé 80% des données pour l'échantillon qui nous servira pour construire la prédiction (c'est-à-dire pour "l'apprentissage")(ici, la courbe orange) et 20% des données pour comparer notre prédiction à la réalité (ici, la courbe bleue).



b. Calcul des coefficients du filtre

Les coefficients du filtre ne sont pas des hyper-paramètres, en effet ce sont des paramètres que l'on calcule, on ne les fournit pas au modèle directement. Nous avons donc implémenté une fonction qui nous permet d'obtenir les coefficients pour la prédiction sur un intervalle donné:

```
xcorr = lambda X,Y:np.correlate(X,Y,mode='full')

def predicteurlineaire_step(x,step,p):
    n = x.shape[0]
    RXX = xcorr(x,x)[n-1:]
    return scl.solve_toeplitz(RXX[:p], RXX[step:step+p]) #on prend
jusque p car on est en python, en théorie on prend jusqu'à p-1
```

c. Prédictions

L'intégralité de notre code se trouve dans les annexes (partie implémentation).

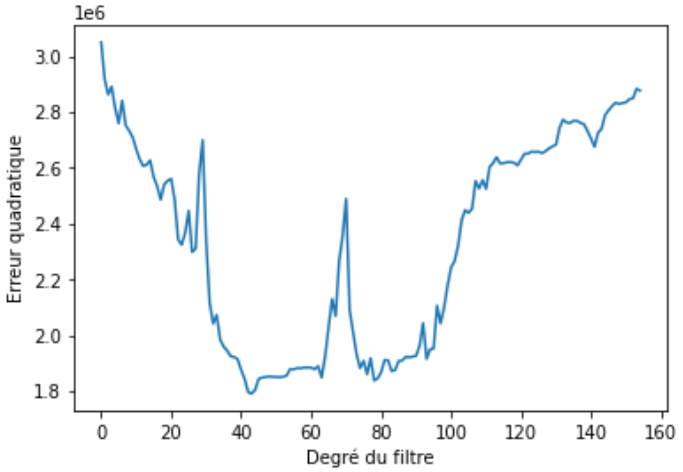
5. Choix des hyper-paramètres

Avec le filtre RIF nous possédons deux paramètres permettant de contrôler le processus de prédiction : la taille de l'échantillon et le degré du filtre. En ce qui concerne la taille de l'échantillon nous avons fait le choix de nous intéresser à la dernière année passée pour différentes temporalités. Par conséquent, les tailles de nos trois échantillons sont fixées et définies précédemment lors de la présentation des jeux de données.

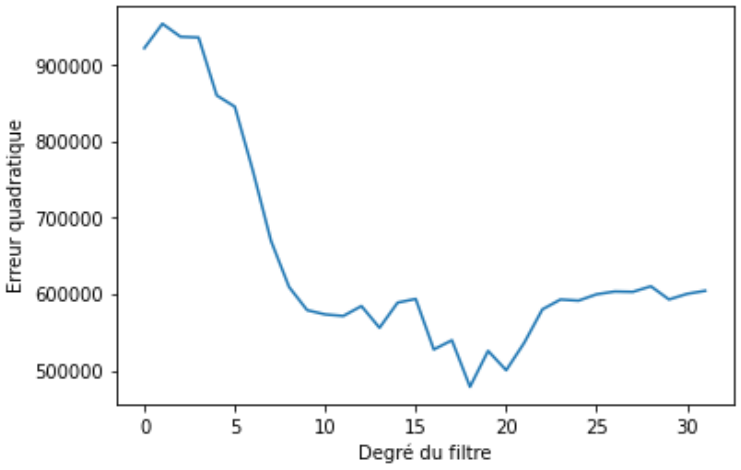
En revanche, nous avons la main sur le second hyper-paramètre qui est le degré du filtre RIF. Se pose alors la question, comment bien le choisir ? Et bien il faut tester notre filtre avec différentes valeurs de degré et voir comment évolue l'erreur quadratique. Lorsque celle-ci ne diminue plus significativement ou alors augmente lorsqu'on croit le degré alors on a trouvé le bon degré du filtre.

Voici ci-dessous, les trois graphiques obtenues. Le code nécessaire à l'obtention de ces graphiques est disponible en annexes.

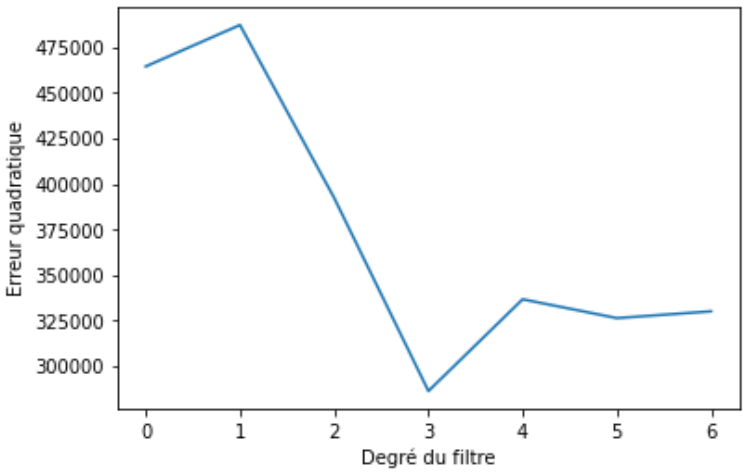
a. Quotidien : degré optimal égal à 43



b. Hebdomadaire : degré optimal égal à 18



c. Mensuel : degré optimal égal à 3

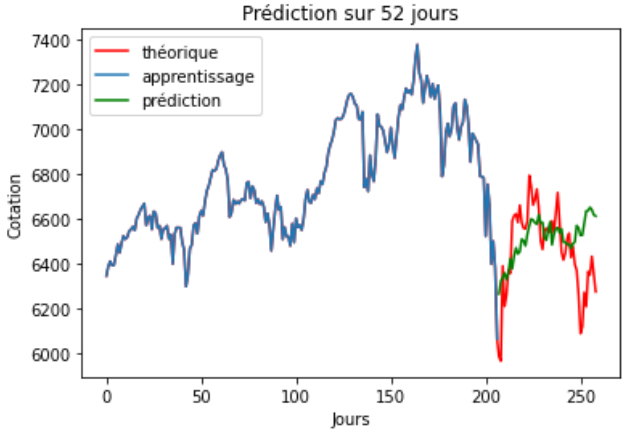


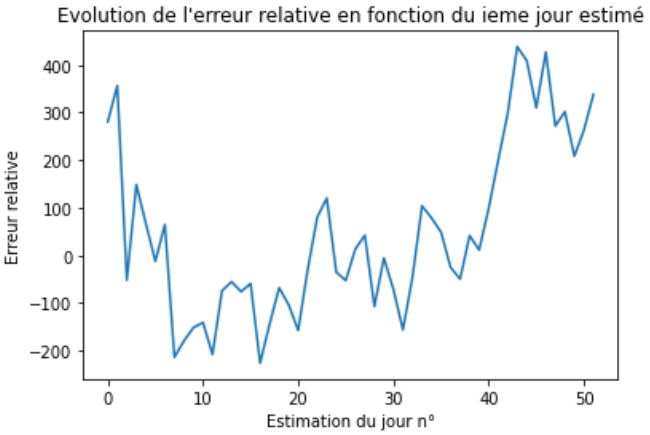
A travers ces trois graphiques on constate la même tendance, une forte diminution de l'erreur quadratique en première partie jusqu'à la valeur optimale du degré puis une remontée. Comment expliquer ce phénomène ? En effet on pourrait se dire que plus le degré du filtre est important plus l'erreur quadratique de la prédiction serait meilleure. C'est le cas jusqu'à la valeur optimale du degré puis on passe dans ce que l'on appelle le sur-apprentissage. Le fait de donner trop de données en entrée de notre filtre a tendance à créer du bruit.

6. Résultats et interprétations

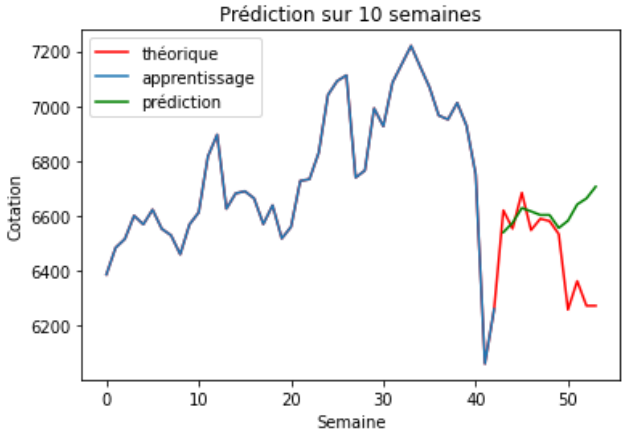
Ci-dessous voici les résultats obtenus à l'aide du modèle de Wiener pour les 3 différentes temporalités. Le premier graphique nous montre les 2 tracés : théorique, ce qui s'est réellement passé et notre prédiction. Le second graphique s'intéresse à la différence analytique entre ces deux tracés.

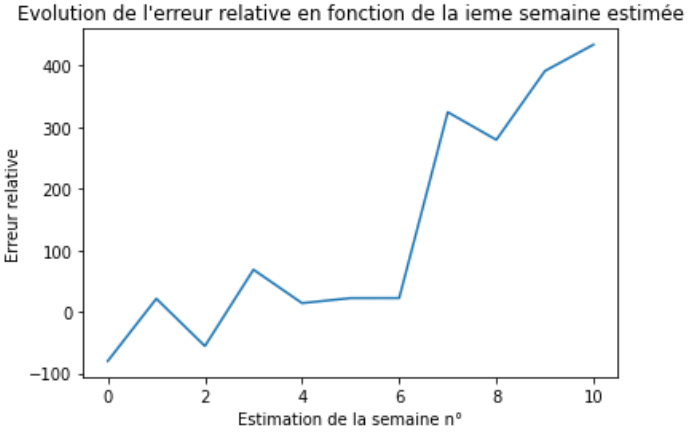
a. Quotidien



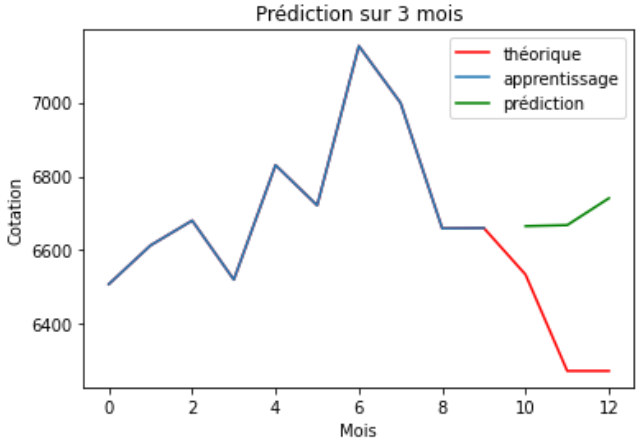


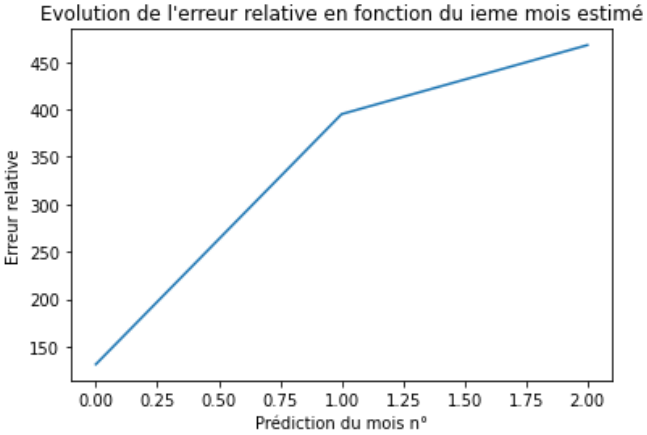
b. Hebdomadaire





c. Mensuel





d. Interprétations

Avant de commencer à entrer dans les détails, nous pouvons constater que la prédiction est satisfaisante sur une durée d'environ 1 mois pour l'évolution de l'indice de manière quotidienne et hebdomadaire. Dès cette première constatation on peut rejeter la troisième prédiction. En effet, les résultats obtenus divergent par rapport à ce qu'il s'est réellement passé. Ce qu'il faut bien comprendre, c'est qu'on ne rejette pas une temporalité (mensuelle) mais un manque de données pour permettre le bon apprentissage du filtre de Wiener.

Les deux premières prédictions nous donnent sensiblement les mêmes résultats. Gardons en tête que nous travaillons sur la prédiction d'un indice boursier qui est sensible à des événements non prévisibles: crise pétrolière, pandémie, guerre civile… Par conséquent, cette prédiction nous donne seulement une tendance de notre indice: haussière ou baissière. Dans ce contexte, notre prédiction sur 1 mois est tout à fait convaincante avec une tendance haussière prédite. L'avantage d'avoir choisi de prédire un indice plutôt que le cours d'une action d'entreprise privée, c'est le fait d'être moins soumis à des perturbations aléatoires. L'indice permet de lisser tous ces événements et est bénéfique à travers l'utilisation du filtre de Wiener.

Les résultats que nous avons obtenus proviennent des données d'apprentissage. Il est possible que les erreurs que nous avons obtenues viennent du fait que nous n'avons pas un signal stationnaire d'ordre 2 en entrée. Pour se faire, il aurait fallu trouver une tendance à nos données d'entrée puis supprimer cette tendance. Ce dernier point n'a pas été réalisé puisque nous pouvions pas trouver une tendance à nos données… Nous aurions très bien pu restreindre nos données afin d'avoir une tendance claire mais l'objectif ici était de prédire le CAC 40 de demain.

Enfin comme expliqué précédemment, la bourse n'est pas une science exacte et dépend de nombreux facteurs que nous ne pouvons pas prévoir. De ce fait, la prédiction boursière se restreint à la prédiction de tendance. Pour obtenir cette tendance, la façon la plus sûre est de croiser les indicateurs techniques. La convergence des résultats de plusieurs indicateurs techniques comme notre prévision à l'aide du modèle de Wiener permet d'avoir un résultat qui tend vers la réalité.

7. Bibliographie

a. Etude bibliographique du modèle

https://fr.wikipedia.org/wiki/Filtre_de_Wiener

COSTANZA, María Belén. Wiener Filter para mapas del Fondo Cósmico de Radiación utilizando redes neuronales. 2022. Thèse de doctorat. Universidad Nacional de La Plata.
http://sedici.unlp.edu.ar/handle/10915/136352

ZHENG, Cong, CHENG, Cheng, et SUN, Qiyu. Wiener filters on graphs and distributed polynomial approximation algorithms. arXiv preprint arXiv:2205.04019, 2022.
https://arxiv.org/abs/2205.04019

b. Etude bibliographique sur l'application

lafinancepourtous, 2022, Le CAC 40, qu'est ce que c'est ?
https://www.lafinancepourtous.com/decryptages/marches-financiers/acteurs-de-la-finance/bourse/le-cac-40/le-cac-40-qu-est-ce-que-c-est/

TRAITEMENT DU SIGNAL, Systèmes Dynamiques auto-adaptatifs et Modélisation d'indices Boursiers, 2004
https://www.univers-bourse.com/forum/m%C3%A9thodes-d-analyses-et-strat%C3%A9gies-de-trading/indicateurs-et-systemes/13008-traitement-du-signal-syst%C3%83%C2%A8mes-dynamiques-auto-ada

Bair J, Haesbroeck G., Modèles chaotiques en économie, Tangente sup., Prévoir pour décider, POLE, 2012, Prévoir l'évolution de la bourse ?
https://www.reflexions.uliege.be/cms/c_42257/fr/prevoir-l-evolution-de-la-bourse

Annexes

a. Implémentation

- Définition des fonctions utiles:

```
def predicteurlineaire(signal, degre):
    n = signal.shape[0]
    signal_CR = (signal - np.mean(signal))/np.var(signal)
    signal_autocorr = np.correlate(signal_CR, signal_CR, "full")
    autocorr = signal_autocorr[n-1:]
    mat_toeplitz = scl.toeplitz(autocorr[0:degre])
    b = -autocorr[1:degre+1].reshape(degre,1)
    return scl.solve(mat_toeplitz,b)

xcorr = lambda X,Y:np.correlate(X,Y,mode='full')
def predicteurlineaire_step(x,step,p):
    n = x.shape[0]
    RXX = xcorr(x,x)[n-1:]
    return scl.solve_toeplitz(RXX[:p], RXX[step:step+p]) #on prend
jusque p car on est en python, en théorie on prend jusqu'à p-1

def degre_max(taille_echantillon, step_max):
    return taille_echantillon - step_max
```

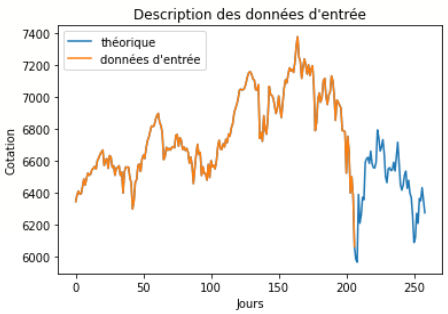
- Chargement des données:

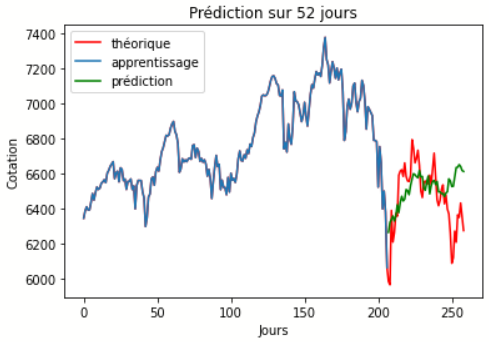
```
nRowsRead1 = 259
CAC_quotidien = pd.read_csv('CAC_quotidien.csv', delimiter=',',
nrows = nRowsRead1)
CAC_hebdo = pd.read_csv('CAC_hebdo.csv', delimiter=',', nrows =
nRowsRead1)
CAC_mensuel = pd.read_csv('CAC_mensuel.csv', delimiter=',', nrows =
nRowsRead1)
```

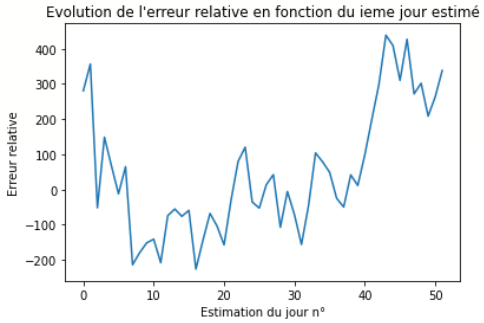
- Prédiction quotidienne:

```
p = 43 ##estimé par le code sur l'optimisation de l'hyper-paramètre

taille_totale = 259
taille_echantillon = int(np.round(taille_totale*0.8))
size = int(np.round(taille_totale*0.2))
signal = CAC_quotidien["Close"]
X = signal[:taille_echantillon]
X_theorique = signal[:taille_totale]
```

```
plt.figure()
plt.title("Description des données d'entrée")
plt.xlabel("Jours")
plt.ylabel("Cotation")
plt.plot(X_theorique, label="théorique")
plt.plot(X, label="données d'entrée")
plt.legend()
plt.show()

mean = np.mean(X)
var = np.var(X)
X_CR = (X - mean)/var

estimation = np.zeros(size)
for i in range(1, size+1):
    coeff = predicteurlineaire_step(X_CR, i, p)
    estimation[i-1] = np.inner(np.flip(X_CR[-p:]),coeff.reshape(p))
    estimation[i-1] = estimation[i-1]*var + mean

plt.figure()
plt.title("Prédiction sur 52 jours")
plt.xlabel("Jours")
plt.ylabel("Cotation")
plt.plot(X_theorique, 'r', label="théorique")
plt.plot(X, label="apprentissage")
plt.plot(np.arange(taille_echantillon,taille_echantillon+size),estim
ation,'g', label="prédiction")
plt.legend()
plt.show()

plt.figure()
plt.title("Evolution de l'erreur relative en fonction du ieme jour
estimé")
plt.xlabel("Estimation du jour n°")
plt.ylabel("Erreur relative")
plt.plot(np.arange(size),(estimation-X_theorique[taille_echantillon:
size+taille_echantillon]))
plt.show()
```



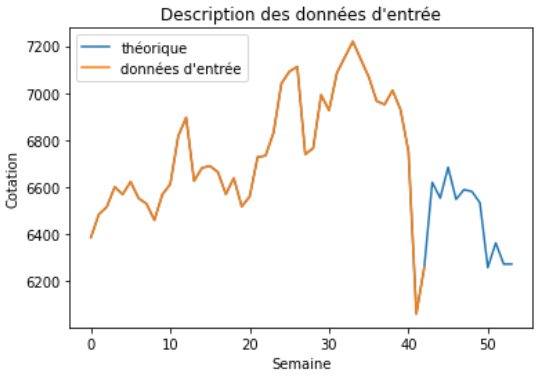


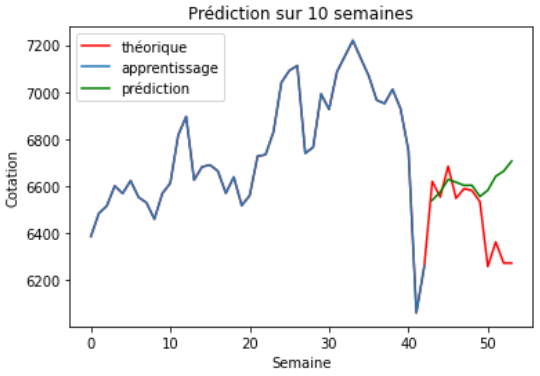


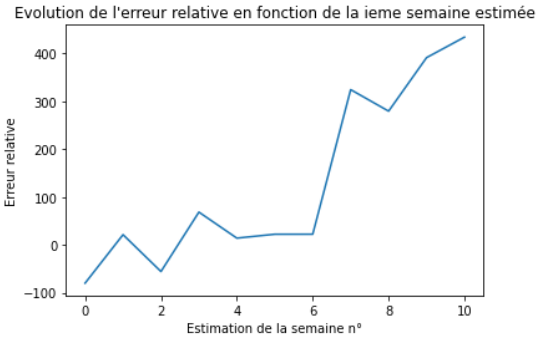
- Prédiction hebdomadaire:

```
p = 18  ##estimé par le code sur l'optimisation de l'hyper-paramètre

taille_totale = 54
taille_echantillon = int(np.round(taille_totale*0.8))
size = int(np.round(taille_totale*0.2))
signal = CAC_hebdo["Close"]
X = signal[:taille_echantillon]
X_theorique = signal[:taille_totale]

plt.figure()
plt.title("Description des données d'entrée")
plt.xlabel("Semaine")
plt.ylabel("Cotation")
plt.plot(X_theorique, label="théorique")
plt.plot(X, label="données d'entrée")
plt.legend()
plt.show()

mean = np.mean(X)
var = np.var(X)
X_CR = (X - mean)/var
```

```
estimation = np.zeros(size)
for i in range(1, size+1):
    coeff = predicteurlineaire_step(X_CR, i, p)
    estimation[i-1] = np.inner(np.flip(X_CR[-p:]),coeff.reshape(p))
    estimation[i-1] = estimation[i-1]*var + mean

plt.figure()
plt.title("Prédiction sur 10 semaines")
plt.xlabel("Semaine")
plt.ylabel("Cotation")
plt.plot(X_theorique, 'r', label="théorique")
plt.plot(X, label="apprentissage")
plt.plot(np.arange(taille_echantillon,taille_echantillon+size),estim
ation,'g', label="prédiction")
plt.legend()
plt.show()

plt.figure()
plt.title("Evolution de l'erreur relative en fonction de la ieme
semaine estimée")
plt.xlabel("Estimation de la semaine n°")
plt.ylabel("Erreur relative")
plt.plot(np.arange(size),(estimation-X_theorique[taille_echantillon:
size+taille_echantillon]))
plt.show()
```



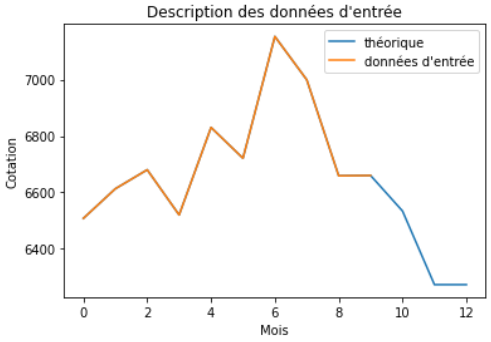


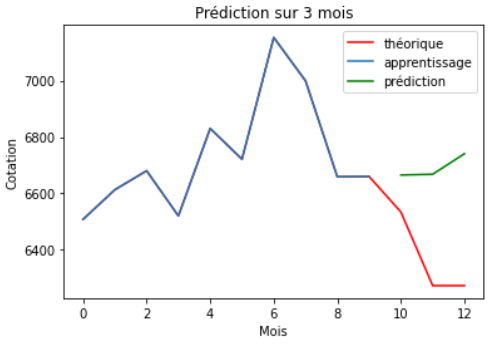


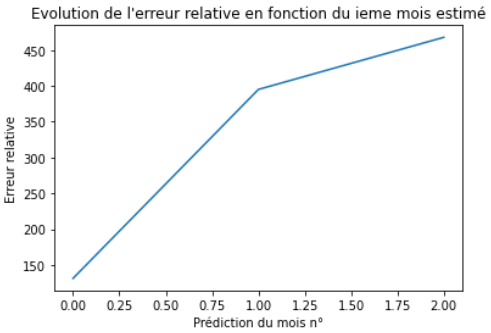
- Prédiction mensuelle:

```
p = 3  ##estimé par le code sur l'optimisation de l'hyper-paramètre

taille_totale = 13
taille_echantillon = int(np.round(taille_totale*0.8))
size = int(np.round(taille_totale*0.2))
signal = CAC_mensuel["Close"]
X = signal[:taille_echantillon]
X_theorique = signal[:taille_totale]

plt.figure()
plt.title("Description des données d'entrée")
plt.xlabel("Mois")
plt.ylabel("Cotation")
plt.plot(X_theorique, label="théorique")
plt.plot(X, label="données d'entrée")
plt.legend()
plt.show()

mean = np.mean(X)
var = np.var(X)
X_CR = (X - mean)/var
estimation = np.zeros(size)

for i in range(1, size+1):
    coeff = predicteurlineaire_step(X_CR, i, p)
    estimation[i-1] = np.inner(np.flip(X_CR[-p:]),coeff.reshape(p))
    estimation[i-1] = estimation[i-1]*var + mean

plt.figure()
plt.title("Prédiction sur 3 mois")
plt.xlabel("Mois")
```

```
plt.ylabel("Cotation")
plt.plot(X_theorique, 'r', label="théorique")
plt.plot(X, label="apprentissage")
plt.plot(np.arange(taille_echantillon,taille_echantillon+size),estim
ation,'g', label="prédiction")
plt.legend()
plt.show()

plt.figure()
plt.title("Evolution de l'erreur relative en fonction du ieme mois
estimé")
plt.xlabel("Prédiction du mois n°")
plt.ylabel("Erreur relative")
plt.plot(np.arange(size),(estimation-X_theorique[taille_echantillon:
size+taille_echantillon]))
plt.show()
```







b. Choix des hyper-paramètres

1. Quotidien

```
taille_totale = 259
taille_echantillon = int(np.round(taille_totale*0.8))
size = int(np.round(taille_totale*0.2))
signal = CAC_quotidien["Close"]
X = signal[:taille_echantillon]
X_theorique = signal[:taille_totale]

estimation = np.zeros(size)
mean = np.mean(X)
var = np.var(X)
X_CR = (X - mean)/var

p_max = degre_max(taille_echantillon,size)
erreur_quad = np.zeros(p_max)

for p in range(1,p_max+1):
   for i in range(1, size+1):
       coeff = predicteurlineaire_step(X_CR, i, p)
       estimation[i-1] =
np.inner(np.flip(X_CR[-p:]),coeff.reshape(p))
       estimation[i-1] = estimation[i-1]*var + mean

   erreur_quad[p-1] =
pd.Series.sum((estimation-X_theorique[taille_echantillon:size+taille
_echantillon])**2)

plt.figure()
plt.xlabel("Degré du filtre")
plt.ylabel("Erreur quadratique")
plt.plot(erreur_quad)
plt.show()

print("Le degré optimale du filtre est :",np.argmin(erreur_quad))
```

2. Hebdomadaire

```
taille_totale = 54
taille_echantillon = int(np.round(taille_totale*0.8))
size = int(np.round(taille_totale*0.2))
signal = CAC_hebdo["Close"]
X = signal[:taille_echantillon]
```

```
X_theorique = signal[:taille_totale]

estimation = np.zeros(size)
mean = np.mean(X)
var = np.var(X)
X_CR = (X - mean)/var

p_max = degre_max(taille_echantillon,size)
erreur_quad = np.zeros(p_max)

for p in range(1,p_max+1):
   for i in range(1, size+1):
       coeff = predicteurlineaire_step(X_CR, i, p)
       estimation[i-1] =
np.inner(np.flip(X_CR[-p:]),coeff.reshape(p))
       estimation[i-1] = estimation[i-1]*var + mean

   erreur_quad[p-1] =
pd.Series.sum((estimation-X_theorique[taille_echantillon:size+taille
_echantillon])**2)

plt.figure()
plt.xlabel("Degré du filtre")
plt.ylabel("Erreur quadratique")
plt.plot(erreur_quad)
plt.show()

print("Le degré optimale du filre est :",np.argmin(erreur_quad))
```

3. Mensuel

```
taille_totale = 13
taille_echantillon = int(np.round(taille_totale*0.8))
size = int(np.round(taille_totale*0.2))
signal = CAC_mensuel["Close"]
X = signal[:taille_echantillon]
X_theorique = signal[:taille_totale]


estimation = np.zeros(size)
mean = np.mean(X)
var = np.var(X)
```

```
X_CR = (X - mean)/var

p_max = degre_max(taille_echantillon,size)
erreur_quad = np.zeros(p_max)

for p in range(1,p_max+1):
   for i in range(1, size+1):
       coeff = predicteurlineaire_step(X_CR, i, p)
       estimation[i-1] =
np.inner(np.flip(X_CR[-p:]),coeff.reshape(p))
       estimation[i-1] = estimation[i-1]*var + mean

   erreur_quad[p-1] =
pd.Series.sum((estimation-X_theorique[taille_echantillon:size+taille
_echantillon])**2)

plt.figure()
plt.xlabel("Degré du filtre")
plt.ylabel("Erreur quadratique")
plt.plot(erreur_quad)
plt.show()

print("Le degré optimale du filre est :",np.argmin(erreur_quad))
```