



المندوبية السامية للتخطيط  
HAUT-COMMISSARIAT AU PLAN

ROYAUME DU MAROC  
\*\_\*\_\*\_\*\_\*  
HAUT COMMISSARIAT AU PLAN  
\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*  
INSTITUT NATIONAL  
DE STATISTIQUE ET D'ECONOMIE APPLIQUEE



**INSEA**

## Projet de Fin d'Etudes

\*\*\*\*\*

**Modélisation de la structure par terme des taux  
d'intérêt via le modèle de Nelson Siegel Dynamique et  
le réseau de neurone LSTM : Cas du marché marocain**

Préparé par : *Mme Salma BELABBES*  
*M Yahya FIRDAOUSSI*

Sous la direction de : *M. Yassine EL QALLI (INSEA)*  
*M. Larbi BENSLAMA (CDG Capital)*

*Soutenu publiquement comme exigence partielle en vue de l'obtention du*

**Diplôme d'Ingénieur d'Etat**

**Filière : Actuariat-Finance**

Devant le jury composé de :

- *M. Yassine EL QALLI (INSEA)*
- *M. Khalil SAID (INSEA)*
- *M. Larbi BENSLAMA (CDG Capital)*



# Résumé

De mois en mois, voire de jours en jours, les rendements des actifs à revenu fixe tels que les bons du Trésor varient en fonction de leur échéance et la relation entre rendement et maturité varie à son tour avec le temps. La structure par terme des taux d'intérêt montre comment les taux d'intérêt varient en fonction de la durée des titres d'endettement auxquels ils se rapportent. Les utilisations de la structure par terme des taux d'intérêt sont multiples, plus particulièrement pour les gérants d'actifs à revenu fixe ou variable. Ces gestionnaires ajustent leurs portefeuilles en fonction de leur stratégie et des caractéristiques des titres qui les composent. L'analyse de la structure des taux d'intérêt, c'est-à-dire la relation entre les taux à court et à long terme, se fait généralement par la courbe des taux qui constitue un outil plus précieux d'aide à la décision pour les opérateurs des marchés financiers. Cette courbe de rendement joue un rôle central pour les décisions financières.

De ce fait, le modèle Nelson-Siegel et son extension dynamique, qui est un modèle à trois facteurs, est largement utilisé dans la pratique pour ajuster la structure par terme des taux d'intérêt et de faire des prévisions en raison de la facilité de linéarisation du modèle. Ainsi, à travers ce projet, nous avons essayé d'effectuer une modélisation desdits facteurs avec deux grandes approches à savoir une approche statistique et une nouvelle approche qui utilise des réseaux de neurones récurrents artificiels (RNN) qui est une discipline du Machine Learning ayant pour objectif l'explication de la dynamique globale de la courbe des taux.

Mots clés : courbe des taux, Nelson Siegel, VAR, Filtre de Kalman, VECM, RNN, LSTM.

# Abstract

From month to month, or even days to days, yields on fixed income assets such as treasury bills vary according to their maturity and the relationship between yield and maturity in turn varies over time. The term structure of interest rates shows how interest rates vary according to the duration of the debt securities to which they relate. There are many uses of the term structure of interest rates, particularly for fixed and variable income asset managers. These managers adjust their portfolios according to their strategy and the characteristics of their securities. The analysis of the interest rate structure, i.e. the relationship between short-term and long-term rates, is usually done through the yield curve, which is a more valuable tool for assisting decision for financial market operators. This yield curve plays a central role for financial decisions.

As a result, the Nelson-Siegel model and its dynamic extension, which is a three-factor model, is widely used in practice to adjust the term structure of interest rates and to make forecasts because of the ease of linearization of the model. Thus, through this project, we tried to carry out a modeling of these factors with two main approaches namely a statistical approach and a new approach that uses networks of artificial recurrent neurons (RNN) which is a discipline of Machine Learning with the aim of objective the explanation of the overall dynamics of the yield curve.

Keywords: yield curve, Nelson Siegel, VAR, Kalman filter, VECM, RNN, LSTM.

# Dédicace

*Je dédie ce projet de fin d'études à mes parents pour leur amour inestimable, leur soutien, leurs sacrifices et toutes les valeurs éthiques qu'ils ont su m'inculquer. A mes deux frères Othmane et Mehdi pour leur amour et leur présence malgré la distance qui nous sépare. A ma famille, mes amis ainsi que les autres stagiaires avec qui j'ai noué de vraies relations.*

*A mon cher binôme, qui sans lui ce travail n'aurait pas vu le jour.*

# Dédicace

*A celle qui m'a donné naissance,  
A celle qui a attendu avec impatience,  
Les fruits de ce long parcours d'endurance,  
A ma mère*

*A celui qui m'a inculqué de bonnes valeurs,  
A celui qui m'a donné le sens de l'honneur,  
A celui qui m'a toujours orienté dans la vie,  
A celui qui a fait de moi, la personne que je suis,  
A mon père*

*A toute ma famille  
A mes professeurs qui m'ont enseigné  
A ma chère binôme que je le dois un respect énorme  
A notre encadrant qui n'hésite pas à nous conseiller  
Un grand Merci à vous !*

*Yahya*

# Remerciements

Au terme de ce travail, nous tenons à adresser tous nos remerciements aux personnes ayant contribué de près ou de loin à la réussite de notre projet de fin d'études.

Ainsi nous tenons à remercier le professeur M. El QALLI pour son encadrement, ses conseils judicieux, et pour toutes les connaissances qu'il nous a transmises.

Nous ne manquerions également d'exprimer nos remerciements à toute l'équipe de la salle de marché de CDG Capital pour leur contribution dans ce travail, et surtout notre encadrant M. BENSLAMA pour son accueil, sa confiance, son aide et ses conseils.

Nous remercions également M. SAID qui nous a honoré en acceptant de juger notre travail.

Nos vifs remerciements vont finalement à nos parents, proches et amis ayant contribué de près ou de loin à l'élaboration de ce modeste travail.

# Table des matières

Résumé.....	3
Remerciements.....	7
Liste des figures .....	10
Liste des tableaux.....	11
Liste des abréviations.....	12
Introduction.....	13
CHAPITRE 1.....	15
MISE EN CONTEXTE ET NOTIONS DE BASE .....	15
I. Présentation de l'organisme d'accueil .....	16
II. Notion de taux d'intérêt .....	19
II.1 Définition.....	19
II.2 Types des taux d'intérêt.....	19
II.3 Le marché obligataire .....	20
II.4 Les mesures analytiques du risque de taux .....	21
III. Modélisation de la courbe des taux .....	23
III.1 La diversité des courbes de taux .....	23
III.2 Technique de l'interpolation de la courbe zéro-coupon : .....	24
III.3 Construction de la courbe zéro-coupon par la méthode de Bootstrapping : .....	25
III.4 Modèles des taux d'intérêts.....	27
CHAPITRE 2.....	29
Modélisation de la courbe des taux via le modèle de Diebold Li : Modèle VAR.....	29
I. Modèle de Nelson & Siegel Statique .....	30
II. Modèle de Diebold & Li .....	31
III. Méthodologie.....	34
III.1 Procédure a deux étapes .....	34
III.2 Filtre de Kalman.....	35
IV. Application .....	36
IV.1 Description du Dataset .....	36
.....	36
IV.2 Application.....	38
V. Résultats et comparaison des paramètres obtenus.....	44
CHAPITRE 3.....	49
Modélisation de la courbe des taux via le modèle de Nelson Siegel Dynamique : Modèle VECM .....	49
I. Théorie du modèle a correction d'erreurs .....	50
II. Application.....	54
CHAPITRE 4.....	61
Modélisation de la courbe des taux via le modèle NSD : réseau de neurone LSTM .....	61

I-Introduction .....	62
I-1 Historique .....	62
I-2 Définition .....	62
II-Modélisation des réseaux de neurones .....	63
II-1-Structure d'un neurone.....	63
II-2 Modèle des réseaux de neurones .....	67
III-Types de réseaux de neurones.....	67
III-1 Réseaux de neurones « feed-forward » .....	68
III-2 Réseaux de neurones récurrents (RNN).....	68
III-3 LSTM.....	68
IV- Architecture LSTM pour le modèle de Nelson-Siegel .....	73
V- Application et Résultats : .....	75
VI. Comparaison entre VECM et LSTM : .....	77
Conclusion .....	79
Bibliographie.....	80
Annexe A : Filtre de Kalman .....	81
Annexe B : Code LSTM sous Python .....	84

# Liste des figures

Figure 1: Courbe des taux zéro-coupon .....	24
Figure 2: Dynamique des paramètres de Diebold & Li .....	33
Figure 3: Evolution des taux marocains depuis 2004 .....	36
Figure 4: Courbe des taux moyenne .....	40
Figure 5: Courbe des taux relative à la date 18/02/2004 .....	41
Figure 6: Courbe des taux relative à la date 30/01/2005 .....	41
Figure 7: Courbe des taux relative à la date 23/12/2008 .....	41
Figure 8: Courbe des taux relative à la date 01/06/2017 .....	41
Figure 9: Comparaison entre la matrice de transition d'état des deux approches .....	44
Figure 10: Comparaison entre la matrice de covariance des deux approches .....	45
Figure 11: Comparaison entre la moyenne des facteurs des deux approches .....	45
Figure 12: Le facteur niveau pour les deux modèles .....	46
Figure 13: Le facteur pente pour les deux modèles .....	46
Figure 14: Le facteur courbure pour les deux modèles .....	47
Figure 15: Test de stationnarité sur les séries .....	55
Figure 16: Test de stationnarité sur les séries différenciées .....	55
Figure 17: Détermination du nombre de retards .....	56
Figure 18: Test de cointégration par la méthode de Johanson .....	57
Figure 19: Estimation du modèles VECM .....	57
Figure 20: Les deux équations de cointégration .....	58
Figure 21: Test d'autocorrélation des résidus .....	58
Figure 22: Test d'homoscédasticité .....	58
Figure 23: Structure d'un neurone artificiel j .....	64
Figure 24: Exemples de fonctions de transfert .....	66
Figure 25: De gauche à droite, la fonction Sigmoïde, la fonction Tangente hyperbolique et la fonction Heaviside .....	66
Figure 26: Un réseau de neurones à 2 couches .....	67
Figure 27: Architecture des RNN .....	68
Figure 28: Cellules du réseau LSTM aux pas de temps t-1, t, t+1 .....	70
Figure 29: Tracé du facteur niveau .....	76
Figure 30: Tracé du facteur pente .....	76
Figure 31: Tracé du facteur courbure .....	77

# Liste des tableaux

Tableau 1:Statistiques Descriptives du Dataset .....	37
Tableau 2:Statistiques des résidus d'estimation des taux zéro coupon pour les deux modèles .....	39
Tableau 3:Ratio des racines carrées de l'erreur quadratique moyenne (RMSE).....	40
Tableau 4:Comparaison entre les trois modèles : DNS-FK, VAR, VECM .....	59

# Liste des abréviations

<b>BAM</b>	Bank Al Maghreb
<b>CDG</b>	Caisse de Dépôt et de Gestion
<b>FK</b>	Filtre de Kalman
<b>LSTM</b>	Long Short Time Memory
<b>NS</b>	Nelson Siegel
<b>NSD</b>	Nelson Siegel Dynamique
<b>RMSE</b>	Root Mean Square Error
<b>TMM</b>	Taux du Marché Monétaire
<b>VAR</b>	Vector Auto Regressif
<b>VECM</b>	Vector Error Correction Model

# Introduction

En raison de l'amélioration de la trésorerie de l'Etat Marocain, le Trésor a assuré la couverture de son besoin de financement par des recours au marché obligataire. La création en 1988 d'un marché des bons du Trésor émis par adjudication avait pour objectif d'instituer un marché actif des valeurs du Trésor où les volumes et les taux seraient déterminés par les règles du marché.

De ce fait, l'étude de ce marché et plus précisément la courbe des taux d'intérêt est devenue incontournable pour chaque gestionnaire de portefeuille opérant sur les salles de marchés.

La littérature de la finance a consacré une large partie de ses études à la courbe des taux d'intérêt. En effet, l'évolution des taux d'intérêt est depuis des années au centre des préoccupations dans les marchés financiers pour l'évaluation et la couverture des titres financiers. Ainsi, les bonnes estimations de la structure par terme des taux d'intérêt revêtent une importance capitale pour les investisseurs et les décideurs. Une multitude de méthodes ont été critiquées pour leurs propriétés économiques indésirables, Nelson & Siegel (1987) a donc suggéré des courbes paramétriques suffisamment souples pour décrire toute une famille de formes de structure de terme observées. Ce modèle est parcimonieux et cohérent avec une interprétation factorielle de la structure par termes (Litterman (1991)) et a été largement utilisé dans les milieux universitaires et dans la pratique. Diebold et Li (2006), se sont basés sur le modèle de Nelson & Siegel qui décrit la structure par terme des taux à un instant précis dans le temps et l'ont transformé en un modèle du mouvement de la structure à terme au fil du temps afin de faire des prévisions de la courbe des taux.

Avec la disponibilité croissante des données, des réseaux de neurones, tels que Long Short Time Memory (LSTM), peuvent être utilisés pour prédire les paramètres de la courbe des taux en tant qu'alternative aux modèles cités précédemment.

L'objectif du présent rapport est de faire une comparaison entre les différentes méthodes d'estimation du modèle de Nelson & Siegel Dynamique (DNS). Pour ce faire, nous étudierons dans un premier temps le modèle Diebold et Li puis l'utilisation du filtre

de Kalman. Nous examinerons par la suite le modèle de Nelson & Siegel Dynamique en optant pour le modèle à correction d'erreur (VECM). Enfin, nous utiliserons une architecture de réseau de neurones récurrents artificiels (RNN).

# **CHAPITRE 1**

## **MISE EN CONTEXTE ET NOTIONS DE BASE**

## **I. Présentation de l'organisme d'accueil**

Filiale à cent pour cent de la Caisse de dépôt et de gestion, CDG Capital a été créée en 2006 suivant l'Arrêté numéro 284-06 du 11 moharrem 1427 (10 février 2006) dans le cadre de la stratégie de filialisation opérée par la CDG, et qui avait pour objectif d'offrir un cadre juridique et légal, ainsi qu'un environnement plus adéquat aux activités financières de la CDG.

Cette jeune banque d'investissement est un acteur majeur du paysage financier marocain et un opérateur de premier plan sur les trois groupes du métier de la banque d'investissement :

- Asset Management ;
- Corporate and Investment Banking ;
- Marché des capitaux.

### **Présentation de la salle de marchés**

Le présent projet s'est déroulé au sein de la salle de marchés de la CDG Capital. Cette salle est un lieu qui regroupe différents services spécialisés permettant à la banque d'intervenir sur les marchés de capitaux locaux et internationaux.

La salle de marchés est organisée en desks, spécialisés par produits ou segment de marché (actions, taux, options, etc.), qui partagent un large open space.

La salle de marchés de la CDG Capital est organisée sous trois pôles distincts :

#### **Front Office :**

Le front office constitue littéralement l'interface de la banque avec le marché. En effet, il centralise et traite tous les besoins de la salle de marchés et de ses clients en termes d'investissement, gestion de portefeuille et trading. Le front office de La CDG Capital se compose de desks suivants :

- **Desk taux**

Le desk taux est en charge, d'une part, de la gestion du portefeuille obligataire de la banque et, d'autre part, de l'intermédiation entre les différents intervenants sur le marché, sur le compartiment primaire mais également sur le compartiment secondaire.

- **Desk actions**

Le desk actions assure les activités de trading et d'arbitrage pour compte propre sur les actions. Le desk prend des positions pour profiter des opportunités à des horizons relativement courts. Pour assurer sa fonction, le desk se base sur l'analyse fondamentale et technique publiée par le desk Analyse et Recherche ainsi que celles de diverses sociétés de bourses sur le marché.

- **Desk Recherche et Analyse**

Parallèlement aux solutions financières proposées, le desk Recherche et Analyse propose une analyse (quotidienne, hebdomadaire, etc.) des marchés financiers. Afin d'assurer un meilleur accompagnement des investissements, l'équipe du desk Recherche et Analyse réalise des notes de recherches relatives tant au volet macroéconomique qu'aux marchés de taux et de change.

- **Desk Monétaire**

Le desk monétaire constitue l'interface du groupe CDG sur le marché monétaire. Il assure la gestion de la position monétaire de la banque CDG Capital et celle de la maison mère CDG. Ce desk offre à ses clients des solutions de placements des excédents de trésorerie et de couverture des besoins de financement et l'accès aux conditions de financement du marché interbancaire (prêt/emprunts).

- **Desk Change**

Le desk change traite les opérations sur devises aussi bien pour le compte de la CDG Capital que pour le compte de sa clientèle.

- **Desk produits dérivés et structurés**

Il a pour mission de fournir des cotations sur les produits dérivés, la prise en charge des opérations de couverture des différents portefeuilles ainsi que la mise en place de nouveaux produits financiers répondant aux besoins d'investissement ou de couverture de ses différents clients (OPCVM, Assurance, Caisse de retraite, etc.).

### **Middle Office :**

Le middle-office assume les tâches suivantes :

- La vérification des négociations effectuées par le Front Office : celles-ci doivent être conformes à la réglementation et aux bonnes pratiques définies en interne (exemples : abus de marché, délit d'initié, lutte contre le blanchiment, etc.) ;
- Le contrôle de non-dépassement des limites d'engagement par contrepartie ;
- Le contrôle de non-dépassement des limites de position, par trader, mais aussi par segment de marché.
- L'envoi de confirmation, à la contrepartie, par fax, par mail, ou via un logiciel dédié. La confirmation se réfère normalement à un contrat-cadre, qui dépend du type d'instrument traité ;

### **Back Office :**

Le back-office s'occupe de toute la gestion post-négociation des opérations financières. Cette entité assure les fonctions suivantes :

- Vérification (cohérence, conformité) des opérations enregistrées par le Front Office ;
- Contrôle de provision (espèces ou titres) ;
- Vérification des encaissements ;
- Gestion de la vie des contrats, suivi des échéances ;
- Facturation des commissions ;
- Comptabilisation ;
- Reporting, notamment à vocation réglementaire.

## **II. Notion de taux d'intérêt**

### **II.1 Définition**

Un taux d'intérêt est un prix qui s'applique à une somme d'argent prêtée ou empruntée. Globalement, les taux d'intérêt se forment sur un marché (le marché des capitaux), à partir de la rencontre d'une offre et d'une demande de capitaux. Si à un moment donné, les agents économiques ont une épargne abondante et qu'à l'inverse peu d'agents économiques souhaitent emprunter, on verra le taux d'intérêt baisser. En sens inverse, il augmentera, si les besoins de financement des agents sont importants et si les capacités de financement (l'épargne) sont faibles.

### **II.2 Types des taux d'intérêt**

#### **II.2.1. Taux de rendement actuariel**

Le taux de rendement actuariel est le taux qui permet d'égaliser la valeur actuelle de l'obligation avec la somme des flux futurs perçus, c'est à dire les coupons et le prix de remboursement à l'échéance du titre. Autrement dit, il est équivalent au taux d'intérêt que percevrait un investisseur qui détiendrait l'obligation jusqu'à son terme. Dans le langage courant, il correspond au rendement de l'obligation.

Pour calculer le taux de rendement actuariel, il suffit d'appliquer la formule suivante :

$$i = \left( \frac{VF}{VA} \right)^{\frac{1}{n}} - 1$$

où :

VF = valeur future

VA = valeur Actuelle

n = nombre d'années

À noter que lorsque le prix de l'obligation (la valeur actuelle) augmente, le taux de rendement actuariel diminue et inversement.

#### **II.2.2. Taux du marché monétaire**

Le taux du marché monétaire (TMM) correspond au taux auquel les banques s'empruntent et se prêtent de l'argent entre elles, généralement à court terme (360 jours).

Il joue un rôle essentiel dans l'activité financière et fluctue en fonction de l'inflation, du chômage, mais aussi de la conjoncture internationale.

Le taux monétaire diffère du taux de rendement actuariel en termes de durée. Le passage du taux monétaire au taux actuariel se fait de la manière suivante :

$$r_a = \left(1 - \frac{r_m * n}{360}\right)^{\frac{365}{n}} - 1$$

où :

$r_a$  = le taux actuariel

$r_m$  = le taux monétaire

n = la durée en jours (maturité correspondante au taux monétaire)

### **II.2.3. Taux zéro-coupon**

On appelle zéro-coupon de maturité T, un titre versant l'unité (1DH) à la date T, et ne donnant aucun flux entre 0 et T. La valeur de ce contrat à  $t < T$  est noté P(t,T) et il est clair que  $P(T,T) = 1$ .

Le paiement à la maturité T est connu sous le nom de la valeur principale. Ces instruments sont caractérisés par un flux déterministe, et pour cette raison ils sont connus sous le nom des instruments à revenu fixe (fixed income securities).

### **II.2.4. Taux forward**

Le taux d'intérêt forward est un taux d'intérêt qui peut être déterminé aujourd'hui pour un investissement dans une période future. Le taux forward peut être vu comme une estimation du taux future  $L(T,S)$  qui est aléatoire à l'instant t, appliquée pour une période future entre T et S.

## **II.3 Le marché obligataire**

### **II.3.1. Définition**

Le marché obligataire est le compartiment du marché financier où s'échangent les titres de créances à moyen et long terme. C'est un marché de prêts/emprunts de capitaux dont le support est constitué par les obligations.

1 <sup>er</sup> Mardi	2 <sup>ème</sup> Mardi	3 <sup>ème</sup> Mardi	4 <sup>ème</sup> Mardi
13 semaines 52 semaines 2 ans	26 semaines 52 semaines 5 ans 15 ans	13 semaines 52 semaines 2 ans	26 semaines 2 ans 10 ans 20 ans

On distingue le marché primaire, sur lequel de nouveaux titres obligataires sont émis et le marché secondaire où ces créances d'occasion s'échangent par la suite. Au Maroc, la majeure partie des obligations proposées sont des obligations d'État.

L'émission des BDT prend lieu chaque mardi, et est organisée au long du mois comme suit :

### II.3.2. Fonctionnement du marché obligataire

On parle de marché obligataire pour les dettes d'une durée supérieure à 1 an et de marché monétaire pour les dettes inférieures à cette durée. Des entreprises, collectivités locales et États s'échangent des obligations sur le marché secondaire, mais aussi des produits dérivés (swaps de taux d'intérêt, futures sur emprunts d'état, etc.). Le marché obligataire est surtout réservé aux grands investisseurs et n'est accessible aux particuliers qu'à travers des OPCVM (FCP, SICAV).

Comme celui des actions, le marché obligataire est scindé entre un marché primaire et un marché secondaire :

- Marché primaire : les échanges sur le marché primaire concernent des obligations nouvellement émises par des États, des collectivités locales, des banques ou de grandes sociétés privées qui vendent directement leurs titres aux investisseurs professionnels.
- Marché secondaire : c'est sur ce marché que les investisseurs s'échangent les obligations d'occasion. Leur cours fluctue comme celui de toutes les valeurs financières. Dans l'ensemble, les transactions s'effectuent de gré à gré. Les acheteurs ou les vendeurs interrogent des « teneurs de marché ». Ceux-ci leur transmettent des prix d'acquisition ou de cession. Les investisseurs se tournent alors vers l'intermédiaire leur faisant la meilleure offre. Un teneur, aussi appelé « market maker », propose des prix d'achat et de vente se situant dans une fourchette de prix et prélève une commission lors des transactions.

### II.4 Les mesures analytiques du risque de taux

Le taux de rendement d'une obligation suit le mouvement du niveau général des taux. Si les taux d'intérêts sur les marchés baissent, celles des obligations suivront ce mouvement, bien que à des degrés différents.

Afin de connaître le niveau à hauteur duquel une obligation en particulier est exposée au risque de taux, il existe plusieurs mesures, qui seront présentées ci-après.

#### II.4.1. La duration

La duration, appelée également duration de Macaulay, est une mesure du temps moyen pondéré jusqu'au paiement des flux générés par l'obligation (coupons et remboursement du principal), et est exprimée en années. Elle donne également une approximation de l'amplitude du changement de prix de l'obligation en pourcent pour une variation du taux de rendement de 1%, soit 100 points de base. Elle est calculée en faisant la somme des valeurs actuelles de chacun des flux futurs, multipliée par le temps restant jusqu'à sa tombée, et en divisant ensuite la valeur obtenue par le prix coupon couru inclus du titre.

Formule de calcul :

$$D = \frac{\sum_{i=1}^n t_i \times \frac{F_i}{(1+r)^{t_i}}}{\sum_{i=1}^n \frac{F_i}{(1+r)^{t_i}}}$$

#### II.4.2. La duration modifiée/ Sensibilité

La duration modifiée est une dérivée de la duration de Macaulay et est basée sur le calcul de cette dernière. Elle exprime la variation du prix de l'obligation pour une variation donnée de 1% du taux de rendement actuariel, mesurée en pourcentage du prix. Elle est définie comme suit :

$$S = \frac{1}{P} \frac{dP}{dr}$$

Étant donné que le prix d'une obligation est généralement donné par :

$$P = \sum_{i=1}^n \frac{F_i}{(1+r)^{t_i}}$$

Alors la sensibilité est égale à :

$$S = -\frac{1}{P} \sum_{i=1}^n \frac{i \times F_i}{(1+r)^{t_{i+1}}}$$

La duration s'exprime en fonction de la sensibilité comme suit :

$$S = -\frac{D}{1+r}$$

### II.4.3. La convexité

Le terme convexité décrit le fait que la courbe prix-taux d'une obligation n'est pas une ligne droite, mais une courbe. Cela signifie que la variation du prix suite à une variation donnée du taux de rendement sera d'une ampleur différente selon l'endroit auquel on se trouve sur la courbe.

Plus la durée de vie d'une obligation est longue, plus la convexité est élevée. Plus le taux de coupon d'une obligation est élevé, plus la convexité est élevée également.

La convexité est calculée à l'aide de la formule suivante :

$$C = \frac{P' + P'' - 2P}{P(\Delta y)^2}$$

où : P' : Prix correspondant au taux de rendement plus un pb

P'' : Prix correspondant au taux de rendement moins un pb

$\Delta y = 1 \text{ pb (0.01\%)}$

### II.4.4. Price Value of a Basis Point (PVBP)

La PVBP indique le changement du prix de l'obligation pour un changement d'un point de base pour son taux de rendement. A la différence de la duration modifiée, la PVBP exprime la variation du prix non pas en pourcent du prix mais en valeur monétaire.

La PVBP peut être déduite de la duration :

$$PVBP = Duration * 0.01\% * \text{Prix de l'obligation}$$

## III. Modélisation de la courbe des taux

### III.1 La diversité des courbes de taux

Sur chaque place financière il existe à tout moment, non pas une seule, mais une multitude de courbes de taux. On peut globalement distinguer deux types de courbes :

- Les courbes observées, ou courbes de marché, qui sont construites directement à partir de cotations sur les marchés (exemples : courbes swap, courbe de rendement des obligations d'Etat) ;

- Les courbes implicites, qui sont déduites à partir de cotations de marché, mais en les transformant (exemples : courbe des taux zéro-coupon, courbe des taux de rendement au pair).

Afin d'obtenir une courbe des taux continue, nous utiliserons la méthode d'interpolation linéaire. Cette méthode s'applique sur la courbe de référence quotidiennement publiée par BAM.

### III.2 Technique de l'interpolation de la courbe zéro-coupon :

La donnée des taux zéro-coupon pour les maturités croissantes  $t_1 < t_2 < \dots < t_N$  est appelée courbe des taux zéro-coupon. En pratique, les taux zéro-coupon pour des maturités  $t \neq t_i$  ( $i=1\dots N$ ) sont calculés par interpolation à partir des taux existants et la transformation des taux.

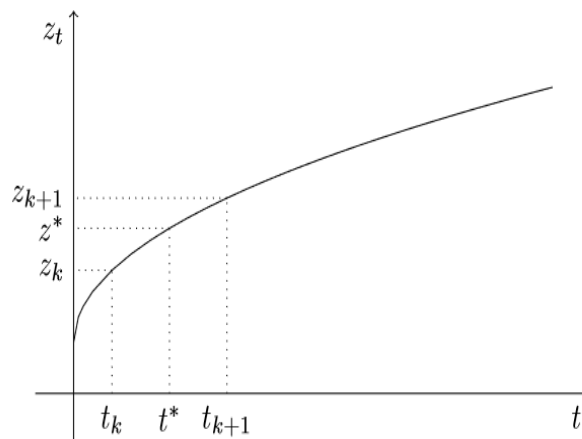


Figure 1: Courbe des taux zéro-coupon

Trois méthodes sont envisageables :

- a- Linéaire
- b- Cubique
- c- Cubique différentiable

Les méthodes d'interpolation cubiques permettent d'introduire de la convexité dans la construction de la courbe des taux. La méthode cubique différentiable permet en outre d'obtenir une courbe « lisse » en chaque point. Bien qu'attractives sur le plan

technique, ces méthodes peuvent générer des taux difficilement justifiables sur le plan financier voir même aberrants (taux négatifs). Dans la suite, on utilisera la méthode d'interpolation linéaire qui est la plus simple à mettre en œuvre et conforme aux usages du marché Marocain.

La méthode d'interpolation linéaire consiste à calculer le taux zéro-coupon  $Z$  pour une durée  $f$  quelconque à partir des taux zéro-coupon adjacents  $Z_k$  et  $Z_{k+1}$  de durées respectives  $f_k$  et  $f_{k+1}$  avec  $f_k < f < f_{k+1}$  en utilisant une combinaison linéaire des deux taux « prorata temporis » :

$$Z = \frac{(f_{k+1} - f) \times Z_k + (f - f_k) \times Z_{k+1}}{f_{k+1} - f_k}$$

C'est cette méthode d'interpolation que nous allons utiliser pour le calcul des taux zéro-coupon à des dates quelconques.

### Transformation des taux :

Sur le court terme, les instruments sont généralement de type zéro-coupon. En effet sur le moins d'un an, le remboursement du capital intervient en même temps que le paiement des intérêts. On a donc bien 2 flux. Le problème se situe uniquement au niveau de la conversion du taux monétaire en base actuarielle :

$$r_a = \left(1 - \frac{r_m * n}{360}\right)^{\frac{365}{n}} - 1$$

Inversement, la conversion du taux actuariel en taux monétaire se fait comme suit :

$$r_m = \left(\left(1 - r_a\right)^{\frac{n}{365}} - 1\right) \frac{360}{n}$$

### III.3 Construction de la courbe zéro-coupon par la méthode de Bootstrapping :

Afin de pallier aux problèmes cités avant, on construit, à partir des prix d'instruments cotés, une courbe de taux zéro-coupon. La technique utilisée pour y parvenir porte le nom Bootstrapping ou, en français, méthode de proche en proche.

Cette méthode est basée sur l'hypothèse, que le prix théorique d'une obligation soit la somme de ses flux actualisés aux taux zéro-coupon de l'échéance de chaque flux.

#### Illustration de la méthode de Bootstrapping :

On choisit un titre de maturité 2 ans qui verse deux flux :

$$P = \frac{VN * R}{(1 + r_{zc1})} + \frac{VN + VN * R}{(1 + r_{zc2})^2}$$

Chapitre 1 : Mise en contexte et notions de base  
où :

R = le taux de rendement de l'obligation.

VN = la valeur nominale de l'obligation

P= Prix de l'obligation

$r_{zc_1}$  = le taux zéro-coupon 1 an.

$r_{zc_2}$  = le taux zéro-coupon 2 ans.

Hypothèse :

On suppose que le prix de l'obligation est égal à sa valeur nominale pour simplifier les calculs tel que  $P=VN$ . Le taux zéro-coupon 1 an est connu puisqu'il est égal au taux de rendement actuariel 1 an. Nous obtenons ainsi le taux zéro-coupon 2 ans qui s'exprime comme suit :

$$r_{zc_2} = \left( \frac{1 + R}{1 + PV_1} \right)^{\frac{1}{2}} - 1$$

avec :

$$PV_1 = \frac{R}{(1 + r_{zc_1})}$$

De manière générale, le taux zéro-coupon n ans est donné par l'expression suivante :

$$r_{zc_n} = \left( \frac{1 + R}{1 + PV_n} \right)^{\frac{1}{n}} - 1$$

avec :

$$PV_n = \sum_{i=1}^{n-1} \frac{R}{(1 + r_{zc_i})^i}$$

R = taux de rendement correspondant à la maturité n ans.

On obtient ainsi la courbe de taux zéro-coupon pour toutes les maturités.

## Interface de calcul :

TAUX DE REFERENCE DU MARCHÉ SECONDAIRE DES BONS DU TRESOR					
Date d'échéance	Transaction	TMP	Value Date	Maturité	Taux d'actualisation
12/11/2012	331.37	3.60%	29/10/2012	14	3.71%
02/12/2012	316.54	3.38%	29/10/2012	34	3.48%
02/12/2012	22.63	3.37%	02/10/2012	61	3.47%
28/01/2013	50.00	3.40%	29/10/2012	91	3.49%
12/09/2013	115.41	3.734%	18/10/2012	329	3.79%
07/07/2014	26.00	4.12%	29/10/2012	616	4.12%
15/12/2014	50.08	4.14%	22/10/2012	784	4.14%
03/02/2019	20.15	4.53%	29/10/2012	2288	4.53%
02/01/2021	65.19	4.64%	23/10/2012	2993	4.64%
03/05/2030	191.68	4.87%	22/08/2012	6463	4.87%
04/12/2036	60.82	4.63%	30/08/2012	8862	4.63%

date début	29/10/2012
date fin	29/10/2012

Maturités Pleines	Maturité	Maturité par jour	Taux actuariel
	13 sem	91	3.49208%
	26 sem	182	3.61%
	52 sem	364	3.83%
	2 ans	730	4.14%
	5 ans	1 826	4.41%
	10 ans	3 652	4.69%
	15 ans	5 478	4.81%
	20 ans	7 305	4.79%
	30 ans	10 957	4.43%

Calcul Taux Zéro-Coupon par la méthode de Bootstrapping			
Maturité	Maturité par jours	Taux d'actualisation	Taux Zéro-Coupon
13 sem	91	3.49%	3.49%
26 sem	182	3.61%	3.61%
52 sem	364	3.83%	3.83%
1	365	3.83%	3.83%
2	730	4.14%	4.14%
3	1095	4.22%	4.23%
4	1461	4.32%	4.33%
5	1826	4.41%	4.43%
6	2191	4.51%	4.53%
7	2556	4.57%	4.61%
8	2922	4.63%	4.67%
9	3287	4.66%	4.71%
10	3652	4.69%	4.73%
11	4017	4.71%	4.76%
12	4383	4.73%	4.79%

Afin de construire la courbe des taux, nous avons mis en place une application sous VBA qui nous permet de télécharger la courbe du marché directement du site de BAM, et ensuite de calculer les taux actuariels pour les maturités pleines (de 13 semaines à 30 ans) par interpolation de ladite courbe, et finalement construire la courbe zéro-coupon par Bootstrapping.

### III.4 Modèles des taux d'intérêts

Il existe deux grandes approches concernant la modélisation de la dynamique des taux d'intérêts.

La première classe de modèles de taux d'intérêt constitue l'approche sans arbitrage. Les modèles de ce groupe décrivent l'évolution du taux court comme un processus stochastique avec des paramètres dépendant du temps. Ces paramètres sont ensuite calibrés afin que les prix des obligations estimés par le modèle correspondent exactement aux prix des obligations observées au moments de l'estimation. Parmi les plus importants des modèles de cette classe sont Ho et Lee (1986) et Heath-Jarrow-Morton, Heath et al (1992).

L'inconvénient de cette approche est que ses propriétés dynamiques peuvent impliquer l'évolution des taux d'intérêts d'une manière qui n'est pas empiriquement justifiée. Cependant, puisque ces modèles correspondent avec précision aux rendements

observés, ils sont largement utilisés, en particulier pour la tarification des produits dérivés à court terme.

D'une autre part, les modèles d'équilibre et les modèles de taux court ne sont pas conçus pour adapter la courbe actuelle avec précision. Ce sont des modèles paramétriques qui ont pour objectifs de décrire l'impact de l'économie sur la courbe de rendement (modèle d'équilibre) ou de capturer l'évolution des taux d'intérêts dans le temps (modèles à taux court). Les paramètres de ces modèles sont généralement estimés à partir de données historiques puisqu'ils sont supposés être constants. Cela, toutefois, peut provoquer une mauvaise modélisation de la courbe bien que les prix des actifs dans ces modèles évoluent également de manière libre d'arbitrage. Les modèles les plus célèbres de cette classe ont été proposés par Vasicek (1977) et Cox, Ingersoll, Ross (dorénavant CIR), Cox et al (1985). Ces deux modèles décrivent l'évolution taux court avec un processus stochastique retour à la moyenne.

Outre ces trois approches principales, il existe également une classe de modèles qui tentent d'identifier la structure par terme des taux d'intérêt en estimant uniquement les paramètres de certaines formes fonctionnelles prédéterminées pour la courbe des taux. Leur seul but est de fournir un bon ajustement aux rendements observés et ils ne nous donnent aucune information sur la dynamique temporelle de la courbe des taux. Pour les autorités monétaires, ces modèles aident à analyser la politique monétaire. Pour les traders d'obligations, ces modèles sont utilisés pour identifier les obligations surévaluées ou sous-évaluées.

Finalement, l'une des classes de modèles les plus populaires est la classe de modèles qui construisent la courbe des taux forward comme une combinaison d'exponentielles et de polynômes. A cette classe de modèle appartient également le modèle proposé par Nelson et Siegel et Nelson Siegel Svensson.

## **CHAPITRE 2**

### **Modélisation de la courbe des taux via le modèle de Diebold Li : Modèle VAR**

Dans ce chapitre, nous allons étudier la modélisation de la courbe des taux via le modèle de Diebold Li ainsi que l'application du filtre de Kalman pour des fins de prévision.

Le modèle de Diebold & Li propose une extension du modèle de Nelson & Siegel (1987) pour anticiper la courbe des taux. Ainsi, avant de se lancer dans cette modélisation, il s'avère nécessaire d'expliquer le modèle de Nelson & Siegel Statique.

## I. Modèle de Nelson & Siegel Statique

Le modèle de Nelson & Siegel, introduit en 1987, est l'un des plus populaires modèles paramétriques utilisés dans la modélisation de la courbe des taux. Il est à la fois simple et flexible, et fournit des résultats statistiquement précis et économiquement significatifs. C'est un modèle statique à trois facteurs servant à lisser les taux forward. Il permet d'obtenir les taux d'échéances qui ne sont pas observés sur les marchés.

### Notation :

Afin d'unifier la notation, nous présentons ci-dessous celle que nous allons suivre tout au long de ce travail :

Le prix d'une obligation zéro-coupon en date  $t$  payant 1 u. m à la date  $T$  est noté  $P(t, T)$ .

Le taux de rendement continu d'une obligation zéro-coupon est noté  $R(t, T)$ . La relation entre ce taux et le prix de l'obligation est :

$$P(t, T) = \exp(-\tau R(t, T)) \text{ avec : } P(T, T) = 1 \text{ et } \tau(t, T) = T - t.$$

Le taux court  $r(t)$  est défini de la manière suivante :

$$r(t) = \lim_{T \rightarrow t} R(t, T)$$

Il est possible que le taux forward (taux à terme) s'écrive :

$$f(t, T_1, T_2) = -\frac{1}{T_2 - T_1} \ln\left(\frac{P(t, T_2)}{P(t, T_1)}\right)$$

Le taux forward instantané est :

$$f(t, T) = \lim_{\varepsilon \rightarrow 0} f(t, T, T + \varepsilon)$$

On peut également exprimer le taux à capitalisation continue e fonction du taux forward instantané :

$$R(t, T) = \frac{1}{\tau(t, T)} \int_t^T f(t, s) ds \quad (1)$$

### Modèle de Nelson & Siegel Statique :

Nelson et Siegel (1987) ont supposé que le taux à terme instantané à la maturité ( $\tau$ ) notée  $f(\tau)$  est donnée par la solution d'une équation différentielle du second ordre tel que :

$$f(\tau) = \beta_0 + \beta_1(-\tau/\theta) + \beta_2[(\tau/\theta)\exp(-\tau/\theta)]$$

En remplaçant f dans l'équation (1) on obtient :

$$\begin{aligned} R(t, T) &= \frac{1}{\tau} \int_0^\tau \beta_0 + \beta_1 \exp(-s/\theta) + \beta_2 [(s/\theta)\exp(-s/\theta)] ds \\ &= \frac{1}{\tau} (\beta_0 [s]_0^\tau + \beta_1 [-\theta \exp(-s/\theta)]_0^\tau + \beta_2 [-s \exp(-s/\theta) - \theta \exp(-s/\theta)]_0^\tau) \\ &= \beta_0 + \beta_1 \frac{1 - \exp(-\tau/\theta)}{\tau/\theta} + \beta_2 \left[ \frac{1 - \exp(-\tau/\theta)}{\tau/\theta} - \exp(-\tau/\theta) \right] \end{aligned}$$

L'utilisation de ce modèle requiert l'estimation d'un vecteur de paramètres contenant les quatre paramètres  $X = (\beta_0, \beta_1, \beta_2, \theta)$ .

Les paramètres  $\beta$  sont interprétés par Nelson & Siegel (1987) de la manière suivante :

- $\beta_0$  représente un facteur à long terme.
- $\beta_1$  représente un facteur à court terme.
- $\beta_2$  représente un facteur à moyen terme.
- $\theta$  représente le paramètre d'échelle, c'est-à-dire qu'il permet la dilatation ou la contraction de l'échelle du temps.

## II. Modèle de Diebold & Li

Bien que le modèle de Nelson-Siegel réussisse à adapter la structure par terme des taux d'intérêt, à une date donnée, il ne dit rien sur l'évolution de la courbe des taux au cours du temps. Diebold et Li ont examiné les performances de prévision des modèles

Chapitre 2 : Modélisation de la courbe des taux via le modèle de Diebold Li : Modèle VAR de structure par terme et ils ont utilisé le cadre de Nelson-Siegel comme leur point de départ. Diebold et Li (2006) ont montré qu'une approche dynamique basée sur Nelson Siegel avait une bonne performance pour la prévision de la structure à terme des taux.

Lors de notre étude nous allons travailler avec la formulation suivante proposée par Diebold et Li pour le modèle Nelson-Siegel :

$$R_t(m) = \beta_{0t} + \beta_{1t} \left( \frac{1 - e^{-\lambda_t m}}{\lambda_t m} \right) + \beta_{2t} \left( \frac{1 - e^{-\lambda_t m}}{\lambda_t m} - e^{-\lambda_t m} \right)$$

où  $\lambda_t = \theta_t^{-1}$

### Interprétation des paramètres :

Le paramètre  $\lambda_t$  reflète la vitesse du déclin du facteur exponentiel, ou la vitesse de convergence vers le taux à long terme ; de faibles valeurs de  $\lambda_t$  produisent une lente décroissance et modélisent mieux la partie long terme de la courbe, tandis que des valeurs élevées de  $\lambda_t$  produisent une décroissance rapide et représentent mieux la partie courte. Ce paramètre représente également la valeur qui maximise la courbure au moyen terme.

Les paramètres  $\beta_{0t}$ ,  $\beta_{1t}$  et  $\beta_{2t}$  sont considérés comme des facteurs latents.

On associe à  $\beta_{0t}$  une constante égale à 1 donc ce paramètre peut être interprété comme un facteur à long terme. D'ailleurs, nous avons pour t donné :

$$\lim_{m \rightarrow \infty} R_{ZC}(m) = \beta_0$$

On associe à  $\beta_{1t}$  la fonction monotone suivante :  $\left( \frac{1 - e^{-\lambda_t m}}{\lambda_t m} \right)$  qui prend la valeur 1 quand  $m=0$  et décroît vers 0 ; ainsi  $\beta_{1t}$  est considéré comme un facteur à court terme. Nous avons :  $\beta_{1t} = R_t(0) - R_t(\infty)$

On associe à  $\beta_{2t}$  la fonction suivante :  $\left( \frac{1 - e^{-\lambda_t m}}{\lambda_t m} - e^{-\lambda_t m} \right)$  qui prend la valeur 0 quand  $m=0$  (Contrairement au cas du court terme), accroît et décroît vers zéro pour des valeurs élevées de  $m$  (ce qui ne correspond non plus au cas du long terme). Et donc, ce dernier paramètre peut être interprété comme facteur à moyen terme.

Nous représentons dans le graphique ci-dessous les variables du modèle. Nous pouvons constater l'évolution de ces variables « factor loading » pour un lambda fixé.

Nous pouvons remarquer que, dans le modèle de Diebold & Li (par la suite noté DNS

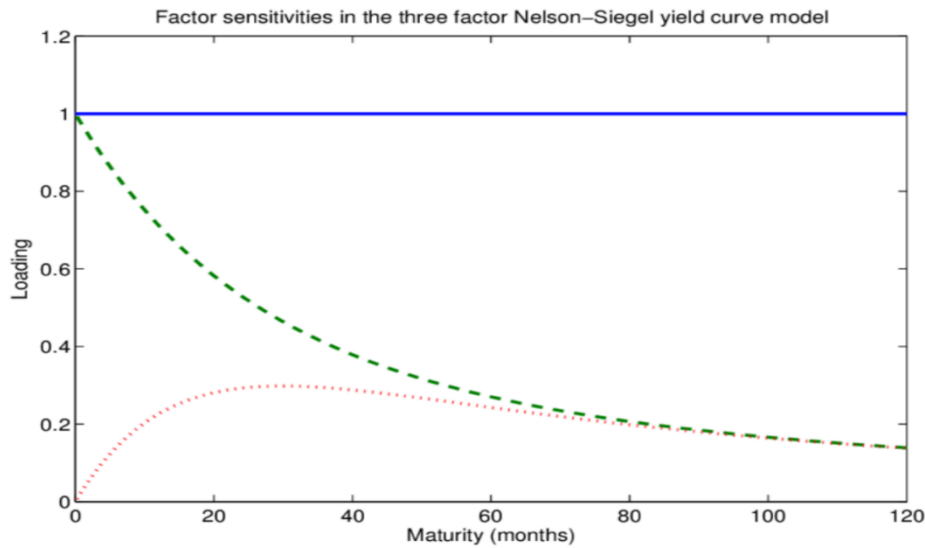


Figure 2: Dynamique des paramètres de Diebold & Li

(Dynamic Nelson-Siegel)), les paramètres ne sont plus statiques. Nous avons donc une version dynamique du modèle Nelson & Siegel. Diebold et Li considèrent que ces paramètres suivent un processus autorégressif VAR(1) qui peut être écrit sous la forme matricielle suivante :

$$\begin{pmatrix} \beta_{0,t} \\ \beta_{1,t} \\ \beta_{2,t} \end{pmatrix} = \begin{pmatrix} u_0 \\ u_1 \\ u_2 \end{pmatrix} + \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} \beta_{0,t-1} \\ \beta_{1,t-1} \\ \beta_{2,t-1} \end{pmatrix} + \begin{pmatrix} \eta_{0,t} \\ \eta_{1,t} \\ \eta_{2,t} \end{pmatrix}$$

où  $\eta \sim N(0, Q)$ .

Les termes d'erreur ont la matrice de variance conditionnelle suivante :

$$Q = \begin{pmatrix} \sigma^2(\beta_0) & cov(\beta_0, \beta_1) & cov(\beta_0, \beta_2) \\ cov(\beta_1, \beta_0) & \sigma^2(\beta_1) & cov(\beta_1, \beta_2) \\ cov(\beta_2, \beta_0) & cov(\beta_2, \beta_1) & \sigma^2(\beta_2) \end{pmatrix}$$

On peut le réécrire de la manière suivante :

$$\beta_t = (I - A)\mu + A\beta_{t-1} + \eta_t$$

Le prix d'une obligation zéro-coupon suivant le modèle de Diebold & Li s'écrit sous la forme suivante :

$$P(t, T) = \exp \left[ -m\beta_{0t} - \beta_{1t} \left( \frac{1 - e^{-\lambda_t m}}{\lambda_t} \right) - \beta_{2t} \left( \frac{1 - e^{-\lambda_t m}}{\lambda_t} - me^{-\lambda_t m} \right) \right]$$

Ce modèle peut être utilisé pour la couverture puisque Litterman et Scheinkman (1991) ont montré que les trois facteurs cités ci-haut décrivent plus de 96% des variations de la structure à terme des taux. Ils ont pu obtenir ces résultats en analysant les composantes principales de la variation des taux d'intérêts. Nous pouvons donc nous protéger d'une variation de la structure à terme des taux en rendant notre portefeuille insensible à une variation des trois facteurs : niveau, pente et courbure.

### III. Méthodologie

Nous présentons ici la méthodologie utilisée afin de réaliser nos prévisions et vérifier la pertinence économique. Nous allons exposer la procédure à deux étapes puis l'utilisation du filtre de Kalman.

#### III.1 Procédure a deux étapes

Afin de réaliser nos prévisions, deux étapes sont nécessaires. Pour l'estimation en échantillon, seule la première étape est nécessaire en utilisant toutes les données disponibles. La première étape consiste à calibrer la courbe à chaque mois. En fixant le paramètre  $\lambda$  à 0.0609, nous obtenons une série de paramètres  $\beta$  par la méthode des moindres carrés ordinaires, représentant les paramètres qui permettent de reproduire la courbe des taux à chaque mois. La seconde étape permet de créer un processus autorégressif à partir de la série de paramètres obtenue lors de la première étape.

Grâce à la méthode des moindres carrés, on obtient les paramètres  $A$  et  $\mu$  qui nous permettent de trouver la prochaine occurrence des paramètres  $\beta$ .

Les taux sont ensuite retrouvés à partir de cette équation :

$$R_t = \Lambda \beta_t + e_t$$

Tel que :

$$\Lambda = \begin{pmatrix} 1 & \frac{1 - e^{-\lambda m_1}}{\lambda m_1} & \frac{1 - e^{-\lambda m_1}}{\lambda m_1} - e^{-\lambda m_1} \\ \vdots & \ddots & \vdots \\ 1 & \frac{1 - e^{-\lambda m_n}}{\lambda m_n} \dots & \frac{1 - e^{-\lambda m_n}}{\lambda m_n} - e^{-\lambda m_n} \end{pmatrix}$$

$$R_t = \begin{pmatrix} R_t(m_1) \\ \vdots \\ R_t(m_n) \end{pmatrix} \quad \beta_t = \begin{pmatrix} \beta_{0t} \\ \beta_{1t} \\ \beta_{2t} \end{pmatrix} \quad e_t \sim N(0, H)$$

où :  $m_i, i=1, \dots, n$  représentent les 33 ténors : 3mois , 6mois, 9mois , 1 an , ..., 30 ans.

Pour un horizon de prévision h jours, l'équation utilisée est la suivante :

$$E_t(\beta_{t+h}) = \left( \sum_{i=1}^h A^i \right) (I - A)\mu + A^h \beta_t$$

Le problème avec cette procédure à deux étapes, c'est que nous incorporons les erreurs de lissage à nos prévisions. Pour remédier à cela, dans la prochaine sous-section nous examinons une procédure à une étape.

### III.2 Filtre de Kalman

En 1960, Kalman a décrit une solution récursive pour le problème de filtrage des données discrètes linéaires en minimisant la moyenne des erreurs quadratiques.

Comme l'explique Harvey (1987), le filtre de Kalman est un ensemble d'équations qui permet à un estimateur d'être mis à jour à chaque fois qu'une nouvelle observation est disponible.

Le filtre est utilisé sur un modèle espace-états c'est-à-dire un modèle qui contient une équation de mesure décrivant un élément aléatoire observable et une équation d'état contenant un élément non observable (voir Annexe A pour plus de détails). Il permet tout d'abord d'avoir la meilleure prédiction de la prochaine observation étant donné les informations disponibles. Ensuite, dès qu'une nouvelle observation est disponible les variables d'états sont mises à jour.

Dans notre cas, les variables d'états sont les coefficients beta des deux modèles alors que les observations sont les taux d'intérêts des différentes échéances.

## IV. Application

### IV.1 Description du Dataset

Nous utilisons pour notre travail les mêmes données utilisées par Diebold & Li. Celles-ci sont les données représentant les taux du marché marocain (les taux zéro coupon) allant du mai 2004 jusqu'à mai 2018, chaque ligne de notre base représente une courbe des taux, c'est à dire, elle représente les taux zéro-coupon en fonction des maturités (33 maturités).

Nous entamons d'abord une étude de l'historique de la courbe. Nous nous intéressons aux niveaux des taux pour les différents segments de maturités, ainsi qu'aux formes historiques de la courbe, qui sont reflétées par la pente et la courbure. Nous représentons ci-dessous l'évolution des taux depuis 2004 pour les maturités 3 mois, 1 an, 2 ans, 5 ans, 10 ans et 30 ans, ainsi que des statistiques des taux.

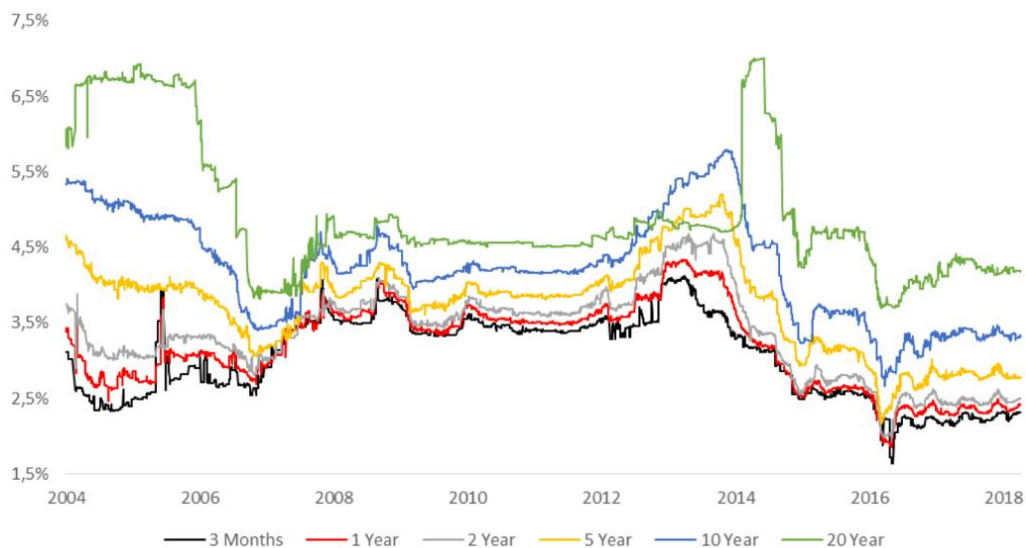


Figure 3: Evolution des taux marocains depuis 2004

Ténors	Moyenne	Maximum	Minimum	Ecart-type	Coefficient de variation
3 mois	3,04%	4,12%	1,63%	5,5E-03	18,27%
6 mois	3,08%	4,25%	1,73%	5,7E-03	18,33%
9 mois	3,13%	4,29%	1,80%	5,7E-03	18,05%
1 an	3,18%	4,34%	1,86%	5,7E-03	17,86%
2ans	3,34%	4,68%	1,98%	5,9E-03	17,71%
3 ans	3,48%	4,86%	2,08%	6,0E-03	17,20%
4 ans	3,61%	5,02%	2,15%	6,0E-03	16,74%
5 ans	3,73%	5,19%	2,17%	6,2E-03	16,53%
6 ans	3,83%	5,28%	2,27%	6,3E-03	16,34%
7 ans	3,93%	5,38%	2,40%	6,4E-03	16,25%
8 ans	4,04%	5,48%	2,53%	6,6E-03	16,23%
9 ans	4,14%	5,62%	2,66%	6,7E-03	16,23%
10 ans	4,22%	5,79%	2,67%	6,9E-03	16,41%
11 ans	4,31%	5,92%	2,79%	7,1E-03	16,36%
12 ans	4,40%	5,98%	2,94%	7,2E-03	16,29%
13 ans	4,49%	6,05%	3,07%	7,3E-03	16,26%
14 ans	4,58%	6,18%	3,16%	7,4E-03	16,25%
15 ans	4,66%	6,27%	3,25%	7,6E-03	16,23%
16 ans	4,71%	6,31%	3,34%	7,3E-03	15,61%
17 ans	4,75%	6,40%	3,44%	7,4E-03	15,61%
18 ans	4,81%	6,51%	3,53%	7,8E-03	16,20%
19 ans	4,87%	6,74%	3,61%	8,3E-03	17,11%
20 ans	4,92%	7,01%	3,69%	8,8E-03	17,97%
25 ans	4,90%	6,75%	3,94%	7,4E-03	15,19%
30 ans	5,02%	7,87%	3,81%	7,9E-03	15,82%

*Tableau 1: Statistiques Descriptives du Dataset*

Nous pouvons constater qu'en général les taux suivent la même tendance sauf pour la partie longue de la courbe qui présente souvent des anomalies. Ces anomalies sont généralement dues au manque de liquidité sur ce segment. Le coefficient de

variation montre que la volatilité des taux diminue avec la maturité, ce qui confirme que la partie courte de la courbe est la plus volatile.

## IV.2 Application

### IV.2.1 Procédure a deux étapes

L'approche qu'on a suivi afin de réaliser nos prévisions se base sur les deux étapes suivantes : dans un premier temps, nous estimons le modèle en fixant  $\lambda$  à partir de la valeur originale proposée par Diebold et Li, à savoir  $\lambda_1 = 0,7308$ . Ensuite nous chercherons une nouvelle valeur optimale qui correspondrait au cas de notre étude. Finalement, nous allons déterminer la meilleure estimation possible en procédant à une comparaison des résultats obtenus par les deux approches.

#### Détermination de lambda de notre modèle :

Nous utilisons pour notre étude les mêmes données utilisées par Diebold et Li. Celles-ci sont les données des taux zéro-coupon avec échéances allant de 3 mois jusqu'à 30 ans. Les taux ont été calculés par un fichier Excel automatisé sous VBA.

Pour pouvoir déterminer la valeur de lambda, on a effectué une régression non linéaire pour pouvoir extraire les quatre paramètres  $X = (\beta_0, \beta_1, \beta_2, \lambda)$  pour chaque date. Pour ce faire on a utilisé l'algorithme de Levenberg-Marquardt permet d'obtenir une solution numérique au problème de minimisation d'une fonction, souvent non linéaire et dépendant de plusieurs variables. L'algorithme repose sur les méthodes derrière l'algorithme de Gauss-Newton et l'algorithme du gradient. Plus stable que celui de Gauss-Newton, il trouve une solution même s'il est démarré très loin d'un minimum. Cependant, pour certaines fonctions très régulières, il peut converger légèrement moins vite. Cet algorithme fut développé par Kenneth Levenberg, puis publié par Donald Marquardt.

Pratiquement, on a implémenté l'algorithme de LM sous R en utilisant la fonction `nls`. On a obtenu comme résultats l'estimation du vecteur X défini précédemment pour chaque date, puis on a déterminé  $\lambda$  de la façon suivante :  $\hat{\lambda} = \frac{1}{n} \sum_1^n \lambda_i$

On retrouve que  $\lambda_2 = 0.0614$

Nous notons « modèle 1 » le modèle correspondant à  $\lambda$  égale à 0.7308 qui est la valeur proposée par les auteurs, et « modèle 2 » celui qui correspond à la valeur que nous avons trouvée, qui est égale à 0.0614. Nous comparons les résultats obtenus.

### Comparaison entre les deux modèles :

Pour comparer entre les deux modèles, on a commencé par une analyse des résidus des deux modèles :

Modèle	Modèle 1				Modèle2			
	Mean	Min ( $10^{-2}$ )	Max	3ème quantile	Mean	Min ( $10^{-2}$ )	Max	3ème quantile
3 mois	0.228	0.005	0.740	0.348	0.088	0.006	0.737	0.113
9 mois	0.076	0.004	0.029	0.011	0.062	0.001	0.456	0.022
1 an	0.173	0.011	0.791	0.259	0.046	0.000	0.623	0.065
3ans	0.239	0.052	0.771	0.363	0.047	0.001	0.461	0.065
5 ans	0.065	0.006	0.395	0.084	0.063	0.001	0.447	0.080
10ans	0.269	0.027	0.881	0.415	0.077	0.001	0.488	0.115
13 ans	0.265	0.001	1.014	0.399	0.080	0.000	0.453	0.121
15 ans	0.245	0.002	1.290	0.335	0.091	0.001	1.357	0.191
20 ans	0.193	0.008	1.926	0.233	0.177	0.001	1.821	0.237
25 ans	0.192	0.004	0.840	0.272	0.054	0.005	0.516	0.041
30 ans	0.304	0.007	2.262	0.444	0.132	0.013	0.926	0.179

*Tableau 2: Statistiques des résidus d'estimation des taux zéro coupon pour les deux modèles*

On remarque que le modèle 2 réalise en moyenne une meilleure estimation en échantillon par rapport au modèle 1 car, sur l'ensemble des échéances, la moyenne et le 3ème quantile sont plus petits pour ce modèle.

Afin de mieux comparer nos résultats, nous nous basons sur la racine carrée de l'erreur moyenne quadratique. Cette statistique est mieux connue sous le nom de RMSE pour Root Mean Squared Error.

Nous présentons dans le tableau suivant le ratio des racines carrées de l'erreur quadratique moyenne pour toutes les échéances des deux modèles.

<b>3mois</b>	<b>6mois</b>	<b>9mois</b>	<b>1an</b>	<b>2ans</b>
2.06	0.87	1.22	3.07	5.28
<b>3ans</b>	<b>4ans</b>	<b>5ans</b>	<b>6ans</b>	<b>7ans</b>
4.22	2.36	1.007	1.25	1.78
<b>8ans</b>	<b>9ans</b>	<b>10ans</b>	<b>20ans</b>	<b>30 ans</b>
2.2	2.55	2.94	1.09	2.37

Tableau 3: Ratio des racines carrées de l'erreur quadratique moyenne (RMSE)

On remarque que le modèle 2 réalise en moyenne une meilleure estimation par rapport au premier modèle vu que, sur l'ensemble des échéances, la racine carrée de l'erreur quadratique moyenne est plus petite pour ce modèle.

Pour conclure, nous avons opté pour un Backtesting afin de mieux choisir notre modèle.

### Backtesting :

Pour choisir la valeur adéquate du lambda, on va procéder par un Backtesting pour trancher sur la valeur qui ajuste au mieux les taux.

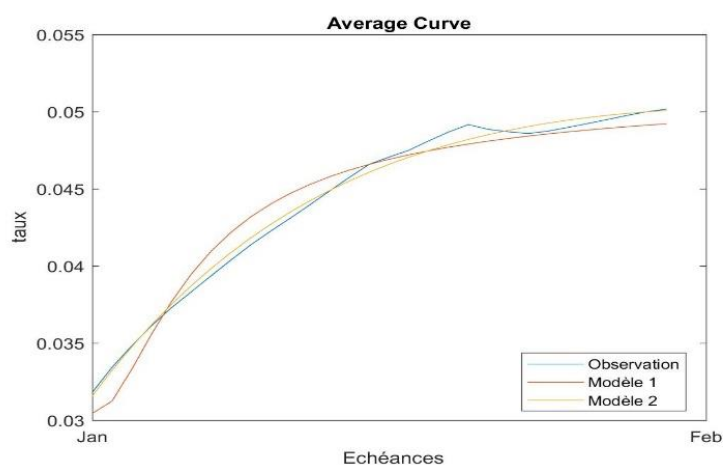


Figure 4: Courbe des taux moyenne

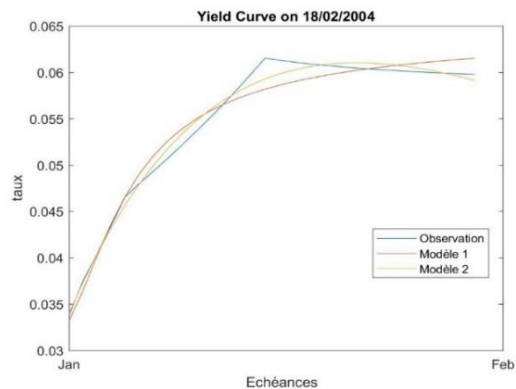


Figure 5: Courbe des taux relative à la date 18/02/2004

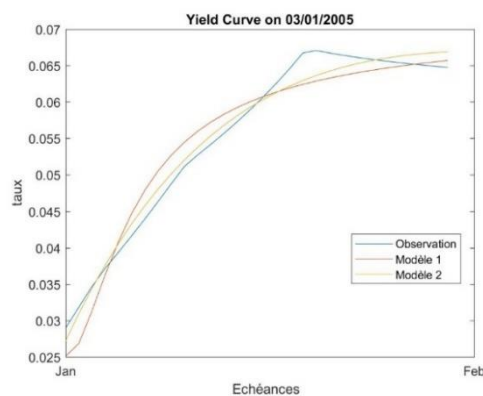


Figure 6: Courbe des taux relative à la date 30/01/2005

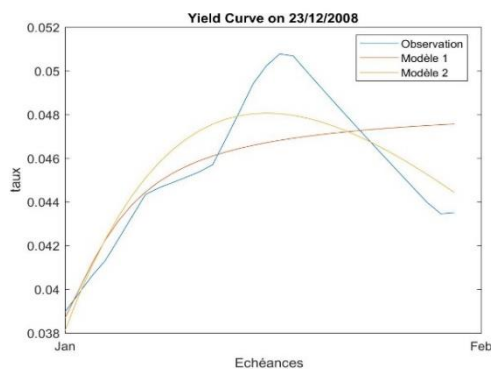


Figure 7: Courbe des taux relative à la date 23/12/2008

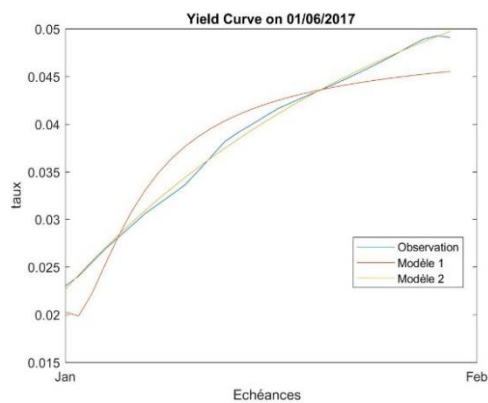


Figure 8: Courbe des taux relative à la date 01/06/2017

La courbe moyenne est concave et croissante dans les trois cas. Nous remarquons que pour toutes les dates, et pour la courbe moyenne également, les courbes obtenues par le modèle2 sont les plus proches à celles observées. Les écarts sont beaucoup plus faibles par rapport au modèle1. On retient alors la valeur  $\lambda=0.061461$  pour la suite de notre étude.

Après avoir fixé  $\lambda$ , on réestime les autres paramètres par la méthode des moindres carrés ordinaires (régression linéaire cette fois-ci). Cette estimation nous a fourni pour chacun des facteurs une série d'estimations sur le long de la période que nous avons considéré.

### Dynamique des facteurs :

La seconde étape permet de créer un processus autorégressif VAR (1) à partir de la série de paramètres obtenue lors de la première étape.

Modèle VAR (1) obtenu :

$$\begin{pmatrix} \beta_{0t} \\ \beta_{1t} \\ \beta_{2t} \end{pmatrix} = \begin{pmatrix} 0,0032 \\ -0,0031 \\ -0,0044 \end{pmatrix} + \begin{pmatrix} 0.9028 & -0.0842 & -0.032 \\ 0.0942 & 1.0818 & 0.0029 \\ 0.1372 & 0.1216 & 1.0032 \end{pmatrix} \begin{pmatrix} \beta_{0t-1} \\ \beta_{1t-1} \\ \beta_{2t-1} \end{pmatrix} + \begin{pmatrix} \eta_t(\beta_0) \\ \eta_t(\beta_1) \\ \eta_t(\beta_2) \end{pmatrix}$$

Avec :

$$\eta_t \sim N(0, Q)$$

$$Q = \begin{pmatrix} 2,420e - 05 & -2,350e - 05 & -3,532e - 05 \\ -2,350e - 05 & 2,288e - 05 & 3,408e - 05 \\ -3,532e - 05 & 3,408e - 05 & 5,305e - 05 \end{pmatrix}$$

## IV.2.2 Procédure a une étape : Filtre de Kalman

### Création du modèle SSM :

Avant d'entamer l'estimation de notre modèle, nous devons créer un modèle d'état-espace (SSM) correspondant au modèle de Diebold-Li. Pour ce faire, l'utilisation d'une fonction de mappage s'avère nécessaire. Dans notre cas, nous avons opté pour l'utilisation de la fonction de mappage EXAMPLE\_DIEBOLDLI qui est spécifique pour le modèle de Diebold-Li.

Considérer la fonction de mappage comme argument dans la fonction SSM permet de créer un espace-état de la forme :

- Equation d'état :  $X_t = AX_{t-1} + Bu_t$
- Equation d'observation :  $Y_t = CX_t + De_t$

Dans notre exemple, on dispose de 3 facteurs (états) décrivant le comportement des betas dans le temps, ainsi  $X_t$  est un vecteur 3 x 1. La dimension du vecteur de rendement observé  $Y_t$  est déterminé par le nombre d'échéances incluses dans la série chronologique des courbes de rendement (33 maturités dans notre cas).

Pour prendre en charge le modèle de Diebold-Li, le modèle SSM est créé implicitement en spécifiant la fonction de mappage en tant que fonction mappant la colonne d'entrée (vecteur colonne de paramètres) vers les matrices A, B, C et D du modèle SSM. Ce vecteur contient les matrices A, B et D, le coefficient lambda permettant le calcul de la matrice C (ce calcul se fait dans son intégralité dans la fonction de mappage) et le coefficient  $\mu$  permettant de sa part de dégonfler les rendements  $Y_t$  (procédure expliquée ci-dessus).

De plus, cette fonction de cartographie impose également des contraintes sur la covariance des processus bruit et dégonfle les rendements d'entrée observés  $Y_t$  pour faire l'estimation. La dégonflation s'avère nécessaire dans notre cas nous disposons des observations sur les rendements  $Y_t$  alors que le modèle de Diebold\_Li se base sur l'équation d'observation suivante :

$$Y'_t = Y_t - \Lambda\mu.$$

Le fait de soustraire le terme  $\Lambda\mu$  représente le principe de dégonflation avec  $\mu$  est le vecteur 3 x 1 des moyennes de chaque beta :

$$\mu = \begin{pmatrix} \mu_0 \\ \mu_1 \\ \mu_2 \end{pmatrix}$$

### Initialisation du modèle :

Maintenant que nous avons créé notre modèle d'état-espace SSM, nous allons lui insérer les valeurs initiales vu que le filtre de Kalman est notoirement sensible aux valeurs des paramètres initiaux. Dans notre cas, nous utilisons les résultats de l'approche en deux étapes c'est-à-dire les résultats du modèle VAR pour initialiser l'estimation.

### Estimation du modèle :

Maintenant que les valeurs initiales ont été calculées, nous allons estimer notre modèle via le filtre de Kalman. L'ensemble de ce travail a été effectué sous MATLAB.

## V. Résultats et comparaison des paramètres obtenus

Nous allons maintenant comparer les résultats obtenus avec le SSM à ceux de l'approche en deux étapes.

- **Comparaison entre la matrice de transition d'état :**

Matrice de transition d'état SSM (A) :		
-----		
0.9032	-0.0839	-0.0031
0.0949	1.0815	0.0031
0.1379	0.1219	1.0032
Matrice de transition d'état en deux étapes (A) :		
-----		
0.9028	-0.0842	-0.0032
0.0942	1.0818	0.0029
0.1372	0.1216	1.0032

*Figure 9: Comparaison entre la matrice de transition d'état des deux approches*

Nous remarquons à quel point les résultats des deux modèles sont assez similaires. De plus, les éléments de la diagonale sont positifs et les plus grands, indiquant une autodynamique de chaque facteur, alors que les éléments non diagonaux, indiquent une faible dynamique des facteurs croisés.

- **Comparaison entre la matrice de covariance des perturbations de l'état**

```

Matrice de covariance des perturbations de l'état SSM (Q = BB'') :
-----
1.0e-04 *
0.1310  -0.1307  -0.1250
-0.1307  0.1310  0.1212
-0.1250  0.1212  0.2774

Matrice de covariance de perturbation d'état en deux étapes (Q) :
-----
1.0e-04 *
0.2420  -0.2350  -0.3533
-0.2350  0.2289  0.3409
-0.3533  0.3409  0.5305
    
```

*Figure 10: Comparaison entre la matrice de covariance des deux approches*

Nous remarquons que les matrices de covariance estimées sont différentes et que la variance estimée augmente au fur et à mesure qu'on avance du paramètre niveau jusqu'à celui de courbure. Cette augmentation est beaucoup plus remarquante dans le cas du modèle à deux étapes.

- **Comparaison entre la moyenne des facteurs niveau, pente et courbure**

```

Moyenne du facteur SSM:
-----
0.0389  -0.0091  0.0514

Moyenne du facteur Two-Step :
-----
0.0389  -0.0091  0.0514
    
```

*Figure 11: Comparaison entre la moyenne des facteurs des deux approches*

Nous remarquons que les moyennes estimées sont presque similaires pour l'ensemble des facteurs.

- **Comparaison des facteurs estimés :**

Les facteurs non observés, ou états latents, qui correspondent aux facteurs de niveau, de pente et de courbure du modèle Diebold & Li, sont d'un intérêt primordial pour la

Chapitre 2 : Modélisation de la courbe des taux via le modèle de Diebold Li : Modèle VAR  
prévision de l'évolution des futures courbes de rendement. Nous examinons maintenant  
les états déduits de chaque approche.

- Dans l'approche en deux étapes, les états latents (facteurs) sont les coefficients de la régression faite à l'étape 2.
- Dans l'approche SSM, on doit implémenter le lissage de Kalman pour  $t = 1, 2, \dots, T$ .

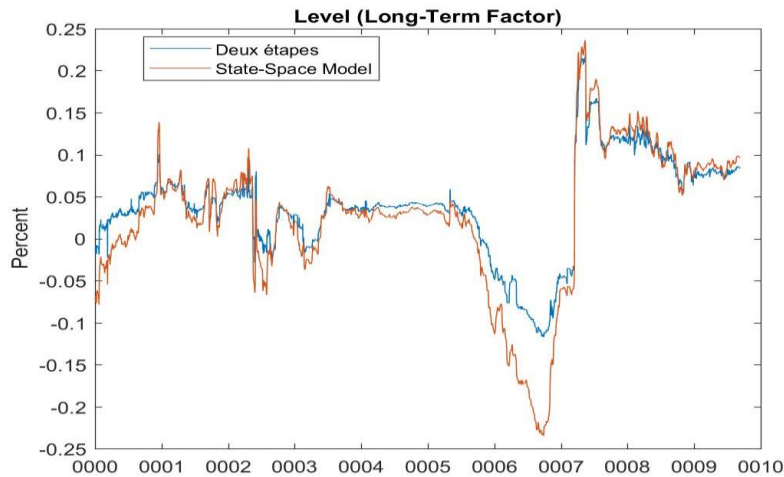


Figure 12: Le facteur niveau pour les deux modèles

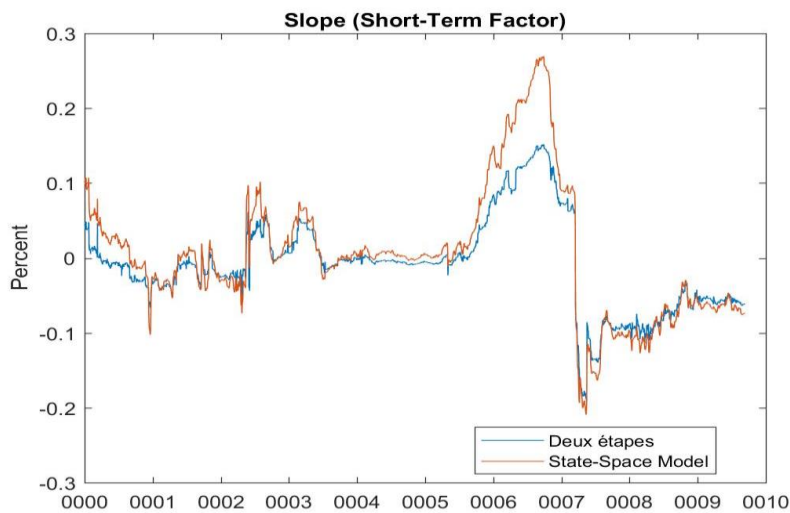


Figure 13: Le facteur pente pour les deux modèles

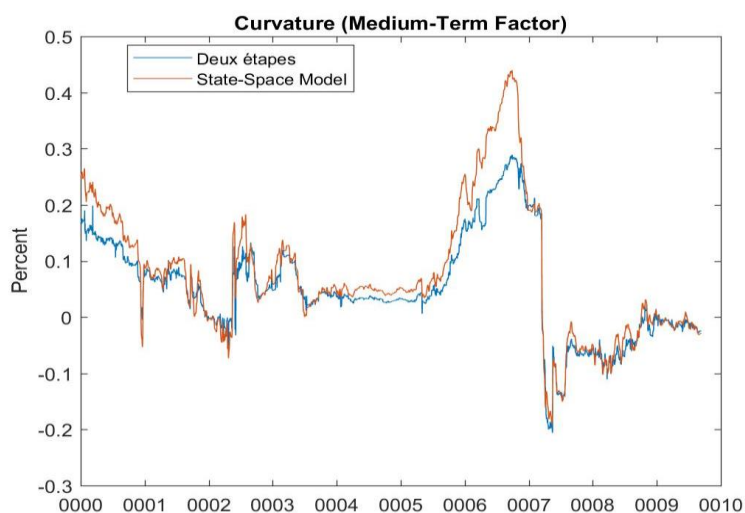


Figure 14: Le facteur courbature pour les deux modèles

On remarque que les trois facteurs suivent presque la même dynamique avec des écarts entre les deux modèles.

- **Comparaison des erreurs d'estimation**

Afin de mieux comparer nos résultats, nous nous basons sur la racine carrée de l'erreur moyenne quadratique.

Maturity (Months)	State-Space Model			Two-Step		
	Mean (bps)	Standard Deviation (bps)	RMSE (bps)	Mean (bps)	Standard Deviation (bps)	RMSE (bps)
3.0000	-0.0173	0.1021	0.1035	0.0127	0.1306	0.1312
6.0000	-0.0104	0.0506	0.0516	0.0154	0.0895	0.0908
9.0000	-0.0001	0.0251	0.0251	0.0217	0.0693	0.0726
12.0000	0.0052	0.0294	0.0299	0.0232	0.0623	0.0664
24.0000	0.0158	0.0699	0.0717	0.0200	0.0634	0.0664
36.0000	0.0137	0.0726	0.0739	0.0063	0.0683	0.0686
48.0000	0.0110	0.0659	0.0668	-0.0059	0.0750	0.0752
60.0000	0.0041	0.0684	0.0686	-0.0206	0.0919	0.0942
72.0000	-0.0084	0.0533	0.0539	-0.0393	0.0968	0.1045
84.0000	-0.0124	0.0412	0.0430	-0.0481	0.1027	0.1134
96.0000	-0.0096	0.0337	0.0351	-0.0491	0.1063	0.1171
108.0000	-0.0032	0.0302	0.0304	-0.0454	0.1057	0.1150
120.0000	0.0001	0.0453	0.0453	-0.0440	0.1012	0.1103
132.0000	0.0057	0.0697	0.0699	-0.0396	0.1044	0.1116
144.0000	0.0202	0.0961	0.0981	-0.0257	0.1101	0.1131
156.0000	0.0430	0.1261	0.1332	-0.0031	0.1227	0.1228
168.0000	0.0695	0.1581	0.1727	0.0237	0.1416	0.1435
180.0000	0.0970	0.1897	0.2131	0.0517	0.1568	0.1651
192.0000	0.0915	0.1816	0.2034	0.0469	0.0967	0.1075
204.0000	0.0881	0.2336	0.2496	0.0444	0.1356	0.1427
216.0000	0.1060	0.2873	0.3061	0.0631	0.1908	0.2009
228.0000	0.1246	0.3504	0.3718	0.0827	0.2586	0.2715 ...

L'examen du tableau ci-dessus révèle que le modèle SSM convient beaucoup mieux aux échéances allant de 3 mois à 144 mois (court – moyen terme).

## **CHAPITRE 3**

### **Modélisation de la courbe des taux via le modèle de Nelson Siegel Dynamique : Modèle VECM**

L'approche vue dans le chapitre précédant ne prenait pas en considération la stationnarité des facteurs de Nelson Siegel Dynamique, ce qui constitue une limite pour ce genre de modèle vu que l'étude de la stationnarité constitue une étape cruciale dans une étude économétrique. Pour pallier à ce problème, nous avons opté pour une étude basée sur la stationnarité ainsi que le choix du modèle adéquat.

## **I. Théorie du modèle à correction d'erreurs**

Afin de modéliser des séries temporelles, il est nécessaire de vérifier la stationnarité des variables. Pour cela, nous disposons de plusieurs tests de non-stationnarité. Si les variables sont non stationnaires, la régression standard des Moindres Carrés Ordinaires (MCO) est dite fallacieuse ou illusoire. En effet, la non-stationnarité crée des problèmes d'inférence susceptibles de causer la non-validité de notre régression. Dans ce cas, une solution est envisageable : une relation de cointégration i.e. lorsque la combinaison linéaire de plusieurs variables non stationnaires est stationnaire.

Le fait qu'un processus soit stationnaire ou non conditionne le choix de la modélisation que l'on doit adopter. En règle générale, si l'on se tient notamment à la méthodologie de Box et Jenkins, si la série étudiée est issue d'un processus stationnaire, on cherche alors le meilleur modèle parmi la classe des processus stationnaires pour la représenter, puis on estime ce modèle. En revanche, si la série est issue d'un processus non stationnaire, on doit avant toutes choses, chercher à la « stationnariser », c'est-à-dire trouver une transformation stationnaire de ce processus.

Nous allons introduire dans cette partie les étapes suivies à l'application de cette approche :

### **Etape 1 : Détermination du niveau d'intégration d'une série temporelle**

On dit qu'une série est intégrée d'ordre  $d$ , noté  $I(d)$  où  $d$  désigne l'ordre d'intégration s'il convient de la différencier  $d$  fois avant qu'elle ne soit stationnaire, c'est-à-dire la série  $(x_t)$  sera intégrée d'ordre  $d$  si  $\Delta_{d-1}(x_t)$  n'est pas stationnaire tandis que  $\Delta_d(x_t)$  l'est.

Ainsi, on peut définir une classe de processus stochastiques qui ne satisfont pas les conditions de la stationnarité, mais dont la différence à l'ordre  $d$  satisfait elle les propriétés de la stationnarité.

## Etape 2 : Application du Test de Dickey-Fuller Augmenté (ADF)

Le test de Dickey-Fuller simple (1979) est un test de racine unitaire (ou de non-stationnarité) dont l'hypothèse nulle est la non-stationnarité d'un processus autorégressif d'ordre un. Ce test, comme déjà mentionné, ne concerne que les processus autorégressifs d'ordre un tandis que le test de Dickey-Fuller Augmenté a été prolongé afin de détecter la présence d'une racine unitaire pour les processus autorégressifs d'ordre  $p$ . La construction de ce test se base sur trois modèles :

- Modèle sans constante ni dérive temporelle
- Modèle avec constante et sans dérive temporelle
- Modèle avec constante et dérive temporelle

Ces équations peuvent également s'écrire :

$$\Delta x_t = \theta x_{t-1} - \sum_{j=2}^p \theta_j \Delta x_{t-j+1} + \varepsilon_t$$

$$\Delta x_t = \theta x_{t-1} - \sum_{j=2}^p \theta_j \Delta x_{t-j+1} + c + \varepsilon_t$$

$$\Delta x_t = \theta x_{t-1} - \sum_{j=2}^p \theta_j \Delta x_{t-j+1} + c + bt + \varepsilon_t$$

L'hypothèse nulle du test de ADF est l'hypothèse de la nullité du coefficient de  $X_{t-1}$ . Si l'hypothèse nulle n'est pas rejetée alors le processus n'est pas stationnaire.

## Etape 3 : Détermination du nombre de retards optimal

Si le test de stationnarité montre que les séries sont intégrées d'un même ordre, il y a alors risque de cointégration. On peut donc envisager l'estimation du modèle VECM. Pour ce faire, on commence par déterminer le nombre de retards optimal. Cette étape est considérée comme étant une étape cruciale dans l'estimation du modèle car une détermination adéquate du nombre de retards permet d'assurer que les résidus du modèle sont des bruits blancs.

Le nombre de retards  $p$  peut être déterminé à l'aide des critères d'information à savoir le critère d'AKAIKE (AIC), de SCHAWRZ (SC) et de Hannan-Quinn (HQ). On retient le retard  $p$  qui minimise les critères AIC et SC.

Les fonctions AIC( $p$ ), SC( $p$ ) et HQ( $p$ ) sont calculées de la manière suivante :

$$AIC(p) = \ln(|\Sigma_e|) + \frac{2k^2p}{n}$$

$$SC(p) = \ln(|\Sigma_e|) + \frac{k^2p \ln(n)}{n}$$

$$HQ(p) = \ln(|\Sigma_e|) + \frac{2k^2p \ln(n)}{n}$$

Avec :

$k$ = nombre de variables du système ;

$n$ =nombre d'observations i.e. longueur de la série observée ;

$p$ =nombre de retards ;

$\Sigma_e$ =matrice des variances covariances des résidus du modèle.

Ces critères, qui sont des estimateurs, comprennent deux termes : le premier est une fonction croissante de l'estimateur de la variance du bruit blanc ; il décroît lorsque l'ordre augmente pour s'approcher de la valeur théorique de cette variance ; le second déterministe, est un terme pénalisant la croissance de  $p$ .

#### **Etape 4 : Application du Test de cointégration**

La cointégration entre deux variables a été conceptualisée par Engle et Granger (1987). Cependant, la méthode d'Engle et Granger ne permet pas de distinguer plusieurs relations de cointégration. Ce n'est que quelques années plus tard que Johansen (1991) met au point une procédure capable de tester l'existence de  $(n-1)$  relations de cointégration entre  $n$  variables ( $n > 2$ ).

#### **Définition :**

Si deux variables  $x_t$  et  $y_t$  sont intégrés d'ordre 1 et la combinaison linéaire de ces deux variables est stationnaire, on dira alors que  $x_t$  et  $y_t$  sont cointégrées d'ordre  $C(1,1)$ .

La cointégration se définit comme une relation stable à long terme qui relie les variables tout en analysant leur dynamique à court terme. L'identification de cette

Chapitre 3 : Modélisation de la courbe des taux via le modèle de Nelson Siegel Dynamique : Modèle VECM relation lorsqu'elle existe et sa prise en compte dans l'estimation du modèle VECM permet d'éviter de se retrouver avec une régression fallacieuse.

Plusieurs approches sont utilisées afin de déterminer le nombre de relations de cointégrations, nous avons choisi de présenter l'approche de Johansen :

La méthode de cointégration de Johansen considère un modèle vectoriel autorégressif VAR(p) à M variables, p retards et T observations écrit sous la forme matricielle :

$$x_t = \mu + \sum_{j=1}^p \pi_j x_{t-j} + \varepsilon_t$$

Cette équation peut être réécrite sous la forme de différences :

$$\Delta x_t = \beta_1 \Delta x_{t-1} + \dots + \beta_{p-1} \Delta x_{t-p+1} + \pi x_{t-p} + \mu + \varepsilon_t$$

La matrice  $\pi$  peut s'écrire sous la forme  $\pi = \alpha\beta'$  où  $\alpha$  est une matrice (k,r) avec  $k < r$  qui contient les vitesses d'ajustement et  $\beta'$  est une matrice (r,k) comprenant les coefficients des relations de cointégration (relations de long terme). Chaque combinaison linéaire représente donc une relation de cointégration. Le rang de la matrice  $\pi$  détermine donc le nombre de relations de cointégration. Johansen (1988) propose un test fondé sur les valeurs propres de la matrice  $\pi$ . A partir de ces valeurs propres, on calcule la statistique suivante (appelée « trace statistic ») :

$$TR = -n \sum_{i=r+1}^k \log(1 - \lambda_i)$$

Avec  $\lambda_i$  est la  $i$ ème valeur propre de la matrice  $\pi$  et r est le rang de la matrice  $\pi$ .

Cette statistique suit une loi de probabilité (similaire à un  $\chi^2$ ) tabulée à l'aide de simulations par Johansen et Juselius (1990). Ce test de Johansen fonctionne par exclusion d'hypothèses alternatives :

- Si  $r=0$ , soit  $H_0: r=0$  vs  $H_1: r>0$ , il n'existe donc pas de relation de cointégration. On ne peut pas alors estimer un modèle VECM mais il est possible d'estimer un modèle VAR.
- Si  $rg(\pi)=r$ , il existe alors r relations de cointégration. Un modèle VECM peut être alors estimé.

### Etape 5 : Estimation des relations de long terme

Si on a :  $x_t$  et  $y_t$  sont I(1) alors on estime par MCO la relation de long terme :

$$y_t = ax_t + b + \varepsilon_t$$

Pour qu'il y ait cointégration, il faut que le résidu  $e_t$  issu de la régression soit stationnaire :

$$e_t = y_t - \hat{a}x_t - \hat{b} \text{ est I}(0).$$

Si le résidu est stationnaire, nous pouvons alors estimer un modèle appelé à correction d'erreur (MCE). L'emploi d'un modèle à correction d'erreur dans le cas de la cointégration permet d'obtenir des prévisions plus fiables que si on avait utilisé la relation de long terme car les résultats de l'estimation de cette relation sont faussés par la non-stationnarité des séries.

### Etape 6 : Estimation du modèle MCE et validation des tests usuels (autocorrélation et homoscédasticité des résidus)

Modèle à correction d'erreur :

Si on a deux séries cointégrées, on peut estimer le modèle à correction d'erreur (MCE) suivant :

$$\Delta y_t = \gamma \Delta x_t + \delta (y_{t-1} - ax_{t-1} - b) + v_t \text{ avec } \delta < 0$$

On peut remarquer que le paramètre  $\delta$  doit être négatif pour qu'il y ait un retour de  $Y_t$  à sa valeur d'équilibre de long terme qui est  $(ax_{t-1}+b)$ . Dans le cas contraire, la spécification de type MCE n'est pas valable. En effet, lorsque  $y_{t-1}$  est supérieur à  $(ax_{t-1}+b)$ , il n'y a une force de rappel vers l'équilibre que si  $\delta < 0$ .

Le MCE permet de modéliser conjointement les dynamiques de court terme (représentées par les variables en différence première) et de long terme.

## II. Application

Dans ce qui suit, tous les tests effectués ont été réalisés sous R.

### → Stationnarité des variables

Afin d'étudier la stationnarité des variables et de définir l'ordre de d'intégration, nous avons eu recours au test de Dickey-Fuller Augmenté (ADF) pour chaque variable.

```

> adf.test(beta[,1])

Augmented Dickey-Fuller Test

data: beta[, 1]
Dickey-Fuller = -2.2795, Lag order = 15, p-value = 0.46
alternative hypothesis: stationary

> adf.test(beta[,2])

Augmented Dickey-Fuller Test

data: beta[, 2]
Dickey-Fuller = -2.1096, Lag order = 15, p-value = 0.5319
alternative hypothesis: stationary

> adf.test(beta[,3])

Augmented Dickey-Fuller Test

data: beta[, 3]
Dickey-Fuller = -2.3766, Lag order = 15, p-value = 0.4189
alternative hypothesis: stationary

```

Figure 15: Test de stationnarité sur les séries

```

> adf.test(diff(beta[,1]))

Augmented Dickey-Fuller Test

data: diff(beta[, 1])
Dickey-Fuller = -14.947, Lag order = 15, p-value = 0.01
alternative hypothesis: stationary

warning message:
In adf.test(diff(beta[, 1])) : p-value smaller than printed p-value
> adf.test(diff(beta[,2]))

Augmented Dickey-Fuller Test

data: diff(beta[, 2])
Dickey-Fuller = -14.816, Lag order = 15, p-value = 0.01
alternative hypothesis: stationary

warning message:
In adf.test(diff(beta[, 2])) : p-value smaller than printed p-value
> adf.test(diff(beta[,3]))

Augmented Dickey-Fuller Test

data: diff(beta[, 3])
Dickey-Fuller = -14.816, Lag order = 15, p-value = 0.01
alternative hypothesis: stationary

warning message:
In adf.test(diff(beta[, 3])) : p-value smaller than printed p-value

```

Figure 16: Test de stationnarité sur les séries différenciées

En effectuant ce test sous R, nous avons trouvé que toutes les variables sont non stationnaires alors que leurs différences premières sont stationnaires. On conclut donc que toutes nos variables sont intégrées d'ordre 1 (I(1)) :

Variables	Niveau	Différence première
Beta0	Non stationnaire	Stationnaire
Beta1	Non stationnaire	Stationnaire
Beta2	Non stationnaire	Stationnaire

→ **Détermination du nombre de retards**

Cette étape est considérée comme étant une étape cruciale dans l'estimation du modèle car elle influence les estimations et le nombre d'équations de cointégration. Dans cette étape, nous allons déterminer le nombre de retards convenable.

Nous avons trouvé que le retard optimal est de 4 en utilisant le critère d'information Hannan-Quinn (HQ). En effet, ce critère adoucit quelque peu la sévérité de la fonction de pénalité de AIC et SC relativement à la croissance de la taille de l'échantillon tout en maintenant une forte convergence dans l'identification de l'ordre réel du modèle.

```
> VARselect( mc,9, type="const")
$selection
AIC(n)  HQ(n)  SC(n)  FPE(n)
      9      4      1      9

$criteria
      1      2      3      4      5
AIC(n) -4.151413e+01 -4.152685e+01 -4.152866e+01 -4.154061e+01 -4.154190e+01
HQ(n)   -4.150665e+01 -4.151377e+01 -4.150997e+01 -4.151632e+01 -4.151200e+01
SC(n)   -4.149317e+01 -4.149017e+01 -4.147627e+01 -4.147250e+01 -4.145807e+01
FPE(n)  9.346398e-19 9.228244e-19 9.211532e-19 9.102088e-19 9.090391e-19
      6      7      8      9
AIC(n) -4.154955e+01 -4.154811e+01 -4.154839e+01 -4.155310e+01
HQ(n)   -4.151404e+01 -4.150700e+01 -4.150167e+01 -4.150077e+01
SC(n)   -4.145000e+01 -4.143285e+01 -4.141742e+01 -4.140641e+01
FPE(n)  9.021150e-19 9.034074e-19 9.031550e-19 8.989144e-19
```

Figure 17: Détermination du nombre de retards

Nous passerons maintenant à la détermination du nombre d'équations de cointégration.

→ **Test de cointégration par la méthode de Johansen**

Le test de la trace a rejeté les hypothèses nulles qui supposent respectivement qu'il n'y a aucune relation de cointégration et qu'il y'a au plus une relation de cointégration. Ainsi, on peut conclure qu'il existe deux équations de cointégration entre les variables. Autrement dit, il existe deux relations de long terme entre nos variables.

```

> co=ca.jo(mc,K=4,type = "trace",ecdet = "const")
> summary(co)

#####
# Johansen-Procedure #
#####

Test type: trace statistic , without linear trend and constant in cointegration

Eigenvalues (lambda):
[1] 7.259471e-03 5.973305e-03 1.028748e-03 -9.711946e-19

Values of teststatistic and critical values of test:

      test 10pct 5pct 1pct
r <= 2 | 3.64 7.52 9.24 12.97
r <= 1 | 24.84 17.85 19.96 24.60
r = 0 | 50.62 32.00 34.91 41.07
    
```

Figure 18: Test de cointégration par la méthode de Johanson

Ces résultats nous permettent d'entamer le modèle vectoriel à correction d'erreur (VECM).

→ **Estimation du modèle VECM**

```

> vecm=cajor1s(co,r=2)
> vecm
$rlm

Call:
lm(formula = substitute(form1), data = data.mat)

Coefficients:
          x1.d          x2.d          x3.d
ect1      -0.07133      0.07106      0.09349
ect2      -0.06249      0.06219      0.08499
x1.d11     -1.17603      1.16271      1.50753
x2.d11     -0.99992      0.97227      1.43531
x3.d11     -0.10893      0.11878      0.04109
x1.d12     -1.03093      1.04841      1.57189
x2.d12     -0.73001      0.74322      1.16502
x3.d12     -0.20013      0.20481      0.27366
x1.d13     -0.74351      0.89593      0.70561
x2.d13     -0.55795      0.67125      0.54556
x3.d13     -0.11837      0.14765      0.09597
    
```

Figure 19: Estimation du modèles VECM

D'après ces résultats, l'estimation du modèle VECM nous donne l'équation suivante :

$$\begin{pmatrix} \Delta\beta_t^0 \\ \Delta\beta_t^1 \\ \Delta\beta_t^2 \end{pmatrix} = \begin{pmatrix} -0.071 & -0.0625 \\ 0.071 & 0.062 \\ 0.093 & 0.084 \end{pmatrix} \begin{pmatrix} ect1 \\ ect2 \end{pmatrix} + \begin{pmatrix} -1.176 & -0.999 & -0.108 & -1.03 & -0.73 & -0.2 & -0.743 & -0.557 & -0.118 \\ 1.162 & 0.972 & 0.118 & 1.048 & 0.743 & 0.204 & 0.895 & 0.671 & 0.147 \\ 1.507 & 1.435 & 0.041 & 1.571 & 1.165 & 0.273 & 0.705 & 0.545 & 0.095 \end{pmatrix} * \begin{pmatrix} \Delta\beta_{t-1}^0 \\ \Delta\beta_{t-1}^1 \\ \Delta\beta_{t-1}^2 \\ \Delta\beta_{t-2}^0 \\ \Delta\beta_{t-2}^1 \\ \Delta\beta_{t-2}^2 \\ \Delta\beta_{t-3}^0 \\ \Delta\beta_{t-3}^1 \\ \Delta\beta_{t-3}^2 \end{pmatrix}$$

Avec :

ect1 : la première équation de cointégration.

ect2 : la deuxième équation de cointégration.

```

$beta
          ect1      ect2
x1.14    1.000000e+00  0.00000000
x2.14    2.220446e-16  1.00000000
x3.14    5.547708e-01 -0.62049073
constant -6.030381e-02  0.03239552
    
```

Figure 20: Les deux équations de cointégration

D'après ces résultats, les deux équations ECT1 et ECT2 peuvent s'écrire sous la forme :

$$ect1 = \beta_{t-4}^0 + 0.5547\beta_{t-4}^2 - 0.0603$$

$$ect2 = \beta_{t-4}^1 - 0.6204\beta_{t-4}^2 + 0.0324$$

### Validation du modèle :

#### → Test d'autocorrélation des résidus :

Afin de tester l'autocorrélation des résidus, nous utilisons le test de Durbin-Watson.

```

> dwtest(vecm$r1m)

Durbin-Watson test

data:  vecm$r1m
DW = 2.0002, p-value = 0.497
alternative hypothesis: true autocorrelation is greater than 0
    
```

Figure 21: Test d'autocorrélation des résidus

Nous remarquons que la p-value est supérieure à 5% donc nous acceptons l'hypothèse nulle et par conséquent les résidus ne sont pas corrélés.

#### → Test d'homoscédasticité :

Pour le test d'homoscédasticité, nous utilisons le test de Breusch-Pagan.

```

> bptest(vecm$r1m)

studentized Breusch-Pagan test

data:  vecm$r1m
BP = 17.289, df = 10, p-value = 0.06821
    
```

Figure 22: Test d'homoscédasticité

Nous remarquons que la p-value du test est supérieur à 5%, donc nous acceptons l'hypothèse nulle, c'est-à-dire l'homoscédasticité des résidus.

**Comparaison entre modèle VECM et VAR :**

Lors de l'estimation hors échantillon, les 15 années sont utilisées pour obtenir les paramètres initiaux. Par la suite, chaque nouvelle observation est ajoutée pour obtenir récursivement nos prévisions.

<b>Modèle</b>	<b>Date de la courbe</b>	<b>Moyenne des erreurs</b>	<b>Ecart type des erreurs</b>	<b>RMSE</b>
<b>DNS-FK</b>	1mois	0.083	0.040	0.092
	2mois	0.083	0.040	0.092
	3mois	0.094	0.050	0.106
	4mois	0.075	0.044	0.087
	5mois	0.077	0.045	0.089
	6mois	0.059	0.053	0.079
	7mois	0.065	0.053	0.083
	8mois	0.060	0.054	0.080
	9mois	0.064	0.047	0.079
	10mois	0.197	0.068	0.208
	11mois	0.226	0.082	0.239
	12mois	0.245	0.084	0.258
<b>VAR</b>	1mois	0.066	0.035	0.075
	2mois	0.048	0.039	0.062
	3mois	0.051	0.043	0.066
	4mois	0.058	0.042	0.071
	5mois	0.065	0.052	0.083
	6mois	0.097	0.065	0.117
	7mois	0.126	0.088	0.153
	8mois	0.134	0.082	0.156
	9mois	0.151	0.078	0.170
	10mois	0.136	0.113	0.176
	11mois	0.146	0.140	0.200
	12mois	0.169	0.142	0.219
<b>VECM</b>	1mois	0.067	0.032	0.074
	2mois	0.043	0.040	0.058
	3mois	0.041	0.045	0.060
	4mois	0.053	0.043	0.068
	5mois	0.074	0.047	0.087
	6mois	0.119	0.064	0.135
	7mois	0.151	0.078	0.169
	8mois	0.168	0.068	0.181
	9mois	0.185	0.062	0.195
	10mois	0.076	0.068	0.101
	11mois	0.079	0.083	0.114
	12mois	0.070	0.070	0.099

*Tableau 4: Comparaison entre les trois modèles : DNS-FK, VAR, VECM*

D'après le tableau, on se rend compte que les trois modèles permettent de faire une bonne prévision des taux. Le modèle VECM est celui qui permet d'obtenir les meilleures prévisions sur l'ensemble des horizons puisqu'il a la plus petite racine moyenne des erreurs au carré dans la majorité des échéances. On remarque également que la moyenne de l'erreur des prévisions est inférieure à dix points de base pour la majorité des échéances.

## **CHAPITRE 4**

### **Modélisation de la courbe des taux via le modèle NSD : réseau de neurone LSTM**

L'avènement des réseaux de neurones avec le Big Data s'est avéré d'une grande importance dans la prédiction des séries temporelles. Dans ce chapitre, nous avons opté pour le réseau de neurone de type LSTM permettant la prédiction des facteurs du modèle de Nelson Siegel Dynamique.

## **I-Introduction**

### **I-1 Historique**

L'Intelligence Artificielle, branche de l'Informatique fondamentale s'est développée avec pour objectif la simulation des comportements du cerveau humain. Les premières tentatives de modélisation du cerveau sont anciennes et précèdent même l'ère informatique. C'est en 1943 que Mc Culloch (neurophysiologiste) et Pitts (logicien) ont proposé les premières notions de neurone formel. Dans un article publié dans le journal Brain Theory, les deux chercheurs présentent leur théorie selon laquelle l'activation de neurones est l'unité de base de l'activité cérébrale.

En 1957, le Perceptron fut inventé. Il s'agit du plus ancien algorithme de Machine Learning, conçu pour effectuer des tâches de reconnaissance de patterns complexes. C'est cet algorithme qui permettra plus tard aux machines d'apprendre à reconnaître des objets sur des images. Cette approche dite connexionniste a atteint ses limites technologiques, compte tenu de la puissance de calcul de l'époque, mais aussi théoriques au début des années 70.

Il aura fallu attendre le début des années 2010, avec l'essor du Big Data et du traitement massivement parallèle, pour que les Data Scientists disposent des données et de la puissance de calcul nécessaires pour exécuter des réseaux de neurones complexes. En 2012, lors d'une compétition organisée par ImageNet, un Neural Network est parvenu pour la première fois à surpasser un humain dans la reconnaissance d'image. C'est la raison pour laquelle cette technologie est de nouveau au cœur des préoccupations des scientifiques. A présent, les réseaux de neurones artificiels ne cessent de s'améliorer et d'évoluer de jour en jour.

### **I-2 Définition**

Les réseaux de neurones artificiels sont un type de système mathématique, dont le modèle a été inspiré par la conception du cerveau humain. Ils structurent le système

Chapitre 4 : Modélisation de la courbe des taux via le modèle NSD : réseau de neurone LSTM  
en neurones qui sont connectés les uns aux autres. Chaque neurone reçoit un poids (coefficient) différent et chaque connexion entre neurones se voit également attribuer un tel poids.

En règle générale, un réseau de neurones repose sur un grand nombre de processeurs opérant en parallèle et organisés en tiers. Le premier tiers (input layer) reçoit les entrées d'informations brutes, un peu comme les nerfs optiques de l'être humain lorsqu'il traite des signaux visuels.

Par la suite, chaque tiers reçoit les sorties d'informations du tiers précédent. On retrouve le même processus chez l'homme, lorsque les neurones reçoivent des signaux en provenance des neurones proches du nerf optique. Le dernier tiers (output layer), quant à lui, produit les résultats du système.

Par le biais d'un algorithme, le réseau de neurones artificiels permet à l'ordinateur d'apprendre à partir de nouvelles données. L'ordinateur doté du réseau de neurones apprend à effectuer une tâche en analysant des exemples pour s'entraîner. Ces exemples ont préalablement été étiquetés afin que le réseau puisse savoir de quoi il s'agit.

Contrairement à d'autres types d'algorithmes, les réseaux de neurones ne peuvent pas être programmés directement pour effectuer une tâche. A la manière du cerveau en développement d'un enfant, la seule instruction qu'ils ont est d'apprendre.

Dans le cas de l'apprentissage non-supervisé, les données ne sont pas étiquetées. Le réseau de neurones analyse l'ensemble de données, et une fonction-coût lui indique dans quelle mesure il est éloigné du résultat souhaité. Le réseau s'adapte alors pour augmenter la précision de l'algorithme.

Enfin, avec la méthode de l'apprentissage renforcé, le réseau de neurones est renforcé pour les résultats positifs et sanctionné pour les résultats négatifs. C'est ce qui lui permet d'apprendre au fil du temps, de la même manière qu'un humain apprend progressivement de ses erreurs.

## **II-Modélisation des réseaux de neurones**

### **II-1-Structure d'un neurone**

#### **II-1-1 Architecture**

Un réseau de neurones est en général composé d'une succession de couches de neurones dont chacune prend ses entrées sur les sorties de la précédente.

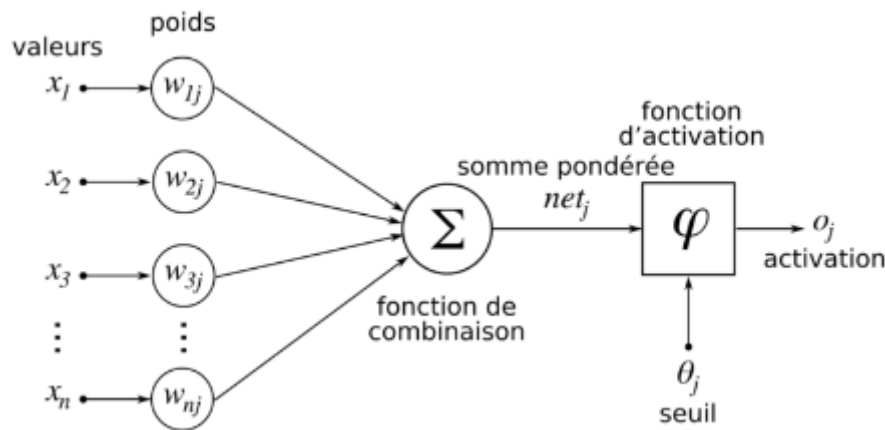


Figure 23: Structure d'un neurone artificiel j

Un neurone artificiel reçoit plusieurs stimuli via les poids. Il analyse ces informations et fournit un résultat en suivant.

- Chaque poids possède une valeur notée  $w_{ij}$ . Cette notation, la plus répandue dans la littérature scientifique, désigne le poids allant d'un neurone artificiel  $i$  au neurone formel  $j$ . Ces poids désignent l'influence de chaque neurone  $i$  sur  $j$ .
- Chaque poids transmet une information/ un stimulus provenant du neurone source  $i$  noté  $x_i$ .
- Ce stimulus (sa valeur), correspondant à l'information envoyé par le neurone source  $i$ , est modulé par le poids liant les neurones  $i$  et  $j$ . Mathématiquement cela se traduit par :  $w_{ij} * x_i$
- Ainsi, le neurone  $j$  reçoit autant de stimuli que de poids dont il fait la somme :

$$\sum w_{ij}x_i$$

Si l'on note  $n$  le nombre de neurones sources liées au neurone  $j$ , une notation mathématique plus complète serait :

$$\sum_{i=1}^n w_{ij} * x_i$$

C'est cette somme que le neurone artificiel  $j$  doit traiter. Pour ce faire, il utilise la fonction de combinaison.

### **II-1-2- Fonction de combinaison :**

Dans le domaine du traitement du signal, une fonction de transfert est « un modèle mathématique de la relation entre l'entrée  $x$  et la sortie  $y$  d'un système linéaire, le plus souvent invariant ».

Dans le domaine des réseaux de neurones, cette fonction peut aussi porter le nom de fonction de combinaison.

La fonction de combinaison  $p$  peut être formalisée comme étant une fonction vecteur-à-scalaire, notamment :

- Les réseaux de type MLP (multi-layer perceptron) calculent une combinaison linéaire des entrées, c'est-à-dire que la fonction de combinaison renvoie le produit scalaire entre le vecteur des entrées et le vecteur des poids : c'est le cas le plus simple. Mathématiquement :  $p = \sum_{i=1}^n w_{ij} * x_i$
- Les réseaux de type RBF (radial basis function) calculent la distance entre les entrées, c'est-à-dire que la fonction de combinaison renvoie la norme euclidienne du vecteur issu de la différence vectorielle entre les vecteurs d'entrées.

### **II-1-3 Fonction d'activation :**

La fonction d'activation «  $f$  » tel que  $a=f(p)$  (où  $a$  détermine la sortie) sert à déterminer l'état du neurone (en sorte). Généralement, cette fonction prend la même somme pondérée (vue précédemment) puis la transforme une fois de plus avant de la sortir finalement.

Plusieurs fonctions d'activation peuvent être utilisées en fonction des situations. Les plus courantes dans la littérature sont énumérées dans la figure ci-dessous :










Nom de la fonction	Relation d'entrée/sortie	Icône
seuil	$a = 0$ si $n < 0$ $a = 1$ si $n \geq 0$	
seuil symétrique	$a = -1$ si $n < 0$ $a = 1$ si $n \geq 0$	
linéaire	$a = n$	
linéaire saturée	$a = 0$ si $n < 0$ $a = n$ si $0 \leq n \leq 1$ $a = 1$ si $n > 1$	
linéaire saturée symétrique	$a = -1$ si $n < -1$ $a = n$ si $-1 \leq n \leq 1$ $a = 1$ si $n > 1$	
linéaire positive	$a = 0$ si $n < 0$ $a = n$ si $n \geq 0$	
sigmoïde	$a = \frac{1}{1+\exp^{-n}}$	
tangente hyperbolique	$a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$	
compétitive	$a = 1$ si $n$ maximum $a = 0$ autrement	

Figure 24: Exemples de fonctions de transfert

Des exemples classiques de fonctions d'activations sont :

- La fonction sigmoïde : elle est souvent utilisée car sa dérivée étant simple à calculer, elle permet des calculs simplifiés lors de l'apprentissage du réseau de neurones.
- La fonction tangente hyperbolique : c'est la version symétrique de la fonction sigmoïde.
- La fonction de Heaviside : permet l'obtention de sorties binaires, 1 si le seuil a été atteint sinon 0.

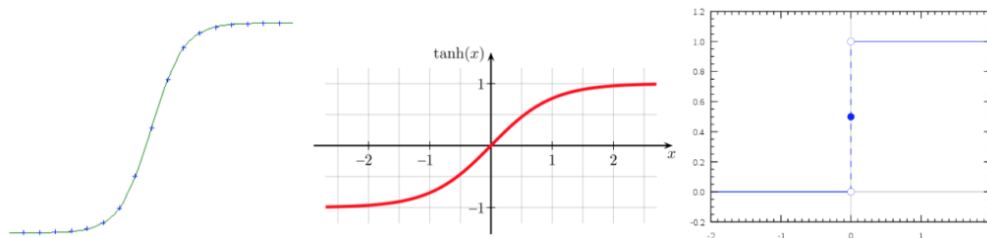


Figure 25: De gauche à droite, la fonction Sigmoïde, la fonction Tangente hyperbolique et la fonction Heaviside

## II-2 Modèle des réseaux de neurones

Maintenant que nous avons décrit les neurones, nous pouvons maintenant définir les réseaux de neurones. Un réseau de neurones est composé d'une série de couches de neurones, de sorte que tous les neurones de chaque couche se connectent aux neurones de la couche suivante.

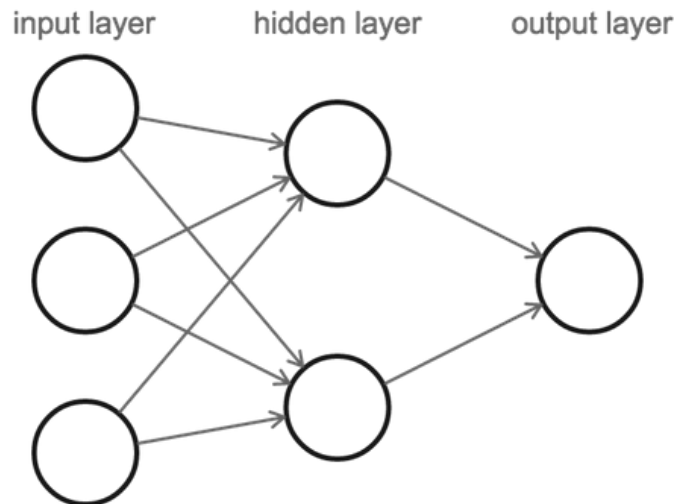


Figure 26: Un réseau de neurones à 2 couches

Lorsque nous comptons le nombre de couches dans un réseau de neurones, nous ne comptons que les couches avec des connexions entrantes (en omettant notre première couche d'entrée). Donc, la figure ci-dessus est un d'un réseau de neurones à 2 couches avec 1 couche cachée. Il contient 3 neurones d'entrée, 2 neurones dans sa couche cachée et 1 neurone de sortie. En effet, le calcul commence avec la couche d'entrée sur la gauche, à partir de laquelle nous passons les valeurs à la couche cachée, puis à son tour, la couche cachée enverra ses valeurs de sortie à la dernière couche, qui contient notre valeur finale.

Il peut sembler que les trois neurones d'entrée envoient des valeurs multiples parce que chacun d'entre eux est connecté aux deux neurones dans la couche cachée. Mais il n'y a vraiment qu'une seule valeur de sortie par neurone, elle est simplement copiée le long de chacune de ses connexions de sortie. Les neurones émettent toujours une valeur, quel que soit le nombre de neurones suivants.

## III-Types de réseaux de neurones

On distingue différents types de réseaux de neurones. En règle générale, les Neural Networks sont catégorisés en fonction du nombre d'épaisseurs qui séparent l'entrée de données de la production du résultat, en fonction du nombre de nœuds cachés du modèle, ou encore du nombre d'entrées et de sorties de chaque nœud.

### III-1 Réseaux de neurones « feed-forward »

C'est la variante la plus simple des réseaux de neurones. Dans ce cas, les informations passent directement de l'entrée aux nœuds de traitement puis aux sorties.

### III-2 Réseaux de neurones récurrents (RNN)

Contrairement aux réseaux de neurones « feed-forward », les réseaux de neurones récurrents, quant à eux, sauvegardent les résultats produits par les nœuds de traitement et nourrissent le modèle à l'aide de ces résultats. Autrement dit, l'architecture des RNN est basée sur des nœuds interconnectés. Ce mode d'apprentissage est un peu plus complexe.

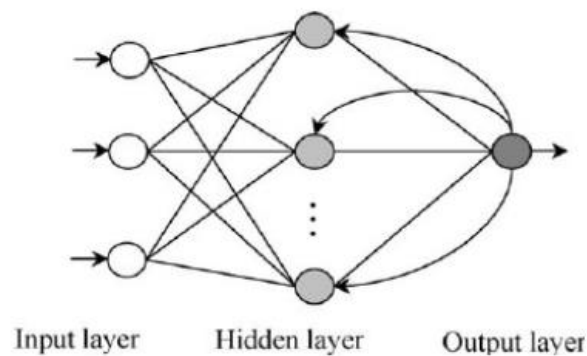


Figure 27: Architecture des RNN

### III-3 LSTM

Le réseau de neurones Long Short Term Memory (LSTM) est un cas particulier du réseau de neurones récurrent qui est capable de capturer les relations à long terme dans des séries chronologiques. En effet, les RNN ne prennent en compte que des dépendances très court terme. C'est pour remédier à ce problème qu'a été introduit en 1997 le modèle LSTM.

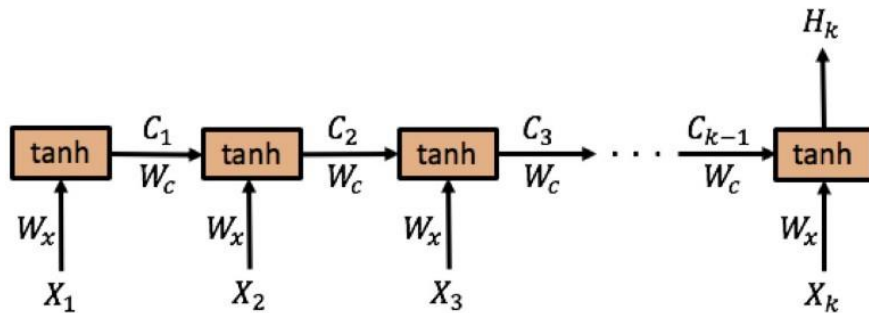
Les RNN souffrent du problème de la disparition du gradient ce qui entrave l'apprentissage de longues séquences de données. Les gradients contiennent les informations utilisées dans la mise à jour des paramètres RNN. Lorsque les gradients

Chapitre 4 : Modélisation de la courbe des taux via le modèle NSD : réseau de neurone LSTM deviennent de plus en plus petits, les mises à jour des paramètres deviennent non significatives, ce qui signifie qu'aucun véritable apprentissage n'est effectué.

Avant d'exposer l'architecture d'un modèle LSTM, il convient d'expliquer le problème de disparition du gradient.

**Problème de disparition du gradient (Vanishing gradients) :**

A titre de simplification, nous allons travailler dans la suite avec un simple RNN à couche cachée avec une seule séquence de sortie. Le réseau ressemble à ceci :



Le réseau a une séquence d'entrée de vecteurs  $(X_1, X_2, \dots, X_k)$  qu'on notera  $X_t$ . Les informations acquises sont représentées dans le vecteur d'état  $(C_1, C_2, \dots, C_{k-1})$  qu'on notera  $C_{t-1}$ . Le vecteur d'entrée  $X_t$  et le vecteur d'état  $C_{t-1}$  sont concaténés pour construire le vecteur d'entrée complet  $(C_{t-1}, X_t)$ .

Le réseau comporte 2 matrices de paramètres  $W_c$  et  $W_x$  reliant le vecteur d'entrée  $(C_{t-1}, X_t)$  à la couche cachée. Pour simplifier, nous omettons les vecteurs biais dans nos calculs et notons  $W = (W_c, W_x)$ . Le réseau produit un seul vecteur  $k$  au dernier pas du temps (les RNN peuvent être modélisés pour produire un vecteur à chaque pas du temps, mais nous utiliserons ce modèle plus simple à analyser).

Une fois que le RNN a généré le vecteur de prédiction  $H_k$ , nous calculons l'erreur de prédiction  $E_k$  et utilisons l'algorithme Back Propagation Through Time (BPTT) pour calculer le gradient  $\frac{\partial E_k}{\partial W}$ .

Le fait de retourner en arrière après le calcul de  $E_k$  sert à mettre à jour les paramètres du modèle en :  $W \rightarrow W - \alpha \frac{\partial E_k}{\partial W}$  où  $\alpha$  est le taux d'apprentissage (learning rate)

Calculons le gradient utilisé pour mettre à jour les paramètres du réseau pour une tâche d'apprentissage comportant  $k$  pas de temps. Nous avons :

$$\frac{\partial E_k}{\partial W} = \frac{\partial E_k}{\partial H_k} \frac{\partial H_k}{\partial C_k} \frac{\partial C_k}{\partial C_{k-1}} \dots \frac{\partial C_2}{\partial C_1} \frac{\partial C_1}{\partial W} = \frac{\partial E_k}{\partial H_k} \frac{\partial H_k}{\partial C_k} \left( \prod_{t=2}^k \frac{\partial C_t}{\partial C_{t-1}} \right) \frac{\partial C_1}{\partial W} \quad (1)$$

Puisque  $W = (W_c, W_x)$ ,  $C_t$  peut s'écrire comme :

$$C_t = \tanh(W_c C_{t-1} + W_x X_t)$$

Alors :

$$\frac{\partial C_t}{\partial C_{t-1}} = \tanh'(W_c C_{t-1} + W_x X_t) * \frac{d}{dC_{t-1}}(W_c C_{t-1} + W_x X_t) = \tanh'(W_c C_{t-1} + W_x X_t) W_c \quad (2)$$

Remplacer (2) dans (1) permet d'obtenir :

$$\begin{aligned} \frac{\partial E_k}{\partial W} &= \frac{\partial E_k}{\partial H_k} \frac{\partial H_k}{\partial C_k} \frac{\partial C_k}{\partial C_{k-1}} \dots \frac{\partial C_2}{\partial C_1} \frac{\partial C_1}{\partial W} \\ &= \frac{\partial E_k}{\partial H_k} \frac{\partial H_k}{\partial C_k} \left( \prod_{t=2}^k \tanh'(W_c C_{t-1} + W_x X_t) W_c \right) \frac{\partial C_1}{\partial W} \end{aligned}$$

La dernière expression a tendance à disparaître lorsque k est grand, ceci est dû à la dérivée de la fonction d'activation tanh qui est inférieure ou égale à 1, alors :  $\frac{\partial E_k}{\partial W} \rightarrow 0$ . La mise à jour des poids du réseau sera donc :  $W \rightarrow W - \alpha \frac{\partial E_k}{\partial W} \approx W$ . Cela montre qu'aucun apprentissage significatif ne sera fait.

Maintenant que nous avons expliqué le problème de disparition du gradient, passons à la description de l'architecture d'un LSTM. Un réseau LSTM a un vecteur d'entrée  $(X_t, H_{t-1})$  à l'instant t. L'état de la cellule du réseau est désigné par  $C_t$ . Les vecteurs de sortie passés à travers le réseau entre t et t+1 sont désignés par  $H_t$ .

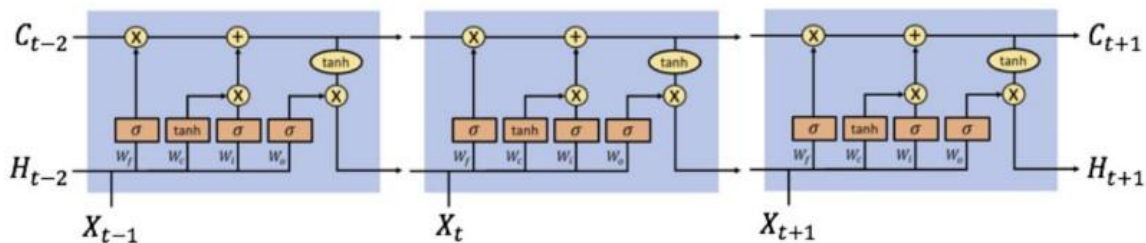
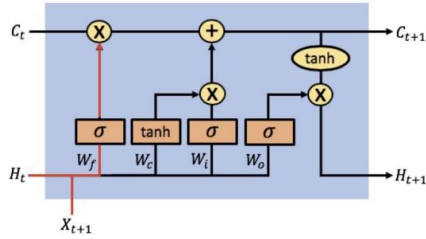


Figure 28: Cellules du réseau LSTM aux pas de temps t-1, t, t+1

Le LSTM a trois portes qui mettent à jour et contrôlent les états des cellules, à savoir la porte d'oubli (forget gate), la porte d'entrée (input gate) et la porte de sortie (output gate). Les portes utilisent des fonctions d'activation tangente hyperbolique et sigmoïdes.

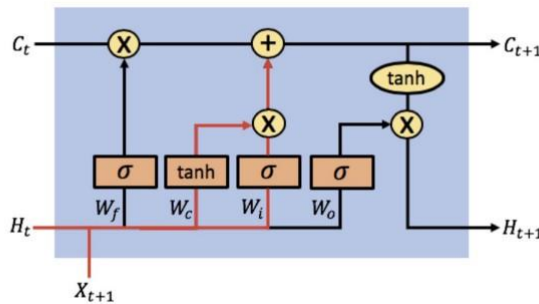
La porte d'oubli contrôle les informations à oublier dans la cellule, en fonction des nouvelles informations entrées dans le réseau. Rappelons que la sortie de la porte d'oubli est donnée par :  $f_{t+1} = \sigma(W_f \circ (H_t, X_{t+1}))$



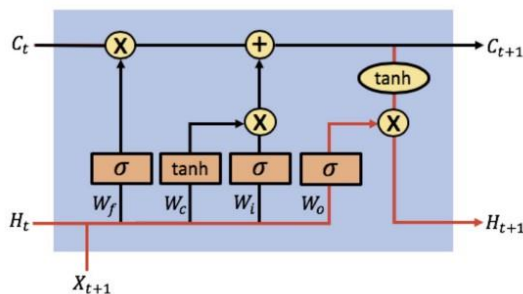
La porte d'entrée, quant à elle, contrôle les nouvelles informations qui seront codées dans l'état de la cellule, en fonction des nouvelles informations d'entrée. Notons que la sortie de la porte d'entrée est :  $\tanh(W_c(H_t, X_{t+1})) \circ \sigma(W_i(H_t, X_{t+1}))$  où :

- $\tilde{C}_{t+1} = \tanh(W_c(H_t, X_{t+1}))$  représente le vecteur d'information à ajouter.
- $i_{t+1} = \sigma(W_i(H_t, X_{t+1}))$  représente le vecteur pour quantité de changement.

On obtient alors :  $C_{t+1} = f_{t+1} \circ C_t + i_{t+1} \circ \tilde{C}_{t+1}$



La porte de sortie contrôle les informations codées dans l'état de la cellule qui sont envoyées au réseau en tant qu'entrée au pas de temps suivant via le vecteur de sortie  $H_{t+1}$  où  $H_{t+1} = o_t \circ \tanh(C_{t+1})$  avec :  $o_t = \sigma(W_o \circ (H_t, X_{t+1}))$



### Backpropagation dans le temps dans les réseaux LSTM :

Comme dans notre modèle RNN, nous supposons que notre réseau LSTM génère un seul vecteur de prédiction  $H_k$  au même pas de temps final. Les connaissances codées dans les vecteurs d'état  $C_t$  capturent les dépendances à long terme et les relations dans les données séquentielles. La longueur des séquences de données peut atteindre des centaines voire des milliers de pas de temps, ce qui rend extrêmement difficile l'apprentissage avec un RNN de base.

Nous calculons le gradient que nous utiliserions pour mettre à jour les paramètres du réseau (le calcul s'effectue sur  $k$  pas de temps) :

$$\frac{\partial E_k}{\partial W} = \frac{\partial E_k}{\partial H_k} \frac{\partial H_k}{\partial C_k} \frac{\partial C_k}{\partial C_{k-1}} \dots \frac{\partial C_2}{\partial C_1} \frac{\partial C_1}{\partial W} = \frac{\partial E_k}{\partial H_k} \frac{\partial H_k}{\partial C_k} \left( \prod_{t=2}^k \frac{\partial C_t}{\partial C_{t-1}} \right) \frac{\partial C_1}{\partial W} \quad (1)$$

Dans un LSTM, le vecteur d'état  $C_t$  a la forme suivante :

$$C_t = C_{t-1} \sigma(W_f(H_{t-1}, X_t)) \tanh(W_c(H_{t-1}, X_t)) \sigma(W_i(H_{t-1}, X_t))$$

On calcule maintenant la dérivée de  $C_t$  par rapport à  $C_{t-1}$  :

$$\frac{\partial C_t}{\partial C_{t-1}} = \sigma(W_f(H_{t-1}, X_t)) + \frac{d}{dC_{t-1}} (\tanh(W_c(H_{t-1}, X_t)) \sigma(W_i(H_{t-1}, X_t)))$$

Pour simplifier, nous omettons le calcul de  $\frac{d}{dC_{t-1}} (\tanh(W_c(H_{t-1}, X_t)) \sigma(W_i(H_{t-1}, X_t)))$

Ceci n'a que peu d'importance pour notre preuve, car nous verrons que pour que les gradients ne disparaissent pas, il suffit que les activations de la porte d'oubli soient supérieures à 0.

Nous écrivons donc simplement :

$$\frac{\partial C_t}{\partial C_{t-1}} \approx \sigma(W_f(H_{t-1}, X_t)) \quad (2)$$

En remplaçant (2) dans (1), on obtient :

$$\frac{\partial E_k}{\partial W} = \frac{\partial E_k}{\partial H_k} \frac{\partial H_k}{\partial C_k} \frac{\partial C_k}{\partial C_{k-1}} \dots \frac{\partial C_2}{\partial C_1} \frac{\partial C_1}{\partial W} = \frac{\partial E_k}{\partial H_k} \frac{\partial H_k}{\partial C_k} \left( \prod_{t=2}^k \sigma(W_f(H_{t-1}, X_t)) \right) \frac{\partial C_1}{\partial W}$$

L'équation (2) signifie que le dégradé se comporte de la même manière que la porte d'oubli. Si cette dernière décide qu'un élément d'information doit être mémorisé, il sera ouvert et aura des valeurs plus proches de 1.

Pour simplifier, nous pouvons penser à l'action de la porte d'oubli comme :

$$\sigma(W_f(H_{t-1}, X_t)) \approx \vec{1}$$

ainsi :  $\frac{\partial C_t}{\partial C_{t-1}}$  ne tend pas vers 0.

Finalement  $\frac{\partial E_k}{\partial W}$  ne tend pas vers 0 et les gradients ne disparaissent pas.

En résumé, nous avons vu que les RNN souffraient de gradients en voie de disparition provoqués par de longues séries de multiplications de petites valeurs, diminuant les gradients et entraînant la dégénérescence du processus d'apprentissage.

Les LSTM résolvent le problème en créant une connexion entre les activations de la porte d'oubli et le calcul des gradients. Cette connexion crée un chemin pour le flux d'informations à travers la porte d'oubli pour les informations que le LSTM ne doit pas oublier.

Sur l'examen des méthodes d'apprentissage, nous allons choisir le modèle LSTM pour le reste de notre étude.

#### **IV- Architecture LSTM pour le modèle de Nelson-Siegel**

Le LSTM est souvent utilisé pour la classification des séquences temporelles (dans des problèmes tels que la reconnaissance etc.), mais peut être utilisé pour prédire les séries temporelles de manière similaire aux méthodes vues précédemment si la bonne architecture est sélectionnée. Similairement aux méthodes autorégressifs (où le choix des variables, leur transformation et le retard sont généralement les paramètres à déterminer), le choix de la taille, de la fonction d'activation, du nombre et taille des couches est cruciale. Il n'y a pas de formule précise pour le choix de ces paramètres, mais plusieurs stratégies peuvent être prises pour trouver l'ensemble optimal tel que le modèle de Larochelle et al (2009) décrit ci-dessus. Les paramètres les plus importants pour la prédiction des paramètres du modèle de Nelson Siegel sont :

##### **Batch Size :**

Dans l'architecture LSTM, il est important de choisir la taille d'un lot de données sur laquelle le modèle va effectuer son apprentissage. Le choix de la taille du lot est compris entre 32 et 512. Sans oublier qu'une grande taille du batch peut entraîner une dégradation de la performance du modèle vu que le modèle évalue le plus petit nombre de chaque lot de données, chose qui peut entraîner l'incapacité de généralisation et de prédiction. Dans notre modèle, on a décidé de travailler avec un Batch Size de 32.

**Taille :**

Dans l'architecture d'un LSTM, il est important de choisir la taille des données.

**Dropout :**

Le « Dropout » est une technique de régularisation utilisée dans les réseaux de neurones afin d'éviter le surapprentissage.

**Nombre d'époques :**

Un autre paramètre important d'un LSTM est le nombre d'époques d'entraînement : un nombre déterminant combien de fois tout l'échantillon passera par le modèle. Le nombre optimal d'époques nécessaire à la formation du modèle dépend de la vitesse de convergence vers le résultat. Il dépend également de la puissance du calcul.

**Nombre de couches cachées :**

Le mot « profond » dans la discipline de l'apprentissage en profondeur indique à quel point le système est profond, autrement dit combien de couches forment le réseau. Il y a généralement une couche d'entrée, la couche de sortie et les couches intermédiaires appelées hidden layers (c'est-à-dire non visibles dans les problèmes de prédiction classiques). Leur choix dépend de la complexité du problème.

**Fonction d'activation :**

Dans les réseaux de neurones, la fonction d'activation est une fonction attribuée à chaque nœud. Son choix dépend du problème à résoudre. Généralement, la fonction linéaire est plus appropriée pour une sortie continue tel est le cas dans le modèle de Nelson Siegel.

**Loss function :**

Loss fonction fait partie intégrante d'un réseau de neurones, car elle représente le prix d'inexactitude de ses prédictions. Comme décrit par Rosasco et al., les loss fonctions couramment utilisées dans des fins de régression sont :

- The square loss function:  $E(\hat{y}, y) = (\hat{y} - y)^2$
- The absolute value loss function:  $E(\hat{y}, y) = |\hat{y} - y|$

où  $y$  est la valeur réelle de la variable de réponse et  $\hat{y}$  son estimation.

Rappelons que dans le problème de régression dans le modèle de Nelson Siegel, la loss fonction utilisée était la moyenne quadratique de l'erreur.

### **Initialisation des poids :**

Il existe différentes techniques pour initialiser les poids d'un LSTM et le bon choix peut accélérer le processus d'apprentissage. Dans notre cas, une initialisation aléatoire a été faite.

### **V- Application et Résultats :**

Dans ce qui suit, nous avons utilisé une bibliothèque de python appelée Keras (pour plus de détails voir Annexe B). C'est une bibliothèque dédiée réseaux de neurones qui permet l'apprentissage et la prédiction.

La méthode qu'on a suivie est de construire un LSTM pour chaque facteur beta du modèle de Nelson Siegel Dynamique. Ainsi, l'architecture adoptée pour chaque LSTM contient une couche d'entrée, une couche cachée et une couche de sortie.

#### **Récapitulatif de l'architecture choisie :**

- Batch Size : 32
- Nombre d'époques : 300
- Couches cachées : 3 couches (200 neurones chacune)
- Fonction de perte : MAE
- Dropout : 0.1

#### **Préparation des données :**

Les réseaux de neurones sont sensibles à la préparation des données, c'est pourquoi les données seront bornées entre (le minimum de la série) et 1 (son maximum).

#### **Résultats de la modélisation par LSTM :**

On a opté pour une répartition 90/10 pour les données modélisation et test.

#### **Résultats de la modélisation par LSTM :**

- Facteur niveau :

<b>Erreur de validation</b>	<b>Erreur de prédiction (test)</b>
0,001	0.000438

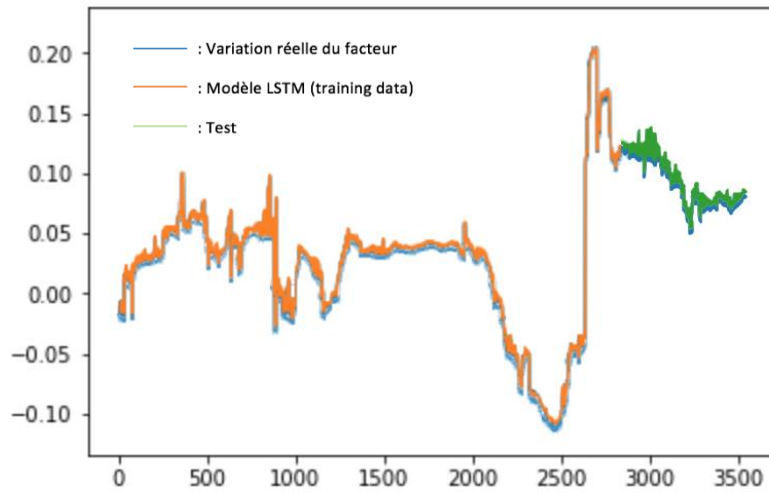


Figure 29: Tracé du facteur niveau

- Facteur pente :

Erreur de validation	Erreur de prédiction (test)
0,0148	0.09487

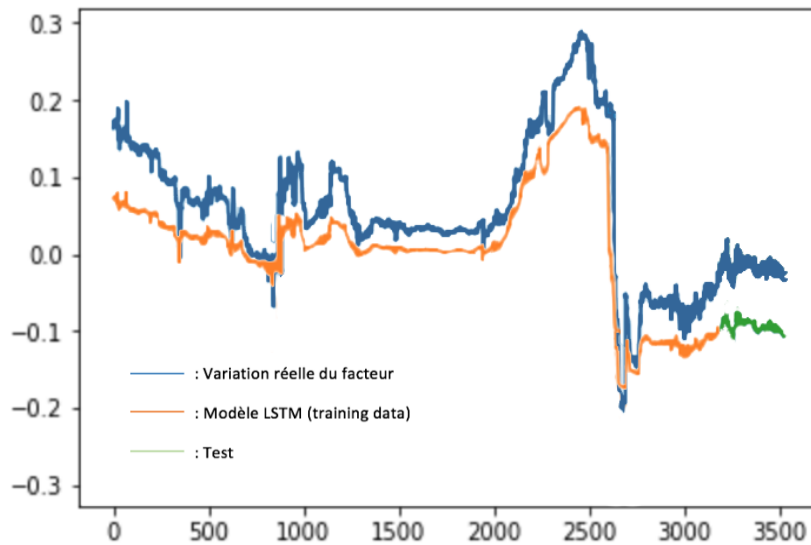


Figure 30: Tracé du facteur pente

- Facteur courbure :

Erreur de validation	Erreur de prédiction (test)
0,0102	0.0837

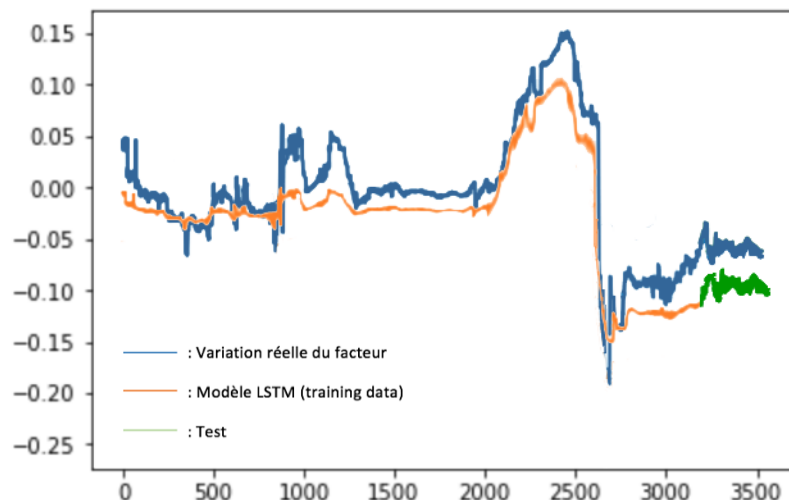


Figure 31: Tracé du facteur courbure

## VI. Comparaison entre VECM et LSTM :

Modèle	Date de la courbe	Moyenne des erreurs	Ecart type des erreurs	RMSE
VECM	1mois	0.067	0.032	0.074
	2mois	0.043	0.040	0.058
	3mois	0.041	0.045	0.060
	4mois	0.053	0.043	0.068
	5mois	0.074	0.047	0.087
	6mois	0.119	0.064	0.135
	7mois	0.151	0.078	0.169
	8mois	0.168	0.068	0.181
	9mois	0.185	0.062	0.195
	10mois	0.076	0.068	0.101
	11mois	0.079	0.083	0.114
	12mois	0.070	0.070	0.099
LSTM	1mois	0.047	0.042	0.063
	2mois	0.034	0.046	0.057
	3mois	0.031	0.065	0.054
	4mois	0.051	0.050	0.059
	5mois	0.064	0.052	0.078
	6mois	0.101	0.067	0.111
	7mois	0.243	0.098	0.261
	8mois	0.209	0.073	0.221
	9mois	0.228	0.071	0.238
	10mois	0.117	0.073	0.137
	11mois	0.101	0.089	0.134
	12mois	0.095	0.073	0.119

D'après le tableau, on se rend compte que le modèle LSTM ainsi que VECM permettent de faire de bonnes prévisions des taux. Le modèle LSTM est celui qui permet d'obtenir les meilleures prévisions sur le court terme puisqu'il a la plus petite racine moyenne des erreurs au carré. On remarque également que le modèle VECM est légèrement meilleur pour des horizons supérieurs à 6 mois.

# Conclusion

Globalement, disposer d'un outil de prévision efficace de la structure par terme des taux peut aider à augmenter la liquidité du marché à des prix précis, aidant ainsi les investisseurs et les émetteurs.

L'objectif du présent rapport est de faire une comparaison entre les différentes méthodes d'estimation du modèle de Nelson & Siegel Dynamique (DNS). Ce dernier modèle est largement utilisé dans les milieux universitaires et dans la pratique puisqu'il est connu pour sa performance de modéliser et prévoir le comportement des taux.

La première approche se base sur le modèle de Diebold & Li qui propose que la dynamique des facteurs suit un modèle VAR (1). Le problème avec cette procédure, c'est que nous incorporons les erreurs de lissage à nos prévisions. Pour remédier à cela, nous avons utilisé le Filtre de Kalman qui essaye de résoudre ce problème.

Concernant la deuxième approche, nous avons mis en place un modèle VECM qui nous a permis de dresser les différentes relations entre les paramètres à long et à court terme. Une comparaison entre les trois approches statistiques montre que les trois modèles permettent de faire de bonnes prévisions pour des horizons avec une priorité pour le modèle VECM.

Dans la troisième approche on a introduit une architecture de réseau de neurones récurrents artificiels (RNN) de type LSTM. Une analyse comparative entre ce modèle et VECM a montré que LSTM est meilleur à court terme. Cependant, des améliorations peuvent être introduites à ce modèle et qui peuvent porter principalement sur la répartition des données, le choix de la fonction d'activation, etc.

# Bibliographie

- [1] BIC (2005). Zero-coupon yield curves: technical documentation, Bank of International Settlements, Basle, Switzerland.
- [2] BOURBONNAIS, R. Econométrie, Dunod, 2004.
- [3] CHRISTENSEN, J.H.E., DIEBOLD, F.X. et RUDERBUSCH, G.D. (2009a). The affine arbitrage-free class of nelson-seigel term structure models. Manuscript
- [4] COX, J.C., INGERSOLL, J.E. et ROSS, S.A. (1985). A theory of the term structure of interest rates. *Econometrica*, 53.
- HARVEY, A.C. (1987). *Time Series Model*. Halsted Press.
- [5] DIEBOLD, F. J et LI, C. (2006). Forecasting the term structure of government bond yields. *Journal of Econometrics*, 130(2) :337-364.
- [6] François, P. (2005). *Les produits dérivés financiers*. Dunod.
- [7] GOURLAOUEN, J. (1997). La structure par terme des taux d'intérêts. *Economica*.
- [8] HULL, J. *Options, futures and other derivatives* (8th edition).
- [9] KALMAN, R.E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, march.
- [10] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 2016.
- [11] MOUARAF J (2018), *Mémoire Finance, Modélisation de la courbe des taux et application à la gestion obligataire*, INSEA
- [12] Yen-Ming Chiang, Li-Chiu Chang, and Fi-John Chang. Comparison of static feedforward and dynamic-feedback neural networks for rainfall runoff modeling. *Journal of hydrology*, 290(3):297-311, 2004

# Annexe A : Filtre de Kalman

## Histoire du filtre de Kalman et ses domaines d'application

Ce filtre a été proposé par R. Kalman en 1960 pour résoudre un problème de poursuite de trajectoire dans la préparation des missions Appollo. Ce filtre vise à estimer de façon optimale l'état d'un système linéaire et corriger les erreurs de manière itérative.

Le filtre de Kalman s'applique à des situations où l'état d'un système n'est connu que par des mesures entachées d'incertitude et obtenues à des intervalles de temps définis. En outre, on suppose que l'évolution à un moment futur du système dépende de l'état actuel et des effets inconnus qui peuvent être modélisés en tant que bruit blanc.

Le problème est qu'il n'y a pas de connaissance de l'état du système, il est à déduire des mesures. Le filtre de Kalman fournit une estimation des variables d'état et permet de prédire l'évolution du système.

La généralité de sa théorie le rend apte à aborder un bon nombre de problèmes tels que :

- La réception et discrimination de signaux radiophoniques
- La transmission de données numériques
- Estimation de la volatilité stochastique sur le marché des taux d'intérêts
- Inférer le taux court à partir d'une série de prix zéro-coupon
- Le traitement d'images numérisées

La prédiction des conditions météorologiques, de la démographie, de la consommation d'énergie électrique d'un pays, de la valeur d'un titre boursier, de l'orbite d'un corps céleste.

## Fondement mathématique du filtre de Kalman

Il existe deux équations fondamentales dans le filtre de Kalman, l'équation de mesure et l'équation de transition. L'équation de mesure concerne une variable non observée ( $X_t$ ) en une variable observable ( $Y_t$ ). En général, l'équation de mesure est de la forme :

$$Y_t = m_t * X_t + b_t + \varepsilon_t$$

L'équation de transition est basée sur un modèle qui permet à la variable non observée de changer dans le temps. En général, l'équation de transition est de la forme :

$$X_{t+1} = a_t * X_t + g_t + \theta_t$$

Avec  $m_t$ ,  $b_t$ ,  $a_t$ ,  $g_t$  les paramètres à estimer,  $\varepsilon_t$  un bruit gaussien dont la variance est de  $v_{1t}$  et  $\theta_t$ , un bruit gaussien dont la variance est de  $v_{2t}$ .

Au temps  $t-1$  de la simulation, il y a 3 étapes à suivre :

### Étape 1 : Estimation ou prévision

Nous calculons alors les deux prévisions suivantes :

- i)  $X_{t|t-1}$ , la prévision de  $X_t$  au temps  $(t-1)$ , soit l'espérance conditionnelle de  $X_t$  étant donné l'information disponible au temps  $(t-1)$
- ii)  $P_{t|t-1}$ , la prévision de  $P_t$  au temps  $t-1$ , avec  $P_t$  la variance de  $X_t$ .
- iii) Ces prévisions, qui sont des estimations conditionnelles non biaisées, se calculent comme suit :

iv) 
$$X_{t|t-1} = a_{t-1} * X_{t-1} + g_{t-1}$$

v) 
$$P_{t|t-1} = a_{t-1}^2 * P_{t-1} + v_{2,t-1}$$

### Étape 2 : La révision ou la correction

Au temps  $t$ , on dispose d'une nouvelle observation de  $Y$ , soit  $Y_t$ . On peut alors calculer l'erreur de prévision  $v_t$  :

$$v_t = Y_t - m_{t-1} * X_{t|t-1} + b_{t-1}$$

La variance de  $v_t$ , représentée par  $\psi_t$ , est de :

$$\psi_t = m_{t-1}^2 * P_{t|t-1} + v_{1,t-1}$$

On se sert de  $v_t$  et de  $\psi_t$  pour mettre à jour  $X_t$  et sa variance,  $P_t$  :

$$X_t = X_{t|t-1} + \frac{m_{t-1} * P_{t|t-1} * v_t}{\psi_t}$$

$$P_t = P_{t|t-1} + \frac{m_{t-1}^2 * P_{t|t-1}^2}{\psi_t}$$

Ces deux derniers estimateurs sont les estimateurs non biaisés conditionnellement qui minimisent la variance de ceux-ci. Le filtre de Kalman est donc optimal en ce sens qu'il est le meilleur estimateur dans la classe des estimateurs linéaires.

### Étape 3 : Estimation des paramètres

On recourt à la méthode du maximum de vraisemblance pour estimer les variances des erreurs du modèle. La fonction de vraisemblance est la suivante :

$$l = -\frac{1}{2} * \sum_t (\log(\psi_t) + \frac{v_t^2}{\psi_t})$$

Puis on passe au temps  $t+1$  et on refait cette procédure en trois étapes jusqu'à la période  $n$  souhaitée.

# Annexe B : Code LSTM sous Python

## Importation des bibliothèques et classes utilisées :

```
In [157]: from __future__ import absolute_import
import numpy as numpy
import numpy as np
import matplotlib.pyplot as plt
from pandas import read_csv
import math
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
from keras.models import load_model
```

## Chargement du Dataset :

La base « bc » contient l'ensemble des betas.

```
In [145]: dataframe = read_csv('bc.csv', usecols=[1], sep=';', engine='python')
dataset = dataframe.values
dataset = dataset.astype('float64')
```

Les LSTM sont sensibles à l'échelle des données d'entrée, en particulier lorsque les fonctions d'activation sigmoïde (par défaut) ou tanh sont utilisées. Il peut être judicieux de redimensionner les données dans une plage allant de 0 à 1, également appelée normalisation. Nous pouvons facilement normaliser nos données en utilisant la classe de [MinMaxScaler](#) de la bibliothèque [scikit-learn](#).

```
scaler = MinMaxScaler(feature_range=(0, 1))
dataset = scaler.fit_transform(dataset)
```

Le code ci-dessus divise les données en données d'apprentissage avec 90% des observations que nous pouvons utiliser pour former notre modèle, les 10% restants étant utilisés pour tester le modèle.

```
train_size = int(len(dataset) * 0.9)
test_size = len(dataset) - train_size
train, test = dataset[0:train_size,:], dataset[train_size:len(dataset),:]
```

Nous pouvons maintenant définir une fonction pour créer un nouvel ensemble de données, comme décrit ci-dessus.

```
def create_dataset(dataset, look_back=1):
    dataX, dataY = [], []
    for i in range(len(dataset)-look_back-1):
        a = dataset[i:(i+look_back), 0]
        dataX.append(a)
        dataY.append(dataset[i + look_back, 0])
    return numpy.array(dataX), numpy.array(dataY)
```

La fonction prend deux arguments: l'**ensemble de données** , qui est un tableau NumPy que nous voulons convertir en ensemble de données, et le **look\_back** , qui correspond au nombre de pas de temps précédents à utiliser comme variables d'entrée pour prédire la prochaine période, dans ce cas.

Utilisons cette fonction pour préparer le train et tester des jeux de données pour la modélisation.

```
: look_back = 5
trainX, trainY = create_dataset(train, look_back)
testX, testY = create_dataset(test, look_back)
```

Look-back=5 (données d'une semaine)

Le réseau LSTM s'attend à ce que les données d'entrée (X) reçoivent une structure de tableau spécifique sous la forme: [samples, time steps, features]

Actuellement, nos données sont sous la forme [samples, features]et nous formulons le problème comme un pas de temps pour chaque échantillon.

Donc pour changer cette structure et l'adapter à la structure demandée :

```
trainX, testX = create_dataset(train, look_back,
                                test, look_back)

]: trainX = numpy.reshape(trainX, (trainX.shape[0], 5, trainX.shape[1]))
testX = numpy.reshape(testX, (testX.shape[0], 5, testX.shape[1]))
```

Nous sommes maintenant prêts à concevoir et à adapter notre réseau LSTM à ce problème.

```
: def create_model():
    lstm_model = Sequential()
    # (batch_size, timesteps, data_dim)
    lstm_model.add(LSTM(200, batch_input_shape=(BATCH_SIZE, TIME_STEPS, x_t.shape[2]),
        dropout=0.0, recurrent_dropout=0.0, stateful=True, return_sequences=True,
        kernel_initializer='random_uniform'))
    lstm_model.add(Dropout(0.1))
    lstm_model.add(LSTM(200, dropout=0.0))
    lstm_model.add(Dropout(0.1))
    lstm_model.add(LSTM(200, dropout=0.0))
    lstm_model.add(Dropout(0.1))
    lstm_model.add(Dense(60,activation='relu'))
    lstm_model.add(Dense(20,activation='relu'))
    lstm_model.add(Dense(1,activation='sigmoid'))
    lstm_model.compile(loss='mean_squared_error', optimizer='adam')
    return lstm_model
```

Une fois que le modèle est ajusté, nous pouvons estimer ses performances de modélisation et de prédiction.

```
# make predictions
trainPredict = model.predict(trainX)
testPredict = model.predict(testX)
```

