Universidad de Cantabria

Facultad de Ciencias

El filtro de Kalman con aplicaciones en inversiones

(The Kalman filter with investment applications)

Trabajo de Fin de Grado
para acceder al

Grado en Matemáticas

Autor:
Marcos Cobo Carrillo

Directores:
Luis Alberto Fernández Fernández
Juan Carlos Fernández Troyano

Junio 2021

A mis padres, familiares y amigos.
A todas las personas que creyeron en mí, me apoyaron y me ayudaron a perseguir mis sueños.

Agradecimientos

En primer lugar, me gustaría dar un inmenso «gracias» a Luis Alberto Fernández y Juan Carlos Fernández por su duro trabajo y dedicación depositados en estos fantásticos nueve últimos meses. Gracias por confiar en mí y darme la oportunidad de trabajar juntos. En segundo lugar, me gustaría dar las gracias a mis padres, por ser mi principal apoyo y los formadores de la persona que soy, sin ellos y sin su amor y cariño jamás habría llegado hasta donde estoy. Gracias a mis familiares y amigos, los cuales me han acompañado durante toda mi vida y con los cuales he compartido mis mejores momentos. También estoy muy agradecido a mis compañeros de facultad, y en especial, a mi compañera Sofía. Ella ha sido quien ha sabido sacar lo mejor de mí y ver más allá de donde el resto de personas son capaces. Por último, quiero dar las gracias a todos los profesores que he tenido a lo largo de mi vida y a todas aquellas personas que han contribuido en mi formación como estudiante y como persona.

Marcos Cobo Carrillo, El Astillero, 2021.

Resumen

El filtro de Kalman es un algoritmo recursivo desarrollado por Rudolf E. Kalman en 1960 que sirve para poder identificar el estado de un sistema dinámico lineal discreto, en presencia de ruido aleatorio. Fue un componente fundamental del sistema de guiado y estabilización del módulo incluido en el Apolo XI (primera nave tripulada en llegar a la Luna en 1969) y desde entonces, su interés y rango de aplicaciones se ha extendido a muchos otros campos. En este TFG deduciremos rigurosamente las ecuaciones que modelan el filtro de Kalman para, posteriormente, diseñar nuestro propio filtro de Kalman atendiendo al objetivo de filtrar y predecir los movimientos de un activo financiero, más en concreto, pares de divisas dentro del mercado FOREX. Desarrollaremos una estrategia fundamentada en el filtro de Kalman y comprobaremos los rendimientos obtenidos de operar con ella en distintos pares de divisas a lo largo de los últimos dos años. Finalmente, extraeremos algunas conclusiones del potencial que posee el filtro de Kalman dentro del campo de la economía, y en concreto, dentro del mundo de las inversiones.

Palabras clave: Filtro de Kalman, matriz de covarianza, error cuadrático medio, regularización de Tikhonov, análisis técnico, bandas de Bollinger, FOREX, Python, BackTesting.

Abstract

The Kalman filter is a recursive algorithm developed by Rudolf E. Kalman in 1960 to identify the state of a discrete linear dynamic system in the presence of random noise. It was a fundamental component of the guidance and stabilisation system of the Apollo XI lander (the first manned spacecraft to reach the Moon in 1969) and since then, its interest and range of applications has been extended to many other fields. In this project we will rigorously derive the equations that model the Kalman filter to, subsequently, design our own Kalman filter with the objective of filtering and predicting the movements of a financial asset, more specifically, currency pairs within the FOREX market. We will develop a strategy based on the Kalman filter and we will check the returns obtained from trading with it on different currency pairs over the last two years. Finally, we will draw some conclusions about the potential of the Kalman filter in the field of economics, and in particular, in the world of investments.

Keywords: Kalman filter, covariance matrix, mean squared error, Tikhonov regularization, Technical analysis, Bollinger bands, FOREX, Python, BackTesting.

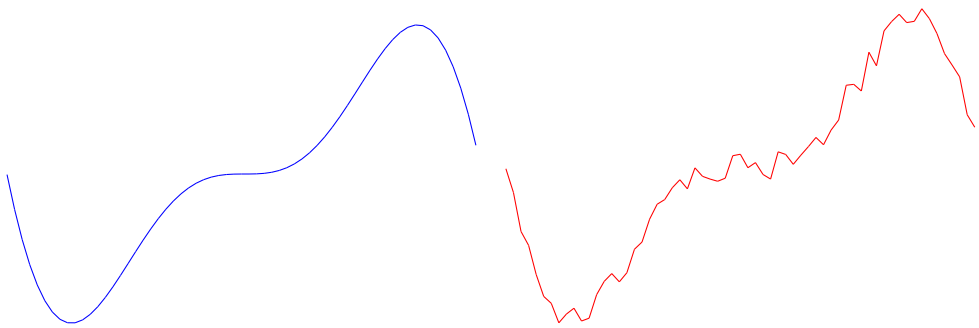
Índice general

Capítulo 1

Introducción

Un sistema es "un objeto complejo cuyas partes o componentes se relacionan con al menos alguna de las demás componentes", ya sea de manera conceptual o material. Podemos entender así un sistema como "una combinación de elementos que actúan juntos y realizan un objetivo determinado". Los sistemas dinámicos son aquellos cuyo estado evoluciona con el tiempo. Los sistemas físicos en situación no estacionaria son ejemplos de sistemas dinámicos, pero también existen modelos económicos, matemáticos y de otros tipos que son sistemas abstractos y, a su vez, dinámicos.

En los modelos ideales no existen perturbaciones, pero, en la vida real, éstas tienen un gran impacto en los sistemas, sobre todo, en los que son a nivel industrial o de gran complejidad y precisión.



(a) Sistema (b) Sistema perturbado

Figura 1.1: Sistema vs Sistema perturbado

El interés por la estimación del estado de un sistema perturbado es muy antiguo y ha sido objetivo de muchos pensadores a lo largo de la historia. En la Edad Moderna, Newton desarrolló la Teoría de la Mecánica y la Gravedad Universal, lo que dió paso al desarrollo de modelos dinámicos. Un siglo después, fue Gauss quien determinó la órbita de cuerpos celestes utilizando el método de "Mínimos Cuadrados". Además de eso, años después desarrolló una variante recursiva de este método que permitía corregir un estimador previo tras una nueva observación. A partir de ese momento, el método diseñado por Gauss se convirtió en la base de numerosas teorías y técnicas de estimación. Es ya a mediados del siglo XX, cuando se comienza a aplicar en el campo de los procesos estocásticos. En ese periodo se dan importantes desarrollos tecnológicos, como la cibernética, el desarrollo de radares o la planificación de trayectorias y guiado de misiles. En plena Guerra Fría, durante la llamada "Era Espacial", la teoría de control cobró muchísima importancia. Fue entonces cuando

Rudolf E. Kalman, en la década de los 60, desarrolló su algoritmo.

Rudolf E. Kalman fue un ingeniero húngaro, nacido en 1930, cuya familia emigró a Estados Unidos en plena Segunda Guerra Mundial. En 1953 obtuvo el título de ingeniero eléctrico por el MIT y siete años después, en 1960, Kalman publicó su famoso artículo "A new approach to linear filtering and prediction problems", en donde desarrollaba un conjunto de ecuaciones matemáticas que permitían calcular y conocer el estado de un proceso a partir de mediciones realizadas en presencia de ruido.
El modelo matemático inventado por Kalman es considerado, hoy en día, como una de las herramientas tecnológicas fundamentales que permitieron al hombre pisar la Luna bajo el Programa Apolo.

El filtro de Kalman es un estimador dinámico de mínimos cuadrados en sistemas lineales discretos gaussianos que utiliza una serie de mediciones observadas a lo largo del tiempo en presencia de ruido aleatorio para calcular el estado del sistema.
El filtro de Kalman se divide en dos fases: predicción y actualización. En el paso de predicción, el filtro de Kalman produce una estimación de las variables de estado actual junto con sus respectivos ruidos. Una vez se obtiene la siguiente medición (incluido el ruido aleatorio), estas estimaciones se actualizan utilizando un promedio ponderado, dando más peso a las estimaciones con mayor certeza.

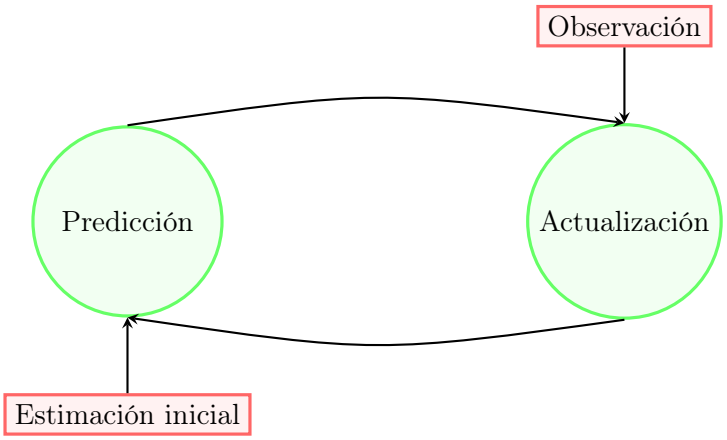


Figura 1.2: Esquema del filtro de Kalman

El algoritmo puede funcionar en tiempo real, utilizando únicamente las medidas de entrada actuales y el estado calculado previamente, a diferencia de otros estimadores recursivos, los cuales necesitan el historial de observaciones y/o estimaciones para calcular el estado del sistema.
Una ventaja adicional que posee el filtro de Kalman es que, si en algún paso de tiempo no poseemos la medición del estado del sistema, podemos obviar la fase de actualización en dicho paso de tiempo y avanzar a la fase de predicción del siguiente paso.

En la actualidad, el filtro de Kalman posee numerosas aplicaciones en el sector de la tecnología. La aplicación más común es para la orientación, navegación y control de vehículos, especialmente coches autónomos, aviones, naves espaciales y barcos posicionados dinámicamente. Los filtros de Kalman también son empleados en el campo del control y planificación del movimiento robótico.
Además, el filtro de Kalman es un concepto ampliamente utilizado en el mundo de las finanzas.

En este trabajo nos centraremos en esto último, y diseñaremos un filtro de Kalman con el fin de predecir el movimiento de la cotización de un instrumento financiero en el mercado, más en concreto, dentro del mercado de divisas o FOREX.

En el siguiente capítulo comenzaremos profundizando en el desarrollo de las ecuaciones que modelan el filtro de Kalman para posteriormente, en el tercer capítulo, desarrollar nuestro propio filtro atendiendo al objetivo de predecir el movimiento de los distintos pares de divisas del mercado FOREX. En este contexto, la señal (cotización) se conoce con mucha precisión, por lo que cambiaremos el enfoque tradicional del filtro de Kalman: en vez de, a partir de la señal con ruido, extraer la señal verdadera, obtendremos una señal suavizada a raíz del vector de cotizaciones, el cual es bastante preciso. En el cuarto capítulo estudiaremos el comportamiento del filtro de Kalman en base a variaciones en los parámetros de éste para, seguidamente, en el quinto capítulo, diseñar e implementar en el lenguaje de programación Python una estrategia basada en el filtro de Kalman, considerando la sensibilidad de los parámetros estudiada, para operar en tiempo real dentro del mercado FOREX. Finalmente, a través de la plataforma de trading MetaTrader5 y el lenguaje de programación Python estudiaremos los rendimientos arrojados por esta estrategia en 5 pares de divisas a lo largo de los 2 últimos años. Podemos adelantar que, de los 5 pares en los que hemos estudiado el rendimiento de dicha estrategia, en 4 de ellos ha resultado un rendimiento acumulado significativamente positivo, lo que nos deja entrever el potencial de predicción que posee el filtro de Kalman.
Todos los scripts que hemos desarrollado en Python para la implementación del filtro de Kalman, la estrategia para operar en FOREX, las simulaciones en los pares de divisas y todos los resultados expuestos a lo largo del trabajo se encuentran adjuntos en el Apéndice A.

Capítulo 2

El filtro de Kalman

En este capítulo trataremos de comprender los fundamentos matemáticos subyacentes en las ecuaciones del filtro de Kalman. Las principales fuentes que hemos seguido a lo largo del capítulo son [4] y [5].

Antes de comenzar con el desarrollo de estas ecuaciones, debemos empezar preparando el marco de trabajo: El filtro de Kalman asume que el estado verdadero del sistema en el paso de tiempo k evoluciona desde el estado en el paso $k-1$ de acuerdo con:

$$x_k = F_k x_{k-1} + B_k u_k + w_k \tag{2.1}$$

donde:

- F_k es la matriz del modelo de transición entre estados, de dimensión $n \times n$, que se aplica al estado del sistema anterior x_{k-1}, de dimensión $n \times 1$.
- B_k es la matriz de control del modelo, de dimensión $n \times n$, que se aplica al vector de control u_k, de dimensión $n \times 1$. Aunque desarrollemos el modelo teniendo en cuenta este vector, en este trabajo nos centraremos en aplicar el filtro de Kalman a instrumentos financieros en los cuales no podemos influir, por lo que dicho vector de control será nulo, pero si quisiéramos aplicar el filtro de Kalman en otros sectores, como en la orientación de vehículos autónomos, el vector de control influiría en el estado del sistema, puesto que podríamos manipular éste con diversos componentes del sistema, como el acelerador, freno, etc.
- w_k es el vector del ruido del proceso, de dimensión $n \times 1$, que se supone que sigue una distribución normal de media cero con matriz de covarianzas Q_k, de dimensión $n \times n$.

El filtro de Kalman también asume que en el paso de tiempo k se hace una observación z_k, el cual es un vector de dimensión $m \times 1$ (no siempre se observan todas las componentes de x_k), del estado verdadero del sistema x_k de acuerdo con:

$$z_k = H_k x_k + v_k \tag{2.2}$$

donde:

- H_k es la matriz del modelo de observación, de dimensión $m \times n$, que se aplica al estado real del sistema x_k.

- v_k es el ruido de la observación, de dimensión $m \times 1$, el cual se supone que sigue una distribución normal de media cero con matriz de covarianzas R_k, de dimensión $m \times m$.

Se supone también que el estado inicial y los vectores de ruido en cada paso son independientes.

En lo que sigue, denotaremos por $\hat{x}_{n|m}$ a la estimación de la variable de estado del sistema, x, en el paso de tiempo o momento n, teniendo en cuenta $m \leq n$ observaciones. Una vez introducida la notación, podemos comenzar con el desarrollo de las ecuaciones del filtro de Kalman.

2.1. Etapa de predicción

El filtro de Kalman se divide en dos etapas: predicción y actualización. En la primera de las etapas se realiza una predicción del estado del sistema, dotando a dicha predicción de una medida de incertidumbre.
De acuerdo con lo visto anteriormente, poseemos un modelo de transición de estados del sistema, expresado a través de la matriz F_k, por lo que, el paso más intuitivo a la hora de dar una predicción del estado de dicho sistema sería aplicar al último vector de estado disponible el modelo de transiciones de estado. Matemáticamente resultaría la siguiente expresión:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1}$$

De acuerdo con el verdadero estado del sistema asumido por el filtro de Kalman, a través del vector de control podemos influir en el estado del sistema, haciendo este más predecible, por lo que, a la hora de predecir el estado del sistema, debemos de tener en cuenta las manipulaciones que hagamos sobre éste a través del vector de control, el cual intervendrá en el estado del sistema a través de la matriz B_k. De esta forma, el estado del sistema calculado en la etapa de predicción viene dado por:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} + B_k u_k \tag{2.3}$$

Una vez calculada la predicción del estado del sistema, hemos de dotar al filtro de una medida de incertidumbre sobre dicho cálculo. Esta medida será la matriz de covarianza, $P_{k|k-1}$, de la diferencia entre el estado real del sistema, x_k, y el estado predicho en la etapa actual, $\hat{x}_{k|k-1}$, es decir:

$$P_{k|k-1} = cov(x_k - \hat{x}_{k|k-1})$$

Antes de continuar con el desarrollo de la matriz $P_{k|k-1}$, prestemos atención a las siguientes proposiciones:

Proposición 2.1.1. *Dada $A \in \mathscr{M}_{n,m}(\mathbb{R})$ una matriz con coeficientes reales y $X \in \mathbb{R}^m$ una variable aleatoria. Entonces se verifica que $cov(AX) = Acov(X)A^T$.*

Demostración.

Si $A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{pmatrix}$ y $X = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix}$, tenemos que $AX = \begin{pmatrix} \sum_{i=1}^m a_{1i}x_i \\ \sum_{i=1}^m a_{2i}x_i \\ \vdots \\ \sum_{i=1}^m a_{ni}x_i \end{pmatrix}$, y por lo tanto:

$$cov(AX) = \begin{pmatrix} cov(\sum a_{1i}x_i, \sum a_{1i}x_i) & cov(\sum a_{1i}x_i, \sum a_{2i}x_i) & \cdots & cov(\sum a_{1i}x_i, \sum a_{ni}x_i) \\ cov(\sum a_{2i}x_i, \sum a_{1i}x_i) & cov(\sum a_{2i}x_i, \sum a_{2i}x_i) & \cdots & cov(\sum a_{2i}x_i, \sum a_{ni}x_i) \\ \vdots & \vdots & \ddots & \vdots \\ cov(\sum a_{ni}x_i, \sum a_{1i}x_i) & cov(\sum a_{ni}x_i, \sum a_{2i}x_i) & \cdots & cov(\sum a_{ni}x_i, \sum a_{ni}x_i) \end{pmatrix}$$

Por otra parte,

$$Acov(X)A^T =$$

$$= \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{pmatrix} \begin{pmatrix} cov(x_1, x_1) & cov(x_1, x_2) & \cdots & cov(x_1, x_m) \\ cov(x_2, x_1) & cov(x_2, x_2) & \cdots & cov(x_2, x_m) \\ \vdots & \vdots & \ddots & \vdots \\ cov(x_m, x_1) & cov(x_m, x_2) & \cdots & cov(x_m, x_m) \end{pmatrix} \begin{pmatrix} a_{11} & a_{21} & \cdots & a_{n1} \\ a_{12} & a_{22} & \cdots & a_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1m} & a_{2m} & \cdots & a_{nm} \end{pmatrix}$$

$$= \begin{pmatrix} \sum_{i=1}^m a_{1i}cov(x_i, x_1) & \sum_{i=1}^m a_{1i}cov(x_i, x_2) & \cdots & \sum_{i=1}^m a_{1i}cov(x_i, x_m) \\ \sum_{i=1}^m a_{2i}cov(x_i, x_1) & \sum_{i=1}^m a_{2i}cov(x_i, x_2) & \cdots & \sum_{i=1}^m a_{2i}cov(x_i, x_m) \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^m a_{ni}cov(x_i, x_1) & \sum_{i=1}^m a_{ni}cov(x_i, x_2) & \cdots & \sum_{i=1}^m a_{ni}cov(x_i, x_m) \end{pmatrix} \begin{pmatrix} a_{11} & a_{21} & \cdots & a_{n1} \\ a_{12} & a_{22} & \cdots & a_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1m} & a_{2m} & \cdots & a_{nm} \end{pmatrix}$$

$$= \begin{pmatrix} \sum_{j=1}^m a_{1j} \sum_{i=1}^m a_{1i}cov(x_i, x_j) & \sum_{j=1}^m a_{2j} \sum_{i=1}^m a_{1i}cov(x_i, x_j) & \cdots & \sum_{j=1}^m a_{nj} \sum_{i=1}^m a_{1i}cov(x_i, x_j) \\ \sum_{j=1}^m a_{1j} \sum_{i=1}^m a_{2i}cov(x_i, x_j) & \sum_{j=1}^m a_{2j} \sum_{i=1}^m a_{2i}cov(x_i, x_j) & \cdots & \sum_{j=1}^m a_{nj} \sum_{i=1}^m a_{2i}cov(x_i, x_j) \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{j=1}^m a_{1j} \sum_{i=1}^m a_{ni}cov(x_i, x_j) & \sum_{j=1}^m a_{2j} \sum_{i=1}^m a_{ni}cov(x_i, x_j) & \cdots & \sum_{j=1}^m a_{nj} \sum_{i=1}^m a_{ni}cov(x_i, x_j) \end{pmatrix}$$

Gracias a la ya conocida propiedad de la covarianza para vectores unidimensionales, $cov(ax, by) = ab\ cov(x, y)$, [6], tenemos que:

$$= \begin{pmatrix} cov(\sum a_{1i}x_i, \sum a_{1i}x_i) & cov(\sum a_{1i}x_i, \sum a_{2i}x_i) & \cdots & cov(\sum a_{1i}x_i, \sum a_{ni}x_i) \\ cov(\sum a_{2i}x_i, \sum a_{1i}x_i) & cov(\sum a_{2i}x_i, \sum a_{2i}x_i) & \cdots & cov(\sum a_{2i}x_i, \sum a_{ni}x_i) \\ \vdots & \vdots & \ddots & \vdots \\ cov(\sum a_{ni}x_i, \sum a_{1i}x_i) & cov(\sum a_{ni}x_i, \sum a_{2i}x_i) & \cdots & cov(\sum a_{ni}x_i, \sum a_{ni}x_i) \end{pmatrix}$$

$$= cov(AX)$$

□

Proposición 2.1.2. *Dadas $X, Y \in \mathbb{R}^n$ dos variables aleatorias, con X e Y son independientes.*

Entonces se verifica que $cov(X+Y) = cov(X-Y) = cov(X) + cov(Y)$.

Demostración. Por un lado tenemos que

$$cov(X+Y) = \begin{pmatrix} cov(x_1+y_1, x_1+y_1) & cov(x_1+y_1, x_2+y_2) & \cdots & cov(x_1+y_1, x_n+y_n) \\ cov(x_2+y_2, x_1+y_1) & cov(x_2+y_2, x_2+y_2) & \cdots & cov(x_2+y_2, x_n+y_n) \\ \vdots & \vdots & \ddots & \vdots \\ cov(x_n+y_n, x_1+y_1) & cov(x_n+y_n, x_2+y_2) & \cdots & cov(x_n+y_n, x_n+y_n) \end{pmatrix}$$

$$cov(X-Y) = \begin{pmatrix} cov(x_1-y_1, x_1-y_1) & cov(x_1-y_1, x_2-y_2) & \cdots & cov(x_1-y_1, x_n-y_n) \\ cov(x_2-y_2, x_1-y_1) & cov(x_2-y_2, x_2-y_2) & \cdots & cov(x_2-y_2, x_n-y_n) \\ \vdots & \vdots & \ddots & \vdots \\ cov(x_n-y_n, x_1-y_1) & cov(x_n-y_n, x_2-y_2) & \cdots & cov(x_n-y_n, x_n-y_n) \end{pmatrix}$$

Por otro lado tenemos que $cov(X) + cov(Y) =$

$$= \begin{pmatrix} cov(x_1,x_1) + cov(y_1,y_1) & cov(x_1,x_2) + cov(y_1,y_2) & \cdots & cov(x_1,x_n) + cov(y_1,y_n) \\ cov(x_2,x_1) + cov(y_2,y_1) & cov(x_2,x_2) + cov(y_2,y_2) & \cdots & cov(x_2,x_n) + cov(y_2,y_n) \\ \vdots & \vdots & \ddots & \vdots \\ cov(x_n,x_1) + cov(y_n,y_1) & cov(x_n,x_2) + cov(y_n,y_2) & \cdots & cov(x_n,x_n) + cov(y_n,y_n) \end{pmatrix}$$

Basta ver que $cov(x_i+y_i, x_j+y_j) = cov(x_i-y_i, x_j-y_j) = cov(x_i,x_j) + cov(y_i,y_j)$.

Teniendo en cuenta la conocida propiedad de la covarianza para vectores unidimensionales, [6]:

$$cov(ax_i + by_i, cx_j + dy_j) = ac\ cov(x_i,x_j) + ad\ cov(x_i,y_j) + bc\ cov(x_j,y_i) + bd\ cov(y_i,y_j)$$

y que, si X e Y son independientes, $cov(x_i,y_j) = cov(x_j,y_i) = 0$, resulta que:

$$cov(x_i+y_i, x_j+y_j) = cov(x_i,x_j) + cov(x_i,y_j) + cov(x_j,y_i) + cov(y_i,y_j) = cov(x_i,x_j) + cov(y_i,y_j)$$

$$cov(x_i-y_i, x_j-y_j) = cov(x_i,x_j) - cov(x_i,y_j) - cov(x_j,y_i) + cov(y_i,y_j) = cov(x_i,x_j) + cov(y_i,y_j)$$

□

Como consecuencia de las dos proposiciones anteriores, podemos deducir que:

$$\begin{aligned} P_{k|k-1} &= cov(x_k - \hat{x}_{k|k-1}) \\ &= cov(F_k x_{k-1} + B_k u_k + w_k - F_k \hat{x}_{k-1|k-1} - B_k u_k) \\ &= cov(F_k(x_{k-1} - \hat{x}_{k-1|k-1}) + w_k) \\ &= F_k cov(x_{k-1} - \hat{x}_{k-1|k-1}) F_k^T + cov(w_k) \\ &= F_k P_{k-1|k-1} F_k^T + Q_k \end{aligned} \tag{2.4}$$

En conclusión, durante la etapa de predicción calculamos el estado del sistema, además de la

incertidumbre generada durante el proceso, a través de las siguientes ecuaciones:

$$\begin{aligned} \hat{x}_{k|k-1} &= F_k \hat{x}_{k-1|k-1} + B_k u_k \\ P_{k|k-1} &= F_k P_{k-1|k-1} F_k^T + Q_k \end{aligned} \tag{2.5}$$

Destacar que las matrices F_k, B_k, $P_{k|k-1}$, $P_{k-1|k-1}$ y Q_k tienen dimensión $n \times n$, mientras que los vectores $\hat{x}_{k|k-1}$, $\hat{x}_{k-1|k-1}$ y u_k tienen dimensión $n \times 1$.

2.2. Etapa de actualización

Una vez calculada la predicción del estado del sistema, el filtro de Kalman, junto con la medición realizada de dicho estado, obtiene, en la etapa de actualización, el estado del sistema actualizado mediante una ponderación entre el estado predicho y el observado.
La idea de esta ponderación es dar "más peso" a aquella medida en la cual tengamos mayor confianza. Más concretamente. Si tenemos la creencia de que los sensores que registran el estado del sistema son muy precisos, daremos mayor importancia al estado observado que al estado predicho, luego, en la fase de actualización, obtendremos un valor más cercano a la medición que a la predicción. Por el contrario, si tenemos la creencia de que nuestro modelo es más preciso que los sensores que registran el estado del sistema, la ponderación entre la predicción y la medición será más cercana a la primera.
Para cuantificar la confianza que tenemos en la precisión de los sensores y la que tenemos en la exactitud del modelo emplearemos lo que se conoce como ganancia óptima de Kalman, K_k.
La idea fundamental de la etapa de actualización del filtro de Kalman es dar una media ponderada entre la predicción de la primera etapa y la medición registrada del sistema en función de la ganancia óptima de Kalman. Tomando un sistema unidimensional en el que el estado del mismo está determinado por un número real, denotando p como el estado del sistema predicho, z como el estado del sistema observado y k como la ganancia óptima de Kalman, matemáticamente, el cálculo de la actualización del estado, $\hat{x}$, consistiría en:

$$\hat{x} = (1 - k)p + kz$$

Extrapolando esta idea a una dimensión cualquiera y siguiendo la notación empleada a lo largo del capítulo, resulta que:

$$\hat{x}_{k|k} = (Id - K_k H_k)\, \hat{x}_{k|k-1} + K_k z_k$$

donde Id es la matriz identidad de dimensión $n \times n$, K_k la ganancia óptima de Kalman, con dimensión $n \times m$, que determinaremos más adelante, H_k la matriz del modelo de observación, con dimensión $m \times n$, y los vectores $\hat{x}_{k|k-1}$ y z_k son los vectores de estado y observación del sistema, con dimensiones $n \times 1$ y $m \times 1$ respectivamente.

Denotando $\tilde{y}_k := z_k - H_k \hat{x}_{k|k-1}$, conocido como residuo del pre-ajuste, obtenemos la siguiente

igualdad:

$$\begin{aligned}\hat{x}_{k|k} &= (Id - K_k H_k)\,\hat{x}_{k|k-1} + K_k z_k \\ &= \hat{x}_{k|k-1} + K_k z_k - K_k H_k \hat{x}_{k|k-1} \\ &= \hat{x}_{k|k-1} + K_k \left(z_k - H_k \hat{x}_{k|k-1}\right) \\ &= \hat{x}_{k|k-1} + K_k \tilde{y}_k\end{aligned} \tag{2.6}$$

Antes de continuar con el proceso de obtención de la ganancia óptima de Kalman, hemos de considerar las siguientes proposiciones:

Proposición 2.2.1. *Dada $S_k := H_k P_{k|k-1} H_k^T + R_k$, con dimensión $m \times m$. Entonces se verifica $P_{k|k} = P_{k|k-1} - K_k H_k P_{k|k-1} - P_{k|k-1} H_k^T K_k^T + K_k S_k K_k^T$.*

Demostración.

$$\begin{aligned}P_{k|k} &= cov(x_k - \hat{x}_{k|k}) = cov(x_k - (\hat{x}_{k|k-1} + K_k \tilde{y}_k)) \\ &= cov(x_k - (\hat{x}_{k|k-1} + K_k(z_k - H_k \hat{x}_{k|k-1}))) \\ &= cov(x_k - (\hat{x}_{k|k-1} + K_k(H_k x_k + v_k - H_k \hat{x}_{k|k-1}))) \\ &= cov(x_k - \hat{x}_{k|k-1} - K_k H_k x_k - K_k v_k + K_k H_k \hat{x}_{k|k-1}))) \\ &= cov((Id - K_k H_k)x_k - (Id - K_k H_k)\hat{x}_{k|k-1} - K_k v_k) \\ &= cov((Id - K_k H_k)(x_k - \hat{x}_{k|k-1}) - K_k v_k)\end{aligned}$$

Teniendo en cuenta que $(Id - K_k H_k)(x_k - \hat{x}_{k|k-1})$ y $K_k v_k$ son independientes (hemos supuesto que los vectores de ruido en cada paso son independientes), por la proposición 2.1.2, resulta que:

$$= cov((Id - K_k H_k)(x_k - \hat{x}_{k|k-1})) + cov(K_k v_k)$$

Gracias a la proposición 2.1.1:

$$\begin{aligned}&= (Id - K_k H_k)\ cov(x_k - \hat{x}_{k|k-1})\ (Id - K_k H_k)^T + K_k\ cov(v_k)\ K_k^T \\ &= (Id - K_k H_k) P_{k|k-1} (Id - K_k H_k)^T + K_k R_k K_k^T \\ &= P_{k|k-1} - K_k H_k P_{k|k-1} - P_{k|k-1} H_k^T K_k^T + K_k H_k P_{k|k-1} H_k^T K_k^T + K_k R_k K_k^T \\ &= P_{k|k-1} - K_k H_k P_{k|k-1} - P_{k|k-1} H_k^T K_k^T + K_k (H_k P_{k|k-1} H_k^T + R_k) K_k^T \\ &= P_{k|k-1} - K_k H_k P_{k|k-1} - P_{k|k-1} H_k^T K_k^T + K_k S_k K_k^T\end{aligned}$$

□

Proposición 2.2.2. *La esperanza matemática de la norma euclídea del residuo del pre-ajuste coincide con la traza de la matriz de covarianzas $P_{k|k}$, es decir, $E[\left\|x_k - \hat{x}_{k|k}\right\|^2] = Tr(P_{k|k})$.*

Demostración. Por un lado tenemos que $P_{k|k} := cov(x_k - \hat{x}_{k|k})$, es decir:

$$P_{k|k} = \begin{pmatrix} E[((x_k)_1 - (\hat{x}_{k|k})_1)((x_k)_1 - (\hat{x}_{k|k})_1)] & \cdots & E[((x_k)_1 - (\hat{x}_{k|k})_1)((x_k)_n - (\hat{x}_{k|k})_n)] \\ \vdots & \ddots & \vdots \\ E[((x_k)_n - (\hat{x}_{k|k})_n)((x_k)_1 - (\hat{x}_{k|k})_1)] & \cdots & E[((x_k)_n - (\hat{x}_{k|k})_n)((x_k)_n - (\hat{x}_{k|k})_n)] \end{pmatrix}$$

Por lo que:

$$Tr(P_{k|k}) = \sum_{i=1}^{n} E[((x_k)_i - (\hat{x}_{k|k})_i)^2]$$

Teniendo en cuenta la linealidad de la esperanza matemática, [7], resulta:

$$Tr(P_{k|k}) = E\left[\sum_{i=1}^{n}((x_k)_i - (\hat{x}_{k|k})_i)^2\right] = E[\left\|x_k - \hat{x}_{k|k}\right\|^2]$$

□

La idea de la ganancia óptima de Kalman es minimizar el error cuadrático medio entre el estado real del sistema y el estimado en la etapa de predicción, esto es, minimizar $E\left[||x_k - \hat{x}_{k|k}||\right]$.
Teniendo en cuenta la proposición 2.2.2, minimizar $E\left[\|x_k - \hat{x}_{k|k}\|\right]$ es equivalente a minimizar la traza de $P_{k|k}$, por lo que para obtener la ganancia óptima de Kalman, K_k, vamos a utilizar la ecuación:

$$\frac{\partial Tr(P_{k|k})}{\partial K_k} = 0$$

Gracias a la proposición 2.2.1, consideramos la siguiente igualdad:

$$\frac{\partial Tr(P_{k|k})}{\partial K_k} = \frac{\partial Tr(P_{k|k-1})}{\partial K_k} - \frac{\partial Tr(K_k H_k P_{k|k-1})}{\partial K_k} - \frac{\partial Tr(P_{k|k-1} H_k^T K_k^T)}{\partial K_k} + \frac{\partial Tr(K_k S_k K_k^T)}{\partial K_k} \quad (2.7)$$

Para resolver esta identidad debemos de hacer una breve introducción al cálculo matricial:

En matemáticas, el cálculo matricial es una notación especializada para hacer cálculo multivariable, especialmente con matrices. Recopila las diversas derivadas parciales de una sola función con respecto a muchas variables y/o de una función multivariable con respecto a una sola variable, en vectores y matrices que pueden tratarse como entidades únicas. Esto simplifica enormemente operaciones como encontrar el máximo o mínimo de una función multivariable y resolver sistemas de ecuaciones diferenciales.

En el cálculo matricial la variable independiente puede ser un escalar, un vector o una matriz, mientras que la variable dependiente también puede ser cualquiera de éstos. Cada situación diferente conducirá a un conjunto diferente de reglas, o un cálculo separado, usando el sentido más amplio del término. La notación matricial sirve como una forma conveniente de recopilar muchas derivadas de forma organizada.
Como primer ejemplo de operación en estos términos consideremos el gradiente del cálculo vectorial.

Para una función escalar de tres variables independientes, $f(x_1, x_2, x_3)$, el gradiente viene dado por la ecuación vectorial $\nabla f = \frac{\partial f}{\partial x_1}\hat{x}_1 + \frac{\partial f}{\partial x_2}\hat{x}_2 + \frac{\partial f}{\partial x_3}\hat{x}_3$, dónde $\hat{x}_i$ representa un vector unitario para x_i con $1 \leq i \leq 3$. Este tipo de derivada puede verse como la derivada de un escalar, f, con respecto a un vector, x, y su resultado puede escribirse en forma de vector: $\nabla f = \frac{\partial f}{\partial x} = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \frac{\partial f}{\partial x_3}\right]^T$.
Ejemplos más complicados incluyen la derivada de una función escalar con respecto a una matriz, conocida como matriz de gradiente, que recoge la derivada con respecto a cada elemento de la matriz en la posición correspondiente de la matriz resultante. En ese caso, el escalar debe ser una función de cada una de las variables independientes de la matriz. Otro posible ejemplo es, si tenemos un vector de dimensión n con variables dependientes, o funciones, de m variables independientes, podríamos considerar que la derivada del vector dependiente con respecto al vector independiente sería una matriz $m \times n$ que consta de todas las posibles combinaciones de derivadas.

Hay un total de nueve posibles combinaciones de derivadas usando escalares, vectores y matrices, pero son seis los tipos de derivadas que se pueden organizar en forma de matriz. Estos seis tipos se recopilan en la siguiente tabla:

Tipos	Escalar	Vector	Matriz
Escalar	$\frac{\partial y}{\partial x}$	$\frac{\partial \hat{y}}{\partial x}$	$\frac{\partial Y}{\partial x}$
Vector	$\frac{\partial y}{\partial \hat{x}}$	$\frac{\partial \hat{y}}{\partial \hat{x}}$	
Matriz	$\frac{\partial y}{\partial X}$		

Tabla 2.1: Tipos de derivada matricial

En la tabla anterior hemos utilizado el término "matriz" en su sentido más general, asumiendo que los vectores y escalares son simplemente matrices con una columna y con una columna y una fila respectivamente. Además, hemos utilizado letras minúsculas para denotar escalares, letras minúsculas con gorro para vectores y mayúsculas para matrices. A continuación definiremos cada una de las seis derivadas que aparecen en la tabla anterior.

- Derivadas con vectores:

  - Vector por escalar:
    La derivada de una función vectorial, $\hat{y} = [\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m]^T$, por un escalar de entrada, x, se define como:
    $$\frac{\partial \hat{y}}{\partial x} = \left[\frac{\partial \hat{y}_1}{\partial x}, \frac{\partial \hat{y}_2}{\partial x}, \ldots, \frac{\partial \hat{y}_m}{\partial x}\right]$$
  - Escalar por vector:
    La derivada de una función escalar, y, por un vector de entrada, $\hat{x} = [\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n]^T$, se define como:
    $$\frac{\partial y}{\partial \hat{x}} = \begin{bmatrix} \frac{\partial y}{\partial \hat{x}_1} \\ \frac{\partial y}{\partial \hat{x}_2} \\ \vdots \\ \frac{\partial y}{\partial \hat{x}_n} \end{bmatrix}$$
  - Vector por vector:
    La derivada de una función vectorial, $\hat{y} = [\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m]^T$, por un vector de entrada,

$\hat{x} = [\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n]^T$, se define como:

$$\frac{\partial \hat{y}}{\partial \hat{x}} = \begin{pmatrix} \frac{\partial \hat{y}_1}{\partial \hat{x}_1} & \frac{\partial \hat{y}_2}{\partial \hat{x}_1} & \cdots & \frac{\partial \hat{y}_m}{\partial \hat{x}_1} \\ \frac{\partial \hat{y}_1}{\partial \hat{x}_2} & \frac{\partial \hat{y}_2}{\partial \hat{x}_2} & \cdots & \frac{\partial \hat{y}_m}{\partial \hat{x}_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \hat{y}_1}{\partial \hat{x}_n} & \frac{\partial \hat{y}_2}{\partial \hat{x}_n} & \cdots & \frac{\partial \hat{y}_m}{\partial \hat{x}_n} \end{pmatrix}$$

- Derivadas con matrices:

  - Matriz por escalar:
    La derivada de una función matricial, Y, de dimensión $n \times m$, por un escalar entrada, x, se define como:
    $$\frac{\partial Y}{\partial x} = \begin{pmatrix} \frac{\partial y_{11}}{\partial x} & \frac{\partial y_{12}}{\partial x} & \cdots & \frac{\partial y_{1m}}{\partial x} \\ \frac{\partial y_{21}}{\partial x} & \frac{\partial y_{22}}{\partial x} & \cdots & \frac{\partial y_{2m}}{\partial x} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_{n1}}{\partial x} & \frac{\partial y_{n2}}{\partial x} & \cdots & \frac{\partial y_{nm}}{\partial x} \end{pmatrix}$$

  - Escalar por matriz:
    La derivada de una función escalar, y, por una matriz de variables independientes, X, con dimensión $n \times m$, se define como:
    $$\frac{\partial y}{\partial X} = \begin{pmatrix} \frac{\partial y}{\partial x_{11}} & \frac{\partial y}{\partial x_{12}} & \cdots & \frac{\partial y}{\partial x_{1m}} \\ \frac{\partial y}{\partial x_{21}} & \frac{\partial y}{\partial x_{22}} & \cdots & \frac{\partial y}{\partial x_{2m}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y}{\partial x_{n1}} & \frac{\partial y}{\partial x_{n2}} & \cdots & \frac{\partial y}{\partial x_{nm}} \end{pmatrix}$$

De aquí en adelante ésta será la notación que usaremos para el cálculo matricial, denominada notación de denominador, la cual recibe el nombre por escribir la derivada de un escalar con respecto a un vector como un vector columna. Para más detalle sobre el cálculo matricial véase [8].

Una vez detalladas las nociones básicas del cálculo matricial, en concreto, la de derivada de un escalar respecto de una matriz, desarrollaremos cada uno de los cuatro términos de la expresión (2.7) gracias a una serie de resultados que se muestran a continuación.

Proposición 2.2.3. *Sea $a \in \mathbb{R}$ y X una matriz de dimensión $n \times m$. Entonces se verifica $\frac{\partial a}{\partial X} = 0$, donde 0 representa la matriz de dimensión $n \times m$ con coeficientes nulos.*

Demostración. Aplicando la definición se demuestra de manera trivial:

$$\frac{\partial a}{\partial X} = \begin{pmatrix} \frac{\partial a}{\partial x_{11}} & \frac{\partial a}{\partial x_{12}} & \cdots & \frac{\partial a}{\partial x_{1m}} \\ \frac{\partial a}{\partial x_{21}} & \frac{\partial a}{\partial x_{22}} & \cdots & \frac{\partial a}{\partial x_{2m}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial a}{\partial x_{n1}} & \frac{\partial a}{\partial x_{n2}} & \cdots & \frac{\partial a}{\partial x_{nm}} \end{pmatrix} = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}$$

□

Gracias a esta proposición, como $Tr(P_{k|k-1}) \in \mathbb{R}$, tenemos que

$$\frac{\partial Tr(P_{k|k-1})}{\partial K_k} = 0 \tag{2.8}$$

Proposición 2.2.4. *Sean A y X dos matrices de dimensión $m \times n$ y $n \times m$ respectivamente, sin ser A una función de X. Entonces se verifica $\frac{\partial Tr(AX)}{\partial X} = \frac{\partial Tr(XA)}{\partial X} = A^T$.*

Demostración. En primer lugar observamos:

$$AX = \begin{pmatrix} \sum_{i=1}^{n} a_{1i}x_{i1} & \sum_{i=1}^{n} a_{1i}x_{i2} & \cdots & \sum_{i=1}^{n} a_{1i}x_{im} \\ \sum_{i=1}^{n} a_{2i}x_{i1} & \sum_{i=1}^{n} a_{2i}x_{i2} & \cdots & \sum_{i=1}^{n} a_{2i}x_{im} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{n} a_{mi}x_{i1} & \sum_{i=1}^{n} a_{mi}x_{i2} & \cdots & \sum_{i=1}^{n} a_{mi}x_{im} \end{pmatrix}$$

En segundo lugar observamos:

$$XA = \begin{pmatrix} \sum_{j=1}^{m} x_{1j}a_{j1} & \sum_{j=1}^{m} x_{1j}a_{j2} & \cdots & \sum_{j=1}^{m} x_{1j}a_{jn} \\ \sum_{j=1}^{m} x_{2j}a_{j1} & \sum_{j=1}^{m} x_{2j}a_{j2} & \cdots & \sum_{j=1}^{m} x_{2j}a_{jn} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{j=1}^{m} x_{nj}a_{j1} & \sum_{j=1}^{m} x_{nj}a_{j2} & \cdots & \sum_{j=1}^{m} x_{nj}a_{jn} \end{pmatrix}$$

Resulta así que:

$$Tr(AX) = Tr(XA) = \sum_{j=1}^{n}\sum_{i=1}^{m} a_{ji}x_{ij}$$

Devivando $Tr(AX) = Tr(XA)$ respecto de X obtenemos:

$$\frac{\partial Tr(AX)}{\partial X} = \frac{\partial Tr(XA)}{\partial X} = \begin{pmatrix} \frac{\partial Tr(AX)}{\partial x_{11}} & \frac{\partial Tr(AX)}{\partial x_{12}} & \cdots & \frac{\partial Tr(AX)}{\partial x_{1m}} \\ \frac{\partial Tr(AX)}{\partial x_{21}} & \frac{\partial Tr(AX)}{\partial x_{22}} & \cdots & \frac{\partial Tr(AX)}{\partial x_{2m}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial Tr(AX)}{\partial x_{n1}} & \frac{\partial Tr(AX)}{\partial x_{n2}} & \cdots & \frac{\partial Tr(AX)}{\partial x_{nm}} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{21} & \cdots & a_{m1} \\ a_{12} & a_{22} & \cdots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{mn} \end{pmatrix} = A^T$$

□

Como consecuencia de esta proposición, y teniendo en cuenta que la matriz $P_{k|k-1}$ es simétrica por definición:

$$\frac{\partial Tr(K_k H_k P_{k|k-1})}{\partial K_k} = P_{k|k-1}H_k^T \tag{2.9}$$

Proposición 2.2.5. *Sean A y X dos matrices de dimensión $n \times m$, sin ser A una función de X. Entonces se verifica $\frac{\partial Tr(AX^T)}{\partial X} = \frac{\partial Tr(X^T A)}{\partial X} = A$.*

Demostración. En primer lugar observamos:

$$AX^T = \begin{pmatrix} \sum_{i=1}^m a_{1i}x_{1i} & \sum_{i=1}^m a_{1i}x_{2i} & \cdots & \sum_{i=1}^m a_{1i}x_{ni} \\ \sum_{i=1}^m a_{2i}x_{1i} & \sum_{i=1}^m a_{2i}x_{2i} & \cdots & \sum_{i=1}^m a_{2i}x_{ni} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^m a_{ni}x_{1i} & \sum_{i=1}^m a_{ni}x_{2i} & \cdots & \sum_{i=1}^m a_{ni}x_{ni} \end{pmatrix}$$

Resulta así que:

$$Tr(AX^T) = Tr(X^TA) = \sum_{j=1}^n \sum_{i=1}^m a_{ij}x_{ij}$$

Derivando $Tr(AX^T) = Tr(X^TA)$ respecto de X obtenemos:

$$\frac{\partial Tr(AX^T)}{\partial X} = \frac{\partial Tr(X^TA)}{\partial X} = \begin{pmatrix} \frac{\partial Tr(AX^T)}{\partial x_{11}} & \frac{\partial Tr(AX^T)}{\partial x_{12}} & \cdots & \frac{\partial Tr(AX^T)}{\partial x_{1m}} \\ \frac{\partial Tr(AX^T)}{\partial x_{21}} & \frac{\partial Tr(AX^T)}{\partial x_{22}} & \cdots & \frac{\partial Tr(AX^T)}{\partial x_{2m}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial Tr(AX^T)}{\partial x_{n1}} & \frac{\partial Tr(AX^T)}{\partial x_{n2}} & \cdots & \frac{\partial Tr(AX^T)}{\partial x_{nm}} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{pmatrix} = A$$

□

Gracias a la proposición anterior podemos afirmar que:

$$\frac{\partial Tr(P_{k|k-1}H_k^TK_k^T)}{\partial K_k} = P_{k|k-1}H_k^T \tag{2.10}$$

Proposición 2.2.6. *Sean A y X dos matrices de dimensión $m \times m$ y $n \times m$ respectivamente, sin ser A una función de X. Entonces se verifica $\frac{\partial Tr(XAX^T)}{\partial X} = X(A + A^T)$.*

Demostración. En primer lugar observamos:

$$XAX^T = \begin{pmatrix} \sum_{i=1}^m x_{1i}\sum_{j=1}^m x_{1j}a_{ji} & \sum_{i=1}^m x_{2i}\sum_{j=1}^m x_{1j}a_{ji} & \cdots & \sum_{i=1}^m x_{ni}\sum_{j=1}^m x_{1j}a_{ji} \\ \sum_{i=1}^m x_{1i}\sum_{j=1}^m x_{2j}a_{ji} & \sum_{i=1}^m x_{2i}\sum_{j=1}^m x_{2j}a_{ji} & \cdots & \sum_{i=1}^m x_{ni}\sum_{j=1}^m x_{2j}a_{ji} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^m x_{1i}\sum_{j=1}^m x_{nj}a_{ji} & \sum_{i=1}^m x_{2i}\sum_{j=1}^m x_{nj}a_{ji} & \cdots & \sum_{i=1}^m x_{ni}\sum_{j=1}^m x_{nj}a_{ji} \end{pmatrix}$$

Resulta así que:

$$Tr(XAX^T) = \sum_{k=1}^n \sum_{i,j=1}^m x_{ki}x_{kj}a_{ji}$$

Teniendo en cuenta la siguiente igualdad:

$$\frac{\partial Tr(XAX^T)}{\partial x_{gh}} = \frac{\partial(\sum_{j=1}^m x_{gh}x_{gj}a_{hj} + \sum_{i=1}^m x_{gh}x_{gi}a_{ih})}{\partial x_{gh}} = \sum_{l=1}^m x_{gl}(a_{lh} + a_{hl})$$

Si derivamos $Tr(XAX^T)$ respecto de X obtenemos:

$$\frac{\partial Tr(XAX^T)}{\partial X} = \begin{pmatrix} \frac{\partial Tr(XAX^T)}{\partial x_{11}} & \frac{\partial Tr(XAX^T)}{\partial x_{12}} & \cdots & \frac{\partial Tr(XAX^T)}{\partial x_{1m}} \\ \frac{\partial Tr(XAX^T)}{\partial x_{21}} & \frac{\partial Tr(XAX^T)}{\partial x_{22}} & \cdots & \frac{\partial Tr(XAX^T)}{\partial x_{2m}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial Tr(XAX^T)}{\partial x_{n1}} & \frac{\partial Tr(XAX^T)}{\partial x_{n2}} & \cdots & \frac{\partial Tr(XAX^T)}{\partial x_{nm}} \end{pmatrix}$$

$$= \begin{pmatrix} \sum_{l=1}^m x_{1l}(a_{l1} + a_{1l}) & \sum_{l=1}^m x_{1l}(a_{l2} + a_{2l}) & \cdots & \sum_{l=1}^m x_{1l}(a_{lm} + a_{ml}) \\ \sum_{l=1}^m x_{2l}(a_{l1} + a_{1l}) & \sum_{l=1}^m x_{2l}(a_{l2} + a_{2l}) & \cdots & \sum_{l=1}^m x_{2l}(a_{lm} + a_{ml}) \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{l=1}^m x_{nl}(a_{l1} + a_{1l}) & \sum_{l=1}^m x_{nl}(a_{l2} + a_{2l}) & \cdots & \sum_{l=1}^m x_{nl}(a_{lm} + a_{ml}) \end{pmatrix} = X(A + A^T)$$

□

Finalmente, como consecuencia de esta proposición, obtenemos que:

$$\frac{\partial Tr(K_k S_k K_k^T)}{\partial K_k} = K_k \left(S_k + S_k^T\right) \tag{2.11}$$

De esta forma, considerando las identidades (2.7), (2.8), (2.9), (2.10) y (2.11) deducimos que:

$$\frac{\partial Tr(P_{k|k})}{\partial K_k} = K_k \left(S_k + S_k^T\right) - 2P_{k|k-1}H_k^T \tag{2.12}$$

Antes de continuar, hagamos una breve parada en el siguiente resultado:

Proposición 2.2.7. *Dada $S_k := H_k P_{k|k-1} H_k^T + R_k$, con dimensión $m \times m$. Entonces se verifica $S_k = S_k^T$.*

Demostración. Gracias a la ya conocida propiedad de la covarianza, $cov(x, y) = cov(y, x)^T$, [6], tenemos que $P_{k|k-1} = P_{k|k-1}^T$ y que $R_k = R_k^T$, por lo que:

$$S_k \;=\; H_k P_{k|k-1} H_k^T + R_k \;=\; H_k P_{k|k-1}^T H_k^T + R_k^T \;=\; S_k^T$$

□

Gracias a esta proposición y la igualdad (2.12), tenemos:

$$\frac{\partial Tr(P_{k|k})}{\partial K_k} = 2K_k S_k - 2P_{k|k-1}H_k^T \tag{2.13}$$

Por lo que, igualando la expresión a cero y despejando la ganancia óptima de Kalman, K_k, concluimos que:

$$K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{2.14}$$

En resumen, el estado del sistema en la fase de actualización se calcula mediante las siguientes

ecuaciones:

$$\begin{aligned} y_k &= z_k - H_k \hat{x}_{k|k-1} \\ S_k &= H_k P_{k|k-1} H_k^T + R_k \\ K_k &= P_{k|k-1} H_k^T S_k^{-1} \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k y_k \end{aligned} \tag{2.15}$$

Análogamente a la fase de predicción, una vez hallamos calculado el estado del sistema en la fase de actualización, debemos aportar una medida de incertidumbre a dicho resultado, es decir, debemos de calcular la matriz de covarianza, $P_{k|k}$, de la diferencia entre el estado real del sistema, x_k, y el estado actualizado, $\hat{x}_{k|k}$. Recordando la proposición 2.2.1:

$$P_{k|k} = P_{k|k-1} - K_k H_k P_{k|k-1} - P_{k|k-1} H_k^T K_k^T + K_k S_k K_k^T$$

y que:

$$K_k = P_{k|k-1} H_k^T S_k^{-1}$$

podemos simplificar la expresión de $P_{k|k}$ como se muestra a continuación:

$$\begin{aligned} P_{k|k} &= P_{k|k-1} - K_k H_k P_{k|k-1} - P_{k|k-1} H_k^T K_k^T + K_k S_k K_k^T \\ &= P_{k|k-1} - K_k H_k P_{k|k-1} - P_{k|k-1} H_k^T K_k^T + P_{k|k-1} H_k^T S_k^{-1} S_k K_k^T \\ &= P_{k|k-1} - K_k H_k P_{k|k-1} = (Id - K_k H_k) P_{k|k-1} \end{aligned} \tag{2.16}$$

Como conclusión, durante la etapa de actualización calculamos el estado del sistema, así como la incertidumbre generada durante el proceso, a través de las ecuaciones (2.15) y (2.16):

$$\begin{aligned} y_k &= z_k - H_k \hat{x}_{k|k-1} \\ S_k &= H_k P_{k|k-1} H_k^T + R_k \\ K_k &= P_{k|k-1} H_k^T S_k^{-1} \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k y_k \\ P_{k|k} &= (Id - K_k H_k) P_{k|k-1} \end{aligned} \tag{2.17}$$

Destacar que $\hat{x}_{k|k-1}$ y $\hat{x}_{k|k}$ son vectores de dimensión $n \times 1$, y_k y z_k son vectores de dimensión $m \times 1$, $P_{k|k-1}$, $P_{k|k}$ e Id son matrices de dimensión $n \times n$, H_k de dimensión $m \times n$, K_k de dimensión $n \times m$ y R_k y S_k de dimensión $m \times m$.

2.3. Ecuaciones del filtro de Kalman

En esta sección recogeremos las ecuaciones del filtro de Kalman que emplearemos en lo que resta de trabajo, deducidas en las secciones 2.1 y 2.2. Antes de detallar dichas ecuaciones, destacar que el filtro de Kalman puede escribirse de multitud de formas distintas, ya sea como una sola ecuación o con un gran número de ellas.

Un aporte diferente al nuestro, pero fundamentalmente equivalente, es el que da el autor de

una de las principales fuentes de este trabajo, [5], el cual describe el filtro a partir de las siguientes ecuaciones:

Predicción:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} + B_k u_k$$
$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k$$

Actualización:

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1}$$
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k (z_k - H_k \hat{x}_{k|k-1})$$
$$P_{k|k} = (Id - K_k H_k) P_{k|k-1} (Id - K_k H_k)^T + K_k R_k K_k^T$$

Por sencillez, tanto en la redacción como en el cálculo, en lo que resta de trabajo emplearemos el filtro de Kalman con ecuaciones (2.5) y (2.17). De este modo, recordando que $\hat{x}_{n|m}$ denota la estimación de la variable de estado del sistema x en el paso de tiempo o momento n, teniendo en cuenta $m \leq n$ observaciones, podemos escribir las ecuaciones del filtro de Kalman de la siguiente manera:

Predicción:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} + B_k u_k$$
$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k$$

Actualización:

$$y_k = z_k - H_k \hat{x}_{k|k-1}$$
$$S_k = H_k P_{k|k-1} H_k^T + R_k$$
$$K_k = P_{k|k-1} H_k^T S_k^{-1}$$
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k y_k$$
$$P_{k|k} = (Id - K_k H_k) P_{k|k-1}$$

Véase [4] para más detalle.

Capítulo 3

Diseño de un filtro de Kalman

En este capítulo diseñaremos un modelo para el filtro de Kalman con el fin de atender a nuestra necesidad de filtrar el movimiento de un activo financiero, separando el valor real de la cotización del ruido de ésta, provocado por factores externos que afectan al precio al corto plazo.

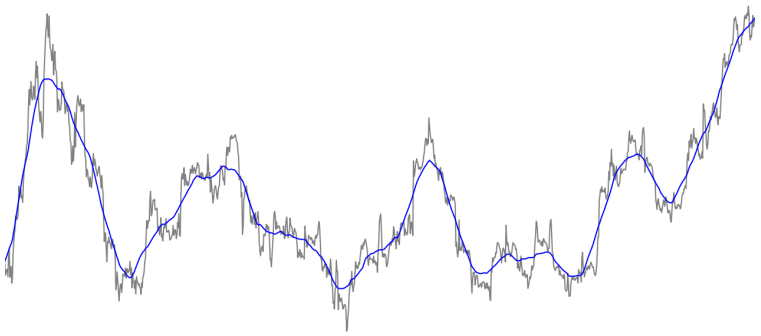


Figura 3.1: Filtrado del vector de cotizaciones, EURUSD

En la imagen anterior podemos ver las cotizaciones del par Euro - Dólar estadounidense en gris, y como éstas tienen una gran inestabilidad, por lo que recurrimos a herramientas como el filtro de Kalman para tener una visión más clara de las tendencias que sigue dicho par.
Para crear nuestro modelo, entenderemos las cotizaciones de un activo como una función de tiempo, es decir, $x(t)$, y emplearemos el desarrollo de Taylor para conocer los valores reales de dicha función, eliminando así gran cantidad de ruido. Siendo $x^{n)}(t_{k-1})$ la derivada n-ésima de x en el instante de tiempo t_{k-1} y δt el diferencial de tiempo entre estados consecutivos, el desarrollo de Taylor para nuestra función de cotizaciones $x(t)$ en el intervalo $[t_k, t_{k-1}]$ es el siguiente:

$$x(t_k) = x(t_{k-1}) + x'(t_{k-1})\delta t + \frac{1}{2!}x''(t_{k-1})\delta t^2 + \frac{1}{3!}x'''(t_{k-1})\delta t^3 + \cdots + \frac{1}{n!}x^{n)}(t_{k-1})\delta t^n + \cdots \quad (3.1)$$

Para diseñar un filtro de Kalman, debemos conocer el modelo de transición entre estados del sistema, así como la matriz F_k asociada a dicho modelo. También debemos conocer la matriz del modelo de observación, H_k, la matriz de control, B_k, y las matrices de covarianzas del ruido del

proceso, Q_k, y del ruido de la medición u observación, R_k.
A lo largo del capítulo iremos diseñando cada una de éstas matrices atendiendo a nuestras necesidades, para, finalmente, agrupar todas en nuestro filtro de Kalman.

3.1. Diseño de las matrices F y Q

En esta sección comenzaremos diseñando las matrices F_k y Q_k de nuestro modelo. Si estuviésemos ante un sistema físico, por ejemplo, un coche autónomo, podríamos modelar el filtro de Kalman en base a la 2ª Ley de Newton, ya que conoceríamos al detalle como evoluciona el estado del sistema entre instantes de tiempo consecutivos, pero en nuestro caso, como no conocemos las leyes que rigen nuestro sistema, vamos a describir las ecuaciones del mismo gracias al desarrollo de Taylor (3.1). Antes de comenzar con el diseño del modelo, destacar que, como estamos diseñando un filtro de Kalman para suavizar el movimiento de las cotizaciones de un activo financiero en un mercado, la matriz de control del sistema, B_k, será la matriz nula, puesto que, teóricamente, no podemos intervenir en el mercado y manipular las cotizaciones de éste. Como consecuencia, en este caso en concreto, el estado del sistema en la etapa de predicción quedará descrito por la siguiente ecuación:

$$x_k = F_k x_{k-1} + w_k$$

dónde F_k es la matriz del modelo de transición entre estados, que se aplica al estado del sistema anterior x_{k-1}, y w_k es el ruido del proceso, el cual se supone que proviene de una distribución normal de media cero con matriz de covarianzas Q_k.

A continuación, realizaremos la deducción de las matrices F_k y Q_k, ambas de dimensión $n \times n$, en filtros de dimensiones pequeñas para, finalmente, generalizarlo a una dimensión cualquiera.

3.1.1. Filtro de Kalman unidimensional

Empezaremos creando la matriz F_k para el modelo más sencillo, el unidimensional. En este caso, empleando el desarrollo de Taylor y suponiendo que no hay ruido en el proceso, el sistema queda descrito por el valor medido de la cotización, por lo que podemos representar éste a través de la siguiente ecuación:

$$x(t_k) = x(t_{k-1}) \tag{3.2}$$

Recordando que en nuestro filtro $x(t_k) = F \cdot x(t_{k-1})$, ya que hemos supuesto que no hay presencia de ruido en el proceso, tendríamos que:

$$F = 1 \tag{3.3}$$

Seguidamente, procedemos a diseñar la matriz Q_k para el modelo unidimensional. En este caso supondremos que sí existe ruido en el proceso, por lo que tendremos en cuenta el error o resto del desarrollo de Taylor. De esta forma, el sistema queda descrito a través de la siguiente ecuación,

donde sumamos el estado real del sistema y el error del modelo:

$$x(t_k) = x(t_{k-1}) + x'(\varepsilon_{k-1})\delta t \tag{3.4}$$

Tenemos así que, en el modelo unidimensional, el error de éste es $w_k = x'(\varepsilon_{k-1})\delta t$. Por otra parte:

$$Q_k = Var(w_k)$$

Si desarrollamos el término la matriz Q_k tenemos que:

$$Var(x'(\varepsilon_{k-1})\delta t) = E[(x'(\varepsilon_{k-1})\delta t))^2] - E[x'(\varepsilon_{k-1})\delta t]^2 = \delta t^2(E[x'(\varepsilon_{k-1})^2] - E[x'(\varepsilon_{k-1})]^2) = \delta t^2 Var(x'(\varepsilon_{k-1}))$$

Por lo tanto, la matriz Q_k quedaría expresada en función de un parámetro, $Var(x'(\varepsilon_{k-1}))$, el cual, a partir de este momento denotaremos por σ^2_{model}, haciendo alusión a la varianza del error del modelo o ruido del proceso. En conclusión, para el modelo de una dimensión tenemos que:

$$Q_k = \delta t^2 \cdot \sigma^2_{model} \tag{3.5}$$

3.1.2. Filtro de Kalman bidimensional

A continuación introduciremos una nueva variable de estado del sistema, $x'(t_k)$. De este modo, el sistema quedará descrito por $(x(t_k), x'(t_k))$. Utilizando el desarrollo de Taylor y definiendo δt como $\delta t := t_k - t_{k-1}$, nuestro sistema, sin presencia de ruido, quedaría descrito por las siguientes ecuaciones:

$$\begin{cases} x(t_k) = x(t_{k-1}) + x'(t_{k-1})\delta t \\ x'(t_k) = x'(t_{k-1}) \end{cases} \tag{3.6}$$

El anterior sistema lo podemos escribir como $\begin{pmatrix} x(t_k) \\ x'(t_k) \end{pmatrix} = F \begin{pmatrix} x(t_{k-1}) \\ x'(t_{k-1}) \end{pmatrix}$, por lo que la matriz F_k para el modelo de dimensión dos resulta:

$$F = \begin{pmatrix} 1 & \delta t \\ 0 & 1 \end{pmatrix} \tag{3.7}$$

Considerando el ruido del proceso, el sistema queda descrito por:

$$\begin{cases} x(t_k) = x(t_{k-1}) + x'(t_{k-1})\delta t + \frac{1}{2}x''(\varepsilon_{k-1})\delta t^2 \\ x'(t_k) = x'(t_{k-1}) + x''(\tilde{\varepsilon}_{k-1})\delta t \end{cases} \tag{3.8}$$

Con el objetivo de simplificar el desarrollo de la matriz Q_k, supondremos que $\varepsilon_{k-1} \simeq \tilde{\varepsilon}_{k-1}$. Tenemos, por lo tanto, que el ruido del proceso para dimensión dos es $w_k = \begin{pmatrix} w_k^1 \\ w_k^2 \end{pmatrix} =$

$\begin{pmatrix} \frac{1}{2}\delta t^2 \\ \delta t \end{pmatrix} x''(\varepsilon_{k-1})$. Partiendo de que:

$$Q_k = Cov(w_k) = \begin{pmatrix} Var(w_k^1) & Cov(w_k^1, w_k^2) \\ Cov(w_k^2, w_k^1) & Var(w_k^2) \end{pmatrix}$$

Si desarrollamos los términos de la matriz Q_k tenemos que:

$$Var(\tfrac{1}{2}x''(\varepsilon_{k-1})\delta t^2) = E[(\tfrac{1}{2}x''(\varepsilon_{k-1})\delta t^2)^2] - E[\tfrac{1}{2}x''(\varepsilon_{k-1})\delta t^2]^2 = \tfrac{1}{4}\delta t^4 Var(x''(\varepsilon_{k-1}))$$

$$Var(x''(\varepsilon_{k-1})\delta t) = E[(x''(\varepsilon_{t-1})\delta t)^2] - E[x''(\varepsilon_{k-1})\delta t]^2 = \delta t^2 Var(x''(\varepsilon_{k-1}))$$

$$Cov(\tfrac{1}{2}x''(\varepsilon_{k-1})\delta t^2, x''(\varepsilon_{k-1})\delta t) = E[(\tfrac{1}{2}x''(\varepsilon_{k-1})\delta t^2)(x''(\varepsilon_{k-1})\delta t)]$$
$$- E[\tfrac{1}{2}x''(\varepsilon_{k-1})\delta t^2]E[x''(\varepsilon_{k-1})\delta t] = \tfrac{1}{2}\delta t^3 Var(x''(\varepsilon_{k-1})) = Cov(x''(\varepsilon_{k-1})\delta t, \tfrac{1}{2}x''(\varepsilon_{k-1})\delta t^2)$$

Conseguimos así expresar la matriz Q_k en función de un parámetro, $Var(x''(\varepsilon_{k-1}))$, es decir, σ^2_{model}. Tenemos por lo tanto que:

$$Q_k = \begin{pmatrix} \frac{1}{4}\delta t^4 & \frac{1}{2}\delta t^3 \\ \frac{1}{2}\delta t^3 & \delta t^2 \end{pmatrix} \cdot \sigma^2_{model} \tag{3.9}$$

3.1.3. Filtro de Kalman tridimensional

De manera análoga al filtro de Kalman de dimensión dos, introduciremos una tercera variable de estado, $x''(t_k)$, para desarrollar el modelo de dimensión tres. Resulta así el siguiente sistema de ecuaciones en ausencia de ruido:

$$\begin{cases} x(t_k) = x(t_{k-1}) + x'(t_{k-1})\delta t + \dfrac{1}{2}x''(t_{k-1})\delta t^2 \\ x'(t_k) = x'(t_{k-1}) + x''(t_{k-1})\delta t \\ x''(t_k) = x''(t_{k-1}) \end{cases} \tag{3.10}$$

De la misma forma, tenemos que la matriz F_k para nuestro modelo tridimensional es la siguiente:

$$F = \begin{pmatrix} 1 & \delta t & \frac{1}{2}\delta t^2 \\ 0 & 1 & \delta t \\ 0 & 0 & 1 \end{pmatrix} \tag{3.11}$$

De nuevo, considerando el ruido del proceso, resulta el siguiente sistema:

$$\begin{cases} x(t_k) = x(t_{k-1}) + x'(t_{k-1})\delta t + \dfrac{1}{2}x''(t_{k-1})\delta t^2 + \dfrac{1}{6}x'''(\varepsilon_{k-1})\delta t^3 \\ x'(t_k) = x'(t_{k-1}) + x''(t_{k-1})\delta t + \dfrac{1}{2}x'''(\tilde{\varepsilon}_{k-1})\delta t^2 \\ x''(t_k) = x''(t_{k-1}) + x'''(\tilde{\tilde{\varepsilon}}_{k-1})\delta t \end{cases} \tag{3.12}$$

Asumiendo que $\varepsilon_{k-1} \simeq \tilde{\varepsilon}_{k-1} \simeq \tilde{\tilde{\varepsilon}}_{k-1}$, el error cometido en el proceso para el modelo de dimensión tres es $w_k = \begin{pmatrix} w_k^1 \\ w_k^2 \\ w_k^3 \end{pmatrix} = \begin{pmatrix} \frac{1}{6}\delta t^3 \\ \frac{1}{2}\delta t^2 \\ \delta t \end{pmatrix} x'''(\varepsilon_{k-1})$.

A su vez, tenemos que:

$$Q_k = Cov(w_k) = \begin{pmatrix} Var(w_k^1) & Cov(w_k^1, w_k^2) & Cov(w_k^1, w_k^3) \\ Cov(w_k^2, w_k^1) & Var(w_k^2) & Cov(w_k^2, w_k^3) \\ Cov(w_k^3, w_k^1) & Cov(w_k^3, w_k^2) & Var(w_k^3) \end{pmatrix}$$

Desarrollando los términos de la matriz Q_k, análogamente al caso bidimensional, conseguimos expresar la matriz Q_k en función de un parámetro, $Var(x'''(\varepsilon_{k-1}))$, es decir, σ^2_{model}. Tenemos por lo tanto que:

$$Q_k = \begin{pmatrix} \frac{1}{36}\delta t^6 & \frac{1}{12}\delta t^5 & \frac{1}{6}\delta t^4 \\ \frac{1}{12}\delta t^5 & \frac{1}{4}\delta t^4 & \frac{1}{2}\delta t^3 \\ \frac{1}{6}\delta t^4 & \frac{1}{2}\delta t^3 & \delta t^2 \end{pmatrix} \cdot \sigma^2_{model} \tag{3.13}$$

3.1.4. Filtro de Kalman n-dimensional

Una vez vistos los anteriores modelos para dimensiones uno, dos y tres, es fácil generalizar la dimensión del modelo que proponemos. De esta forma, un sistema de dimensión n quedará descrito por las variables $x(t)$, $x(t)'$, $x(t)''$, ..., $x(t)^{n-1)}$, cuyas aproximaciones por el desarrollo de Taylor nos permiten deducir que la matriz F_k para nuestro modelo de dimensión n es:

$$F = \begin{pmatrix} 1 & \delta t & \frac{1}{2!}\delta t^2 & \frac{1}{3!}\delta t^3 & \frac{1}{4!}\delta t^4 & \cdots & \frac{1}{(n-1)!}\delta t^{n-1} \\ 0 & 1 & \delta t & \frac{1}{2!}\delta t^2 & \frac{1}{3!}\delta t^3 & \cdots & \frac{1}{(n-2)!}\delta t^{n-2} \\ 0 & 0 & 1 & \delta t & \frac{1}{2!}\delta t^2 & \cdots & \frac{1}{(n-3)!}\delta t^{n-3} \\ 0 & 0 & 0 & 1 & \delta t & \cdots & \frac{1}{(n-4)!}\delta t^{n-4} \\ 0 & 0 & 0 & 0 & 1 & \cdots & \frac{1}{(n-5)!}\delta t^{n-5} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \tag{3.14}$$

Del mismo modo que en las secciones anteriores, tenemos que el ruido del proceso para el modelo de dimensión n viene dado por:

$$w_k = \begin{pmatrix} w_k^1 \\ w_k^2 \\ \vdots \\ w_k^n \end{pmatrix} = \begin{pmatrix} \frac{1}{n!}\delta t^n \\ \frac{1}{(n-1)!}\delta t^{n-1} \\ \vdots \\ \delta t \end{pmatrix} x^{n)}(\varepsilon_{k-1})$$

Sabemos que $Cov(w_k^i, w_k^j) = \sqrt{Var(w_k^i)Var(w_k^j)}$, por lo que si consideramos $\sigma_{model} = \sqrt{Var(x^{n)}(\varepsilon_{k-1}))}$ y el vector:

$$\tilde{w}_k = \begin{pmatrix} \tilde{w}_k^1 \\ \tilde{w}_k^2 \\ \vdots \\ \tilde{w}_k^n \end{pmatrix} = \begin{pmatrix} \frac{1}{n!}\delta t^n \\ \frac{1}{(n-1)!}\delta t^{n-1} \\ \vdots \\ \delta t \end{pmatrix} \cdot \sigma_{model}$$

tenemos que $Q_{k_{i,j}} = \tilde{w}_k^i \cdot \tilde{w}_k^j = Q_{k_{j,i}}$, y por lo tanto, $Q_k = \tilde{w}_k \cdot \tilde{w}_k^T$, es decir:

$$Q_k = \begin{pmatrix} \frac{1}{(n!)^2}\delta t^{2n} & \frac{1}{n!(n-1)!}\delta t^{2n-1} & \cdots & \frac{1}{n!}\delta t^{n+1} \\ \frac{1}{n!(n-1)!}\delta t^{2n-1} & \frac{1}{((n-1)!)^2}\delta t^{2n-2} & \cdots & \frac{1}{(n-1)!}\delta t^{n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n!}\delta t^{n+1} & \frac{1}{(n-1)!}\delta t^{n} & \cdots & \delta t^2 \end{pmatrix} \cdot \sigma^2_{model} \tag{3.15}$$

Antes de concluir esta subsección, destacar que existen numerosas propuestas de matriz Q_k elaboradas por diferentes autores. Por ejemplo, en [5], el autor propone, para el filtro de Kalman de dimensión dos, la siguiente matriz de covarianzas del ruido del proceso:

$$Q_k = \begin{pmatrix} \frac{1}{20}\delta t^5 & \frac{1}{8}\delta t^4 \\ \frac{1}{8}\delta t^4 & \frac{1}{3}\delta t^3 \end{pmatrix} \cdot \sigma^2_{model}$$

Otro ejemplo de matriz Q_k es la que propone el autor del artículo [2], el cual utiliza dos parámetros, a y b, distintos, para formular la siguiente matriz:

$$Q_k = \begin{pmatrix} a^2 & ab \\ ba & b^2 \end{pmatrix}$$

Una propiedad que comparte esta última matriz Q_k con la que hemos diseñado es que tienen determinante nulo, a diferencia de la primera de las dos matrices expuestas, que tiene determinante distinto de cero.
En la práctica, tras realizar varias pruebas (análogas a las que realizaremos en el cuarto capítulo), los resultados arrojados por la matriz Q_k con determinante no nulo son muy similares a los proporcionados por las dos matrices con determinante igual a cero. Por este motivo, y destacando que depende de un único parámetro, utilizaremos la matriz Q_k diseñada en lo que resta de trabajo.

3.2. Diseño de la matriz H

En esta sección diseñaremos la matriz del modelo de observación, H_k, de nuestro filtro de Kalman. Debemos de tener en cuenta que, independientemente de la dimensión de nuestro filtro, estamos aplicando éste a un mercado financiero en el cual la única medición del estado del sistema que podemos tomar es la del valor de la cotización, es decir, siguiendo la notación usada para detallar las dimensiones de las matrices que componen el filtro, $m = 1$.

En la etapa de predicción calculamos el estado del sistema, estando éste determinado por $\hat{x}_{k|k-1} = (x(t), x(t)', ..., x(t)^{n-1})^T$, dónde $x(t)$ representa la cotización del sistema predicha. Seguidamente, en la etapa de actualización, calculamos el residuo del pre-ajuste a través de la siguiente fórmula:

$$y_k = z_k - H_k\hat{x}_{k|k-1}$$

Supongamos, por un lado, que al realizar la medición del estado del sistema en el instante de tiempo k, z_k, en nuestro caso al obtener la cotización de un instrumento financiero en el mercado, tenemos que el par de divisas EURUSD cotiza a 1,2201, es decir, en el instante de tiempo k,

$z_k = 1,2201$.
Supongamos, por otro lado, que en la etapa de predicción hemos predicho que la cotización del EURUSD es de 1,2207. Resulta así que a la hora de calcular el residuo del pre-ajuste debemos de tomar la matriz H como un vector fila, o una matriz de dimensión $1 \times n$, con coeficientes nulos a excepción del primero, el cual será 1, para poder comparar la cotización medida con la calculada en la primera etapa:

$$y_k = 1,2201 - (1,0,\ldots,0) \begin{pmatrix} 1,2207 \\ x(t)' \\ \vdots \\ x(t)^{n-1)} \end{pmatrix}$$

Por lo tanto, teniendo en cuenta la dimensión general del filtro, concluimos esta sección definiendo la matriz H_k como:

$$H_k = (1,0,\ldots,0) \tag{3.16}$$

3.3. Diseño de la matriz R

En las ecuaciones del filtro de Kalman, la matriz R_k es la matriz de covarianzas del ruido de la observación v_k, el cual suponemos que proviene de una distribución normal de media cero. En nuestro modelo, independientemente de la dimensión de éste, la matriz R_k va a estar definida por $R_k = Var(v_k)$ y va a ser de dimensión 1×1, ya que, como z_k es de dimensión 1×1 ($m = 1$), v_k es de dimensión 1×1.
Análogamente a cuando construíamos la matriz Q_k, si denotamos al parámetro $\sigma^2_{measure}$ como la varianza del ruido de las mediciones, resulta que la matriz R_k del filtro queda expresada por:

$$R_k = \sigma^2_{measure} \tag{3.17}$$

3.4. Modelo del filtro de Kalman

Gracias a los resultados expuestos en los capítulos 2 y 3, hemos elaborado nuestro propio filtro de Kalman, el cual emplearemos para la regularización de las curvas de cotizaciones de varios instrumentos financieros. Este filtro viene dado por las siguientes ecuaciones:

Predicción:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} + B_k u_k$$
$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k$$

Actualización:

$$\begin{aligned} y_k &= z_k - H_k \hat{x}_{k|k-1} \\ S_k &= H_k P_{k|k-1} H_k^T + R_k \\ K_k &= P_{k|k-1} H_k^T S_k^{-1} \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k y_k \\ P_{k|k} &= (Id - K_k H_k) P_{k|k-1} \end{aligned}$$

donde:

$$F = \begin{pmatrix} 1 & \delta t & \cdots & \frac{1}{(n-1)!}\delta t^{n-1} \\ 0 & 1 & \cdots & \frac{1}{(n-2)!}\delta t^{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \quad B_k = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} \quad \begin{aligned} R_k &= \sigma^2_{measure} \\ H_k &= (1, 0, \ldots, 0) \end{aligned}$$

$$Q_k = \begin{pmatrix} \frac{1}{(n!)^2}\delta t^{2n} & \frac{1}{n!(n-1)!}\delta t^{2n-1} & \cdots & \frac{1}{n!}\delta t^{n+1} \\ \frac{1}{n!(n-1)!}\delta t^{2n-1} & \frac{1}{((n-1)!)^2}\delta t^{2n-2} & \cdots & \frac{1}{(n-1)!}\delta t^{n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n!}\delta t^{n+1} & \frac{1}{(n-1)!}\delta t^{n} & \cdots & \delta t^2 \end{pmatrix} \cdot \sigma^2_{model}$$

Destacar que, en la práctica, a la hora de empezar a iterar el filtro de Kalman, necesitamos inicializar tanto el vector de estado inicial del sistema, $\hat{x}_{0|0}$, como la matriz de covarianzas inicial de la diferencia entre el estado calculado y el medido, $P_{0|0}$.
Tomaremos el estado y la matriz de covarianzas inicial del sistema como:

$$\hat{x}_{0|0} = \begin{pmatrix} z_0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad P_{0|0} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

aunque podemos tomar cualquier otra matriz $P_{0|0}$ "razonable", por ejemplo, la matriz nula o la matriz con todos sus coeficientes unos, puesto que, como el autor de una de las principales fuentes de este trabajo comenta, [1], se dará una rápida convergencia en las covarianzas y, tras varias iteraciones, tendremos la misma matriz $P_{k|k}$, independientemente de la matriz $P_{0|0}$ escogida.

Antes de terminar con este capítulo, debemos de destacar la "no suavidad" de la curva de cotizaciones del instrumento financiero que estemos estudiando en el mercado. En este trabajo hemos supuesto que los cambios de dirección a corto plazo en la cotización del instrumento financiero a filtrar están causados por el ruido aleatorio que estas poseen, por lo que, si consiguiésemos eliminar todo el ruido del proceso, obtendríamos una curva "suave". En la práctica, emplearemos filtros de dimensiones bajas, por ejemplo, dimensión 1 o 2, puesto que, aunque deseamos obtener una curva "suave" como output del proceso de filtrado, no es muy realista pensar que dicha curva será derivable cuantas veces queramos.

Capítulo 4

Sensibilidad de los parámetros del Filtro de Kalman

En este capítulo estudiaremos la sensibilidad en los resultados proporcionados por el filtro de Kalman diseñado anteriormente en base a los parámetros de éste: la dimensión del filtro, la varianza del ruido del proceso, σ^2_{model}, y la varianza del ruido de la observación, $\sigma^2_{measure}$. Además, estudiaremos el comportamiento de nuestra matriz de covarianza Q_k en comparación a la matriz Q_k más sencilla posible, $Q_k := Id \cdot \sigma^2_{model}$. A la hora de estudiar la sensibilidad de cada parámetro, emplearemos el histórico de cotizaciones del EURUSD para estudiar la sensibilidad de la dimensión del filtro, y emplearemos las funciones seno, $f = sin(t)$, y sigmoidea, $f = \frac{1}{1+e^{-t}}$, perturbadas con un ruido aleatorio, para estudiar la sensibilidad del resto de parámetros.

4.1. Sensibilidad en la dimensión

Comenzaremos estudiando las diferencias entre los resultados arrojados por el filtro de Kalman en base a la dimensión de éste. Como ya comentamos en el capítulo anterior, no tiene sentido tomar dimensiones elevadas, puesto que no es realista asumir que la curva sin ruido de las cotizaciones es demasiado suave. Por este motivo, estudiaremos los resultados arrojados por el filtro de Kalman de dimensión 1, 2, 3 y 4.
En la imagen 4.1 podemos ver representado el filtrado de la cotización del par de divisas EURUSD en base a la dimensión del filtro empleado fijados $\sigma_{measure}$ y σ_{model}. Hemos tomado $\sigma_{model} = 0,0002$ y $\sigma_{measure} = 0,0005$.

Gracias a dicho gráfico podemos afirmar que, fijados $\sigma_{measure}$ y σ_{model}, cuanto mayor dimensión posee el filtro de Kalman, mayor "overfitting" o sobre ajuste (producción de un análisis que se corresponde demasiado con un conjunto particular de datos y, por lo tanto, puede fallar en ajustar datos adicionales o predecir observaciones futuras) sufre nuestra señal filtrada, es decir, el output se aproxima demasiado al input. Por el contrario, cuanto menor sea la dimensión del filtro, más suave va a ser la señal filtrada resultante y menos se va a parecer a la señal con ruido.

Como conclusión de esta primera sección obtenemos que, en la práctica, a la hora de filtrar las cotizaciones de un activo en el mercado financiero, no debemos emplear un filtro de dimensión elevada. Lo idóneo sería emplear un filtro de Kalman unidimensional o bidimensional, para evitar así sufrir "overfitting" en los resultados.

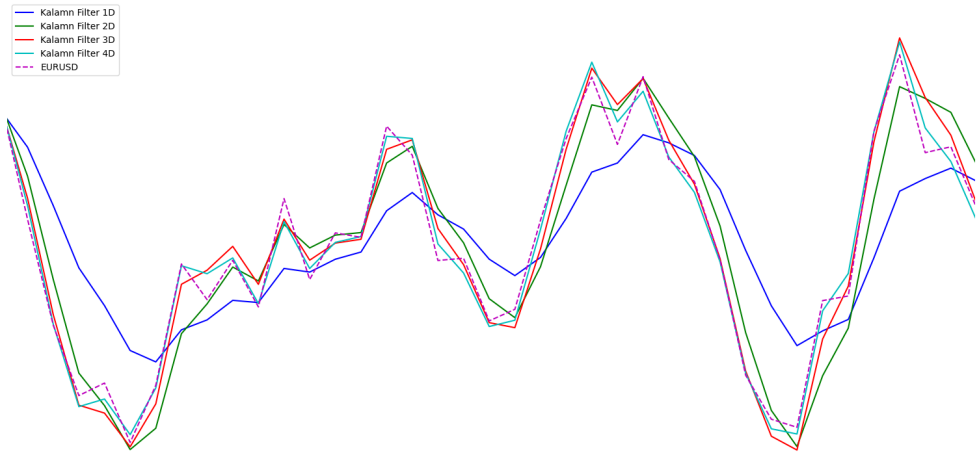


Figura 4.1: Sensibilidad en la dimensión, EURUSD

4.2. Sensibilidad en σ_{model}

Seguiremos el estudio de la sensibilidad de los parámetros de nuestro modelo profundizando en cómo afecta el parámetro σ_{model} a la calidad de los resultados arrojados por el filtro de Kalman. Análogamente al apartado anterior, fijaremos el resto de parámetros que necesita el filtro, es decir, fijaremos la dimensión del modelo y $\sigma_{measure}$, representando así el vector de estados obtenidos tras el filtrado de la señal para varios valores de σ_{model}. Hemos tomado los parámetros $n = 2$ y $\sigma_{measure} = 0,03$.

Gracias a las gráficas 4.2 y 4.3 podemos destacar, fijados la dimensión del filtro de Kalman y $\sigma_{measure}$, que cuanto mayor sea σ_{model}, más "overfitting" sufrimos. Por el contrario, cuanto menor sea σ_{model}, más suave es la señal filtrada respecto de la original.

Antes de realizar este estudio era de esperar obtener estos resultados, puesto que, cuando explicábamos los fundamentos matemáticos subyacentes en las ecuaciones del filtro de Kalman, comentábamos que, al calcular la media ponderada entre el estado del sistema predicho y el medido, debíamos darle un peso a cada uno de los estados en función de la confianza que tuviésemos en cada uno de ellos. Este peso era la ganancia óptima de Kalman, en la cual influyen los parámetros σ_{model} y $\sigma_{measure}$ a través de las matrices Q_k y R_k respectivamente. El parámetro σ_{model} es la varianza del error del proceso, lo cual, groso modo, es la confianza que depositamos en el filtro, luego, cuanto menor sea este parámetro, mayor será la confianza que tenemos en el filtro de Kalman. Por este motivo, vemos que cuanto más pequeño es σ_{model} más suave y alejada es la curva obtenida de la registrada, y cuanto más grande es, más se aproxima el resultado del filtrado a la señal original.

Como conclusión de esta sección obtenemos que, en la práctica, a la hora de filtrar las cotizaciones de un activo en el mercado financiero, debemos emplear un σ_{model} relativamente pequeño para evitar en la medida de lo posible sufrir de sobre ajuste en los resultados, pero sin que el mismo sea demasiado pequeño, puesto que las señales registrada y filtrada estarían demasiado alejadas.

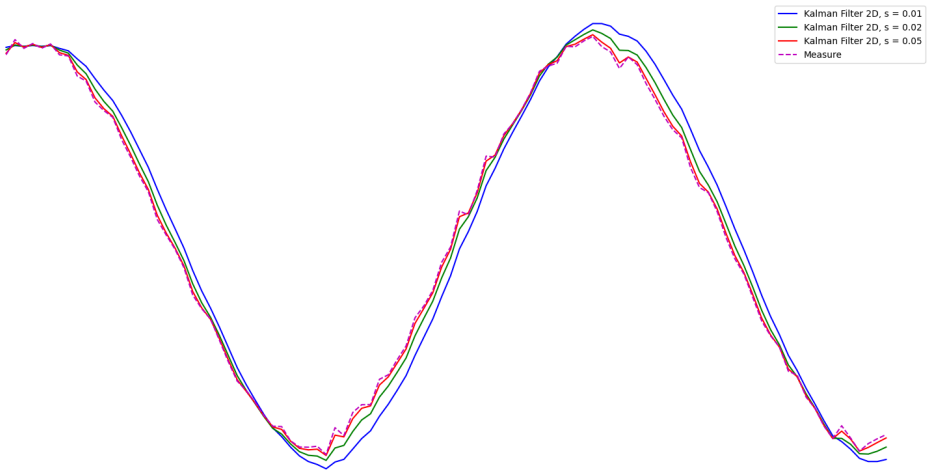


Figura 4.2: Sensibilidad en la σ_{model}, $f = sin(t)$

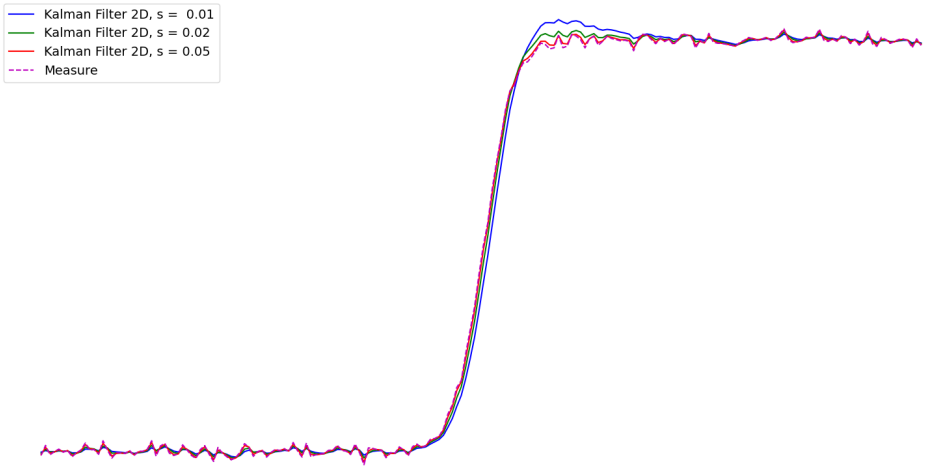


Figura 4.3: Sensibilidad en la σ_{model}, $f = \frac{1}{1+e^{-t}}$

4.3. Sensibilidad en $\sigma_{measure}$

Continuaremos con el estudio de la sensibilidad de los parámetros del filtro observando el comportamiento del filtro en base al parámetro $\sigma_{measure}$. De nuevo, fijaremos el resto de parámetros del filtro, es decir, fijaremos la dimensión de éste y σ_{model}, y representaremos el vector de estados obtenidos tras el filtrado de la señal para varios valores de $\sigma_{measure}$. Hemos tomado los parámetros $n = 2$ y $\sigma_{model} = 0,03$.

Gracias a las gráficas 4.4 y 4.5, fijados la dimensión del filtro de Kalman y σ_{model}, podemos destacar que cuanto mayor sea $\sigma_{measure}$, menos se parece la señal filtrada a la original. Por el contrario, cuanto menor sea $\sigma_{measure}$, más se parecen ambas.

De nuevo, al igual que sucedía con el parámetro σ_{model}, los resultados obtenidos coinciden con los esperados. El parámetro $\sigma_{measure}$ representa la confianza que tenemos en la medida registrada, la cual, cuanto menor sea el valor de $\sigma_{measure}$, mayor será y viceversa. Si otorgamos al filtro de Kalman un $\sigma_{measure}$ relativamente grande, confiaremos más en el filtro que en la medida, por lo que en la etapa de actualización, al hacer la media ponderada entre el estado predicho y el registrado, estaremos dando más peso al primero que al segundo. Si por el contrario, introducimos un $\sigma_{measure}$ relativamente pequeño, la ponderación será más próxima al estado medido que al calculado.

Como conclusión de esta sección obtenemos que, en la práctica, a la hora de filtrar las cotizaciones de un activo en el mercado financiero, debemos de emplear un $\sigma_{measure}$ relativamente grande para evitar en la medida de lo posible sufrir de "overfitting" en los resultados.

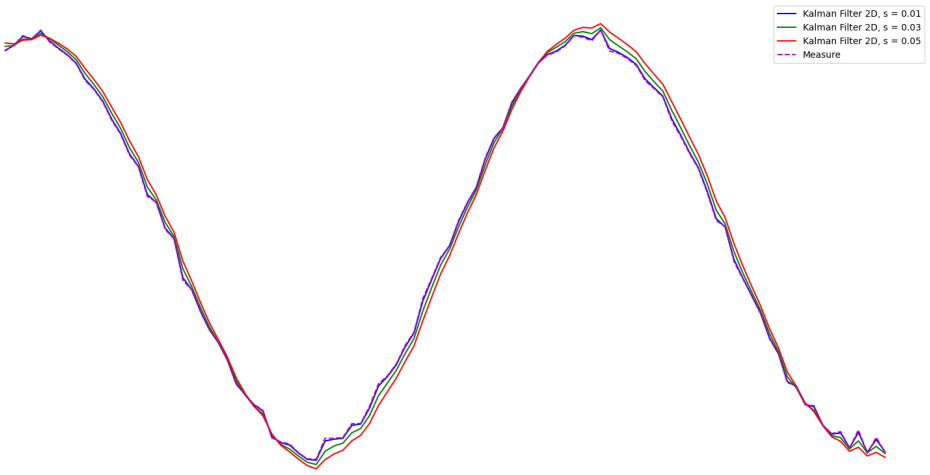


Figura 4.4: Sensibilidad en la $\sigma_{measure}$, $f = sin(t)$

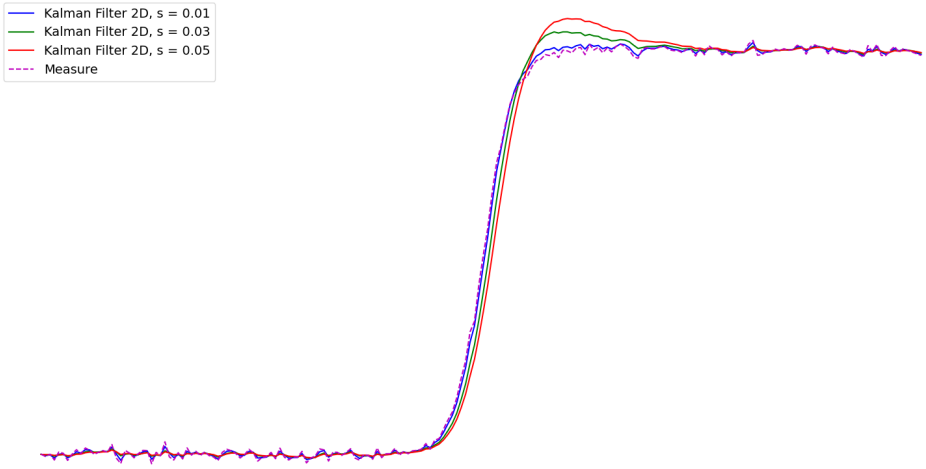


Figura 4.5: Sensibilidad en la $\sigma_{measure}$, $f = \frac{1}{1+e^{-t}}$

4.4. Sensibilidad en la matriz Q

Antes de concluir cómo afectan los parámetros del modelo a la calidad de las soluciones, queremos hacer un breve estudio sobre las diferencias existentes entre los vectores de filtrado obtenidos por el uso de la matriz Q_k trivial, es decir, $Q_k = Id \cdot \sigma^2_{model}$, y los obtenidos por el uso de la matriz Q_k diseñada en el capítulo 3.

Como ya hemos visto en las anteriores secciones, en nuestro caso, en el cual filtraremos las cotizaciones de un activo financiero, carece de sentido tomar dimensiones relativamente grandes para el filtro de Kalman, viéndose restringida nuestra elección a las dimensiones 1 y 2. En el caso unidimensional, las matrices Q_k son esencialmente iguales, por lo que nos limitaremos a comparar los resultados obtenidos en base a la matriz Q_k introducida en el filtro de Kalman bidimensional. Fijados los parámetros del filtro, $n = 2$, $\sigma_{measure} = 0,01$ y $\sigma_{model} = 0,05$, y pasando como input las funciones seno y sigmoidea perturbadas que venimos utilizando en este capítulo, observamos, gracias a las gráficas 4.6 y 4.7, que el vector de filtrado obtenido con el filtro de Kalman que utiliza la matriz $Q_k = Id \cdot \sigma^2_{model}$ sufre de mayor sobre ajuste que el vector obtenido con el filtro de Kalman que emplea la matriz Q_k diseñada en el tercer capítulo.

En esta sección podemos concluir que, en el caso unidimensional, nos es indiferente que matriz Q_k elegir, pero en el caso bidimensional, para evitar en la medida de los posible el "overfitting", en nuestro modelo, es más conveniente utilizar la matriz Q_k diseñada en el capítulo 3.

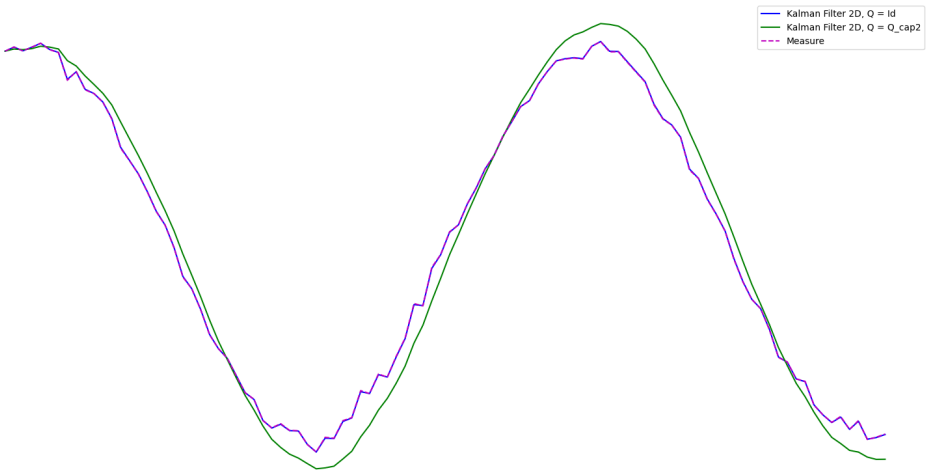


Figura 4.6: Sensibilidad en la matriz Q_k, $f = sin(t)$

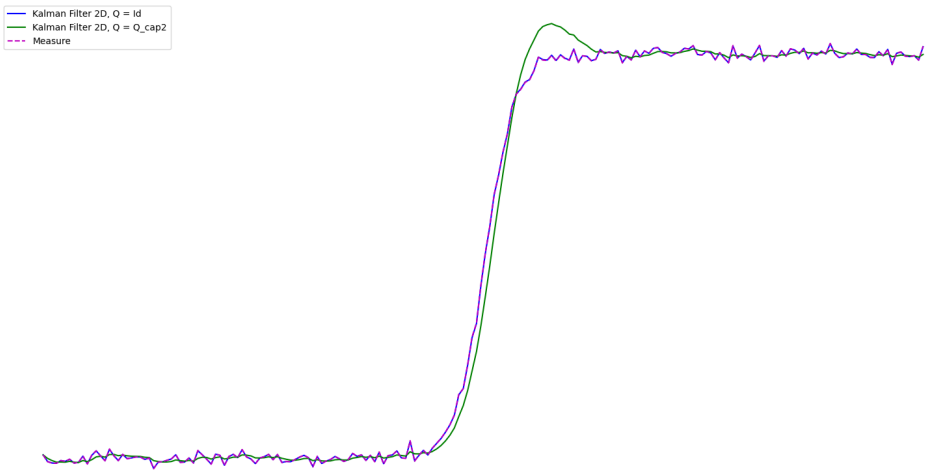


Figura 4.7: Sensibilidad en la matriz Q_k, $f = \frac{1}{1+e^{-t}}$

4.5. Conclusiones

En esta última sección del cuarto capítulo recogemos las conclusiones obtenidas en las anteriores secciones con el fin de dar, en base a nuestro objetivo (filtrar las cotizaciones de una activo financiero), una recomendación de ajuste de estos parámetros.
En primer lugar, estudiando la sensibilidad de los resultados en base a la dimensión del filtro, concluimos que, cuanto mayor dimensión posee el filtro, más se parece la señal filtrada a la original, es decir, mayor sobre ajuste experimentamos, por lo que recomendamos limitar la dimensión del filtro a dimensión 2.
En segundo lugar, mediante el estudio de la sensibilidad de los parámetros σ_{model} y $\sigma_{measure}$ observamos que, cuanto mayor sea el parámetro σ_{model} en comparación con $\sigma_{measure}$ o menor sea $\sigma_{measure}$ en comparación con σ_{model}, mayor "overfitting" experimentaremos, por lo que nuestra recomendación es ajustar estos parámetros siguiendo una relación entre ambos cercana a la unidad, como por ejemplo, $\sigma_{model} = 0{,}0001$ y $\sigma_{measure} = 0{,}0002$, o al revés, pero que no existan grandes diferencias entre ellos, puesto que, si σ_{model} es muy superior a $\sigma_{measure}$, por ejemplo, 50 veces $\sigma_{measure}$, el vector de filtrado resultante se alejará demasiado de las señales registradas, y, si por el contrario, $\sigma_{measure}$ es muy superior a σ_{model}, el vector resultante será prácticamente igual al vector de mediciones.
Por último, observamos cómo la matriz Q_k diseñada en el capítulo 3 suaviza mejor el vector de mediciones que la matriz Q_k trivial, puesto que esta última se acerca bastante a la señal medida y sobre reacciona a los cambios de dirección. Es por este motivo que recomendamos el uso de la matriz Q_k diseñada en lugar de la identidad.

En el siguiente capítulo emplearemos el filtro diseñado con anterioridad, en base a estas recomendaciones de ajuste de los parámetros del filtro, y emplearemos éste para suavizar las cotizaciones de un activo financiero en el mercado y ver el rendimiento obtenido de operar en tiempo real aplicando estrategias basadas en el filtro de Kalman.

Capítulo 5

Aplicación del filtro de Kalman en FOREX

El mercado de divisas, también conocido como FOREX (abreviatura del término inglés Foreign Exchange), FX o Currency Market, es un mercado mundial y descentralizado en el que se negocian divisas.

Este mercado nació con el objetivo de facilitar el flujo monetario que se deriva del comercio internacional. Es, por gran margen, el mercado financiero más grande del mundo, llegando a mover un volumen diario de transacciones de alrededor de 6600 millones de dólares estadounidenses (USD), más que todos los demás mercados bursátiles del planeta combinados. Ha crecido tanto que, en la actualidad, el total de operaciones en moneda extranjera que se debe a operaciones internacionales de bienes y servicios representan un porcentaje casi residual, debiéndose la mayoría de las mismas a compraventa de activos financieros. En consecuencia, este mercado es bastante independiente de las operaciones comerciales reales, y las variaciones entre el precio de dos monedas no puede explicarse de forma exclusiva por las variaciones de los flujos comerciales.
El mercado de divisas es un mercado mundial que, aunque cuenta con acceso las 24 horas, en la práctica se ve limitado por el paréntesis de las operaciones en el fin de semana. Aún en esos periodos de interrupción, los operadores pueden colocar posiciones de compra o de venta que se verán dinamizadas cuando el mercado comience a fluctuar. Los principales centros de negociación son las bolsas de Londres, Nueva York y Tokio. Primero abren los mercados asiáticos, posteriormente los europeos y finalmente abren los mercados americanos. En el horario peninsular, el mercado abre el domingo a las 23:00 y cierra el viernes a las 21:00.

Uno de los mayores logros de la era digital es la virtualización del dinero. Si el dinero impreso permitió transferir crédito entre las personas por medio de un simple billete, las tecnologías de la información o Internet hicieron que este proceso fuera tan sencillo como presionar un botón o tocar una pantalla. En los noventa, esta tecnología se reconoció como una gran oportunidad y se crearon empresas que permitieron el acceso al mercado de divisas y las cuentas con apalancamiento (utilizar deuda para aumentar la cantidad de dinero que podemos destinar a una inversión, es la relación entre capital propio y el realmente utilizado en una operación financiera). Estas empresas llegaron a ser conocidas como brókers de FOREX, y es gracias a ellas que actualmente cualquier persona puede operar en este mercado.

La mayoría de brókers online poseen una interfaz de programación de aplicaciones (API), la

cual es un conjunto de procesos, funciones y métodos que brinda una determinada biblioteca de programación a modo de capa de abstracción para ser empleada por otro programa informático externo. Operar mediante API es una manera sencilla de conseguir datos de mercado en tiempo real, precios históricos y ejecutar operaciones, sin tener que rastrear de forma manual los mercados en busca de datos y precios. En lugar de esto, se recibe directamente la información del servidor, lo que asegura una gran eficiencia y rapidez. Estos avances tecnológicos han revolucionado el mundo del trading, en el cual el trading algorítmico o cuantitativo (modalidad de operación en mercados financieros que se caracteriza por el uso de algoritmos, reglas y procedimientos automatizados para ejecutar operaciones de compra o venta de instrumentos financieros) ha ganado mucho terreno al trading tradicional, consiguiendo más de un 80 % de las operaciones que se cierran en mercados como el americano.

En este trabajo, influenciados por el autor del artículo [3], hemos empleado la plataforma de trading MetaTrader5 y el lenguaje de programación Python para operar en FOREX a través de la API de MetaTrader5, tomando decisiones de compra y venta sobre los distintos pares de divisas que conforman este mercado gracias a la estrategia basada en el filtro de Kalman previamente diseñada.

5.1. Las bandas de Bollinger

El análisis técnico consiste en el estudio de los mercados financieros basado en datos, patrones de precios y tendencias de las cotizaciones. A diferencia del análisis fundamental, que se centra más en el estudio del contexto económico, político y social, el análisis técnico es puramente matemático y algorítmico, basándose siempre en patrones y datos pasados.

Los indicadores técnicos son fórmulas matemáticas y estadísticas que se aplican a las series de precios y volúmenes para ayudar a tomar decisiones en el ámbito del análisis técnico. Existen numerosos tipos de indicadores técnicos, como pueden ser los indicadores tendenciales, contra tendenciales, de volumen, de volatilidad, de fuerza relativa... Nosotros nos centraremos en los dos primeros, indicadores técnicos tendenciales y contra tendenciales, los cuales arrojan señales de compra/venta en función de los movimientos, o tendencias, de los precios.

El más simple y básico es una media móvil, que consiste en el promedio de precios en un periodo determinado. Por ejemplo, si calculamos la media móvil de los últimos 20 días en el par de divisas EURGBP (euro - libra esterlina), sumaríamos el precio del cierre de las cotizaciones de éste en los últimos 20 días y dividiríamos el resultado entre 20. El día de mañana, repetiremos el cálculo del promedio añadiendo el precio de cierre de mañana y eliminando el precio de hace 20 días. En base a este indicador, se desarrollan numerosos indicadores técnicos como el MACD (Moving Average Convergence Divergence), el Canal de Keltner o las bandas de Bollinger.

En esta sección nos centraremos en éste último y estudiaremos cómo el filtro de Kalman puede solventar alguno de los principales inconvenientes que posee este indicador.

En la década de los años 80, John Bollinger introdujo el indicador técnico de las bandas de Bollinger, el cual, desde entonces, se ha convertido en uno de los indicadores técnicos más empleados dentro del análisis técnico. El indicador está formado por dos curvas que envuelven el gráfico de precios a estudiar. Se calcula a partir de una media móvil de n periodos sobre el precio de cierre, a

la que envuelven dos bandas que se obtienen de añadir y sustraer al valor de la media k desviaciones estándar obtenidas sobre los últimos m periodos (normalmente se suele tomar $n = m$). A los enteros n y m se les denomina ventana de muestreo y a k factor de las bandas.

De acuerdo con el análisis técnico, el que los precios sobrepasen las bandas indica que el mercado está sobre comprado si lo hacen por arriba o sobre vendido si lo hacen por abajo. Atendiendo a este razonamiento, cuando el precio se encuentre por debajo de la banda inferior debemos introducir una orden de compra, o posicionarnos en largo (posición en la que el inversor se beneficia si el valor del activo sube). Si por el contrario, el precio sobrepasa la banda superior debemos introducir una orden de venta, o posicionarnos en corto (posición en la que el inversor se beneficia si el valor del activo cae).

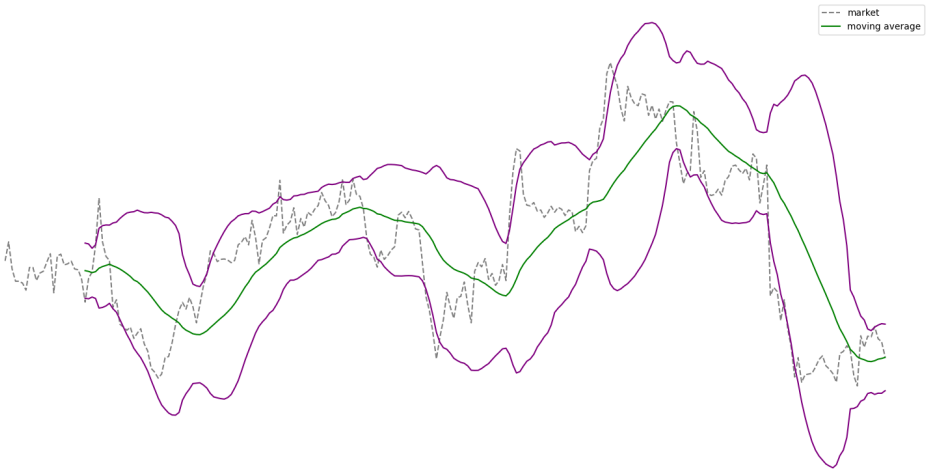


Figura 5.1: Bandas de Bollinger Moving Average, EURGBP

Como podemos observar en la imagen anterior, uno de los mayores inconvenientes que poseen los promedios móviles es que son indicadores retrasados. Podemos ver cómo la línea de promedio, representada en color verde, sigue el movimiento del mercado e indica el surgimiento de una nueva tendencia o su reversión después del inicio del movimiento de la tendencia, por lo que el indicador de las bandas de Bollinger, representado en color morado, es, fundamentalmente, un indicador retrasado. Este hecho nos puede originar un gran número de señales de compra/venta falsas y es por eso que en este trabajo nos centraremos en tratar de solventar el retraso que experimenta este indicador.

En un marco ideal en el que podríamos disponer de datos futuros, podríamos sustituir la media móvil en la que se basan las bandas de Bollinger por una media móvil centrada. Una media o promedio móvil centrado es un promedio móvil en el que los promedios calculados se colocan en el centro del rango tomado para el cálculo de éste en lugar de al final del mismo. Supongamos que la ventana del promedio móvil es de 5 periodos. En ese caso, el promedio móvil centrado coloca el primer valor de promedio móvil numérico en el período 3, el siguiente en el período 4, y así sucesivamente. Esto se hace para posicionar los valores de la media móvil en sus posiciones centrales en el tiempo.

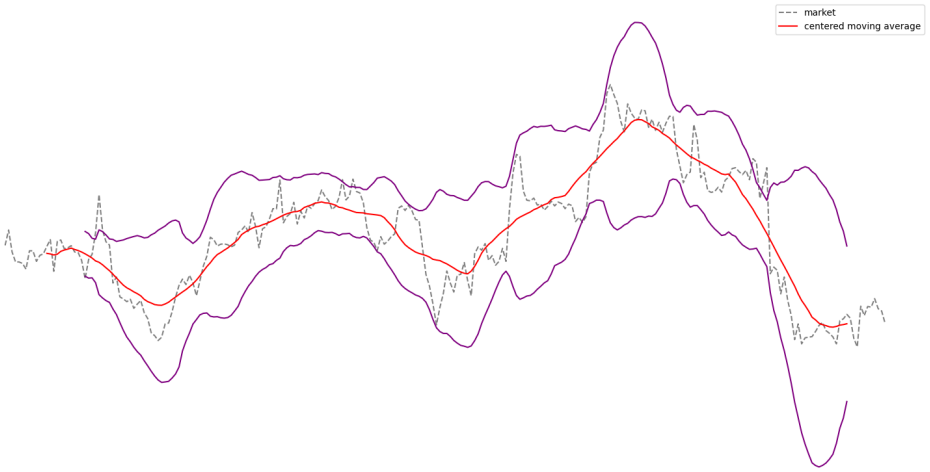


Figura 5.2: Bandas de Bollinger Centered Moving Average, EURGBP

En la imagen anterior podemos ver cómo la media móvil centrada, representada en color rojo, filtra a la perfección el históricos de cotizaciones del par de divisas EURGBP, solventando así el retraso en las señales de entrada al mercado que nos otorgaba el indicador de las bandas de Bollinger basado en una media móvil sin centrar.

Como ya hemos comentado, las medias móviles centradas solo se pueden calcular si se dispone de datos futuros, por lo operar en tiempo real con estas medias resulta una tarea imposible. Es por ello que necesitamos una herramienta que suavice la curva de cotizaciones de manera parecida a la media móvil centrada, para evitar así el retraso en los movimientos, pero sin tomar datos a futuro.

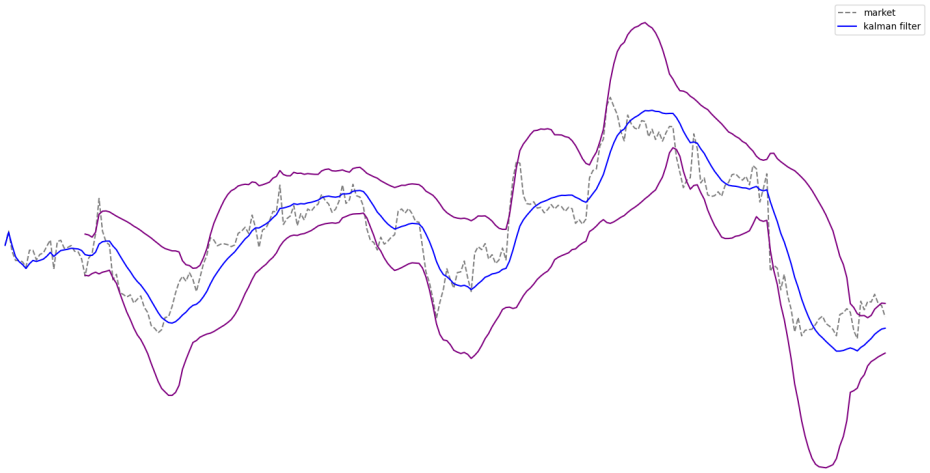


Figura 5.3: Bandas de Bollinger Kalman filter, EURGBP

En la imagen 5.3 se puede apreciar cómo el filtro de Kalman que hemos diseñado en el capítulo 3 cumple el objetivo que perseguimos, suavizar la curva de cotizaciones sin experimentar tanto retraso en los movimientos. Siguiendo las recomendaciones aportadas en el capítulo 4, para suavizar el vector de cotizaciones anterior, hemos tomado los parámetros $\sigma_{model} = 0,001$ y $\sigma_{measure} = 0,001$ para el filtro de Kalman bidimensional.

El problema del filtro de Kalman es que no existen unos parámetros universales que filtren bien todos los instrumentos financieros existentes, por lo que, en cada uno de ellos, debemos ir ajustando los parámetros de manera que obtengamos un filtrado óptimo, buscando que no esté muy alejado de los datos de mercado, pero manteniendo cierto grado de suavidad. En la siguiente sección diseñaremos una estrategia que nos permita operar de forma automatizada en el mercado FOREX gracias al indicador de las bandas de Bollinger basado en el filtro de Kalman, escogiendo de manera automática los parámetros óptimos.

5.2. Diseño de una estrategia

El objetivo de esta sección es diseñar una estrategia rentable para operar en tiempo real dentro del mercado FOREX. Dicha estrategia nos deberá dar, al cierre de cada periodo, una señal de actuación, ya sea de compra, venta o abstención. Antes de comenzar con el desarrollo de la estrategia, cabe destacar la amplitud del rango de posibles timeframes, o duraciones de periodo, que podemos elegir a la hora de operar a través de la plataforma MetaTrader5, los cuales van desde 1 minuto hasta 1 mes, siendo los más usados los periodos de 5 minutos, 15 minutos, 30 minutos, 1 hora o 1 día. Para simplificar las explicaciones, en lo que resta de capítulo supondremos que vamos a operar con la estrategia de manera horaria, es decir, al cierre de cada hora la estrategia evaluará la situación del mercado y nos devolverá una señal de actuación, aunque se podrá escoger la duración de periodo que se desee una vez esté implementada dicha estrategia.

Nuestra estrategia se basa en maximizar, a través de una correcta elección de los parámetros, el poder predictivo del indicador de las bandas de Bollinger basado en el filtro de Kalman, y usar las señales de compra/venta arrojadas por este indicador (comprar si el precio cae por debajo de la banda inferior, vender si el precio sobrepasa la banda superior, y abstenerse en el resto de situaciones) para operar en el mercado.

En un entorno ideal, usando como herramienta de filtrado las medias móviles centradas, dotamos al indicador de las bandas de Bollinger de una enorme capacidad de predicción, por lo que nuestro objetivo será, en la medida de lo posible, tratar de obtener, mediante los parámetros de éste, un filtro de Kalman lo más parecido posible a una media móvil centrada de n periodos.

Para lograr este objetivo, en cada iteración, en este caso, cada hora, dotaremos a la estrategia del histórico de cierres horarios de las últimas h horas, por ejemplo, de las últimas 336 horas (2 semanas), para las cuales calcularemos su media móvil centrada de m periodos. Una vez calculada la media móvil centrada, fijada la dimensión del filtro, $n = 2$, y el parámetro $\sigma_{measure} = 10^{-3}$, obtendremos el parámetro σ_{model}, perteneciente al intervalo $(10^{-5}, \sigma_{measure})$, que minimice el error cuadrático de la diferencia entre la media móvil centrada y el filtro de Kalman resultante de utilizar

esos parámetros, es decir, resolvemos:

$$\begin{cases} minimizar & \|CMA - KF(\sigma_{model})\|_2^2 \\ sujeto\ a: & \sigma_{model} \in (10^{-5}, \sigma_{measure}) \end{cases} \tag{5.1}$$

donde CMA denota la media móvil centrada de m periodos y $KF(\sigma_{model})$ el filtro de Kalman con parámetros $n = 2$, $\sigma_{measure} = 10^{-3}$ y σ_{model}.

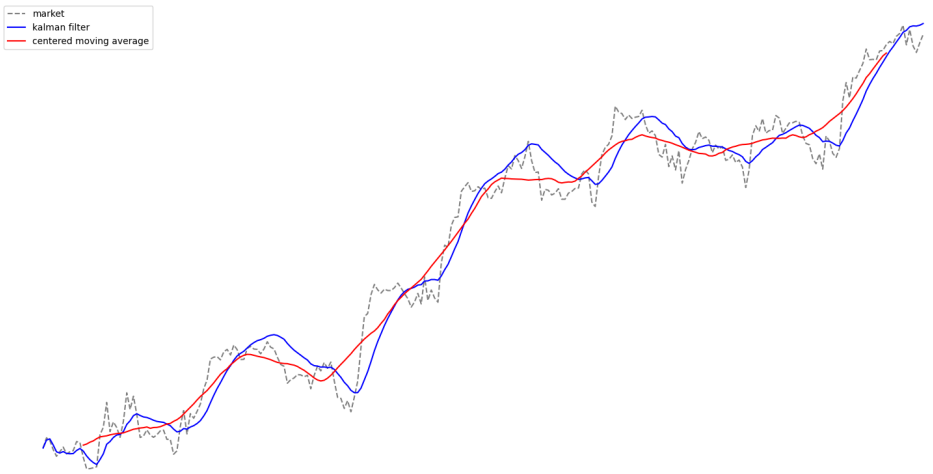


Figura 5.4: Optimización filtro de Kalman, EURUSD

En la figura 5.4 vemos representados, en rojo, la media móvil centrada de 24 periodos (en este caso, 24 horas), y, en azul, el filtro de Kalman con menor error cuadrático posible respecto a la media móvil centrada, sobre la cotización del par EURUSD, en un marco temporal de 2 semanas. En esta imagen podemos observar cómo, a pesar de ser el filtro de Kalman con menor error cuadrático sobre la media móvil centrada, hay demasiados cambios de dirección en comparación con ésta. Para tratar de solventar el problema de "no suavidad" en el filtro resultante del proceso de optimización, emplearemos la regularización de Tikhonov, la cual consiste, con el objetivo de dar preferencia a una solución particular con propiedades deseables, incluir un término de regularización en el problema de minimización. [9]

En nuestro caso, de entre todas las soluciones posibles, buscamos la que cometa un menor error cuadrático respetando cierto grado de suavidad. Este grado de suavidad lo podemos traducir en que haya la menor distancia posible entre posiciones consecutivas del vector de filtrado, es decir, que $\sum_{i=1}^{w-1} \|KF_{i+1} - KF_i\|_2^2$ sea lo menor posible, siendo w la dimensión del vector de filtrado. Considerando este último valor como el término de regularización que estamos buscando, resulta que en cada evaluación del mercado la estrategia resolverá el siguiente problema:

$$\begin{cases} minimizar & \|CMA - KF(\sigma_{model})\|_2^2 + \delta \sum_{i=1}^{w-1} \|KF_{i+1} - KF_i\|_2^2 \\ sujeto\ a: & \sigma_{model} \in (10^{-5}, \sigma_{measure}) \end{cases} \tag{5.2}$$

siendo δ el peso que deseamos darle al término de regularización en el proceso de optimización.

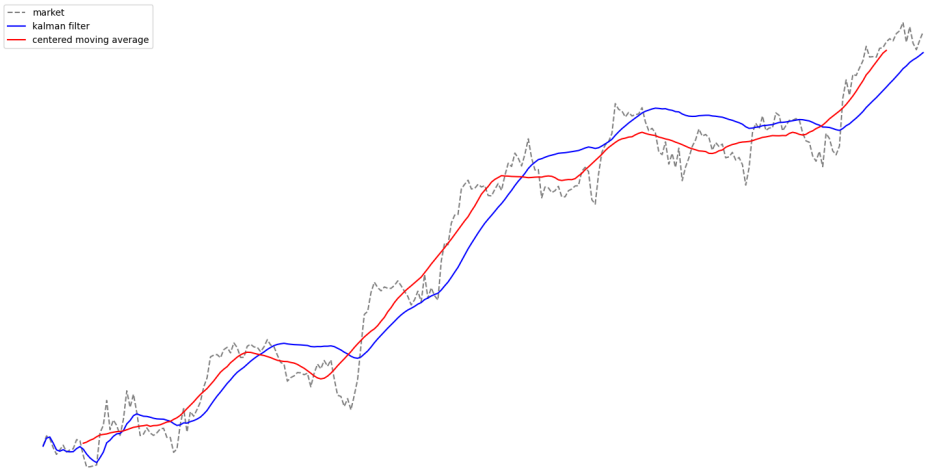


Figura 5.5: Optimización con regularización filtro de Kalman, EURUSD

En la anterior imagen podemos apreciar cómo incluyendo el término de regularización (tomando $\delta = 10$) hemos conseguido obtener una solución equilibrada entre ser parecida, en términos de error cuadrático, a la media móvil centrada y poseer cierto grado de suavidad.

En resumen, nuestra estrategia actual para operar en FOREX con el indicador de las bandas de Bollinger consiste en, de manera horaria, obtener los últimos h precios de cierre horario, por ejemplo, los precios de cierre horarios de las últimas 336 horas, para resolver el problema de optimización (5.2) y así obtener el filtro de Kalman deseado, gracias al cual calculamos el indicador de las Bandas de Bollinger. Observando si el último precio de cierre se sitúa dentro o fuera de las bandas de Bollinger, la estrategia emitirá una señal de acción sobre el mercado.

Una última puntualización que haremos sobre la estrategia es que, con el objetivo de mejorar la eficiencia computacional de ésta y evitar el sobre entrenamiento de la misma, no obtendremos el parámetro σ_{model} óptimo en cada iteración, sino que obtendremos dicho parámetro cada x iteraciones, por ejemplo, cada 24. Esto es debido a que los parámetros σ_{model} obtenidos al resolver el problema de optimización (5.2) en iteraciones consecutivas apenas sufren variaciones, por lo que deberán pasar varias iteraciones hasta que haya una diferencia significativa entre ellos.

Resultaría así que nuestra estrategia, en cada iteración, obtendría una señal de compra/venta basada en el indicador de las bandas de Bollinger obtenido a partir del filtro de Kalman bidimensional de parámetros $\sigma_{measure} = 10^{-3}$ y σ_{model}, resultante del último proceso de optimización, el cual se realiza con una frecuencia dada.

5.3. BackTesting

Un backtesting es una serie de pruebas sobre una estrategia en el pasado para evaluar cuál fue su rendimiento en un rango de datos histórico y extraer conclusiones de los datos obtenidos.

Un backtesting se puede realizar en cualquier mercado: divisas, acciones, criptomonedas,

etcétera, pero siempre necesitaremos disponer de datos históricos de calidad. Hay que destacar que el backtest por sí solo no es suficiente para establecer si una estrategia es robusta o no, ya que el hecho de que una estrategia arroje buenos resultados en el pasado no garantiza que los vaya a dar en el futuro, pero sí nos permite estudiar el comportamiento y extraer conclusiones de ésta.

El backtesting se puede realizar de dos formas: manual y automática. La primera de ellas consiste en probar manualmente una estrategia contra datos históricos ejecutándola paso a paso y tomando las decisiones de trading que te indica la estrategia en cada momento. La segunda de ellas consiste en realizar todos estos pasos de forma automatizada a través de un software. En esta sección nos centraremos en desarrollar un script en Python que nos permita realizar un backtest a la estrategia de las bandas de Bollinger que hemos diseñado en varios pares de divisas del mercado FOREX.

Para realizar dicho backtest, en primer lugar nos descargaremos el histórico de precios del par en el que vayamos a testear la estrategia y lo almacenaremos en una matriz.

time	open	high	low	close	tick_volume	spread	real_volume
2021-06-10 22:00:00+00:00	1.21716	1.21766	1.21700	1.21704	986	0	0
2021-06-10 23:00:00+00:00	1.21703	1.21777	1.21700	1.21761	659	0	0
2021-06-11 00:00:00+00:00	1.21639	1.21764	1.21639	1.21740	311	3	0
2021-06-11 01:00:00+00:00	1.21740	1.21750	1.21706	1.21728	317	0	0
2021-06-11 02:00:00+00:00	1.21728	1.21750	1.21719	1.21728	294	0	0
2021-06-11 03:00:00+00:00	1.21728	1.21788	1.21718	1.21782	1027	0	0
2021-06-11 04:00:00+00:00	1.21782	1.21887	1.21760	1.21872	1569	0	0
2021-06-11 05:00:00+00:00	1.21871	1.21902	1.21831	1.21900	1159	0	0
2021-06-11 06:00:00+00:00	1.21901	1.21914	1.21868	1.21872	745	0	0
2021-06-11 07:00:00+00:00	1.21873	1.21904	1.21868	1.21872	480	0	0
2021-06-11 08:00:00+00:00	1.21872	1.21902	1.21858	1.21879	741	0	0
2021-06-11 09:00:00+00:00	1.21881	1.21930	1.21814	1.21834	1984	0	0
2021-06-11 10:00:00+00:00	1.21834	1.21918	1.21748	1.21828	2874	0	0
2021-06-11 11:00:00+00:00	1.21828	1.21836	1.21640	1.21658	3101	0	0
2021-06-11 12:00:00+00:00	1.21657	1.21672	1.21527	1.21541	2509	0	0
2021-06-11 13:00:00+00:00	1.21542	1.21609	1.21490	1.21512	2395	0	0
2021-06-11 14:00:00+00:00	1.21512	1.21518	1.21298	1.21324	2944	0	0
2021-06-11 15:00:00+00:00	1.21324	1.21378	1.21225	1.21288	3068	0	0
2021-06-11 16:00:00+00:00	1.21288	1.21302	1.21113	1.21216	3112	0	0
2021-06-11 17:00:00+00:00	1.21215	1.21270	1.21006	1.21094	3725	0	0
2021-06-11 18:00:00+00:00	1.21093	1.21146	1.20928	1.21000	2722	0	0
2021-06-11 19:00:00+00:00	1.21000	1.21012	1.20948	1.20965	1594	0	0
2021-06-11 20:00:00+00:00	1.20966	1.21000	1.20926	1.20998	1135	0	0
2021-06-11 21:00:00+00:00	1.20997	1.21030	1.20976	1.21027	1176	0	0
2021-06-11 22:00:00+00:00	1.21028	1.21095	1.21016	1.21092	1016	0	0
2021-06-11 23:00:00+00:00	1.21092	1.21108	1.21043	1.21079	539	0	0

Figura 5.6: Fragmento de la matriz de históricos, EURUSD

Una vez dispongamos de la matriz de históricos, recorreremos cada una de las filas de ésta añadiendo, en una nueva columna que llamaremos “action”, la señal de actuación que nos hubiese arrojado la estrategia es ese instante, conociendo los h anteriores precios de cierre (close). La estrategia diseñada nos devolverá 3 posibles valores: $+1$ para señales de compra, -1 para señales de venta y 0 para señales de abstención.

Una vez conozcamos las decisiones que hubiese tomado la estrategia en cada uno de los periodos que conforman el marco temporal en el que estamos realizando el backtest, debemos determinar, de entre todas las operaciones ejecutadas, cuáles resultaron favorables y cuáles no.

Antes de continuar debemos definir una serie de conceptos que se emplean en el mundo de las inversiones, más en concreto, dentro del FOREX.

time	open	high	low	close	tick_volume	spread	real_volume	action
2021-06-10 22:00:00+00:00	1.21716	1.21766	1.21700	1.21704	986	0	0	0
2021-06-10 23:00:00+00:00	1.21703	1.21777	1.21700	1.21761	659	0	0	0
2021-06-11 00:00:00+00:00	1.21639	1.21764	1.21639	1.21740	311	3	0	0
2021-06-11 01:00:00+00:00	1.21740	1.21750	1.21706	1.21728	317	0	0	0
2021-06-11 02:00:00+00:00	1.21728	1.21750	1.21719	1.21728	294	0	0	0
2021-06-11 03:00:00+00:00	1.21728	1.21788	1.21718	1.21782	1027	0	0	0
2021-06-11 04:00:00+00:00	1.21782	1.21887	1.21760	1.21872	1569	0	0	0
2021-06-11 05:00:00+00:00	1.21871	1.21902	1.21831	1.21900	1159	0	0	-1
2021-06-11 06:00:00+00:00	1.21901	1.21914	1.21868	1.21872	745	0	0	-1
2021-06-11 07:00:00+00:00	1.21873	1.21904	1.21868	1.21872	480	0	0	-1
2021-06-11 08:00:00+00:00	1.21872	1.21902	1.21858	1.21879	741	0	0	-1
2021-06-11 09:00:00+00:00	1.21881	1.21930	1.21814	1.21834	1984	0	0	-1
2021-06-11 10:00:00+00:00	1.21834	1.21918	1.21748	1.21828	2874	0	0	0
2021-06-11 11:00:00+00:00	1.21828	1.21836	1.21640	1.21658	3101	0	0	0
2021-06-11 12:00:00+00:00	1.21657	1.21672	1.21527	1.21541	2509	0	0	0
2021-06-11 13:00:00+00:00	1.21542	1.21609	1.21490	1.21512	2395	0	0	0
2021-06-11 14:00:00+00:00	1.21512	1.21518	1.21298	1.21324	2944	0	0	0
2021-06-11 15:00:00+00:00	1.21324	1.21378	1.21225	1.21288	3068	0	0	1
2021-06-11 16:00:00+00:00	1.21288	1.21302	1.21113	1.21216	3112	0	0	1
2021-06-11 17:00:00+00:00	1.21215	1.21270	1.21006	1.21094	3725	0	0	1
2021-06-11 18:00:00+00:00	1.21093	1.21146	1.20928	1.21000	2722	0	0	1
2021-06-11 19:00:00+00:00	1.21000	1.21012	1.20948	1.20965	1594	0	0	1
2021-06-11 20:00:00+00:00	1.20966	1.21000	1.20926	1.20998	1135	0	0	1
2021-06-11 21:00:00+00:00	1.20997	1.21030	1.20976	1.21027	1176	0	0	1
2021-06-11 22:00:00+00:00	1.21028	1.21095	1.21016	1.21092	1016	0	0	1
2021-06-11 23:00:00+00:00	1.21092	1.21108	1.21043	1.21079	539	0	0	0

Figura 5.7: Fragmento de la matriz de históricos con “action”, EURUSD

En el mercado de divisas, un porcentaje en puntos o punto de interés de precio, pip, es una unidad de variación en el tipo de cambio de un par de divisas. Las principales monedas (excepto el yen japonés) se cotizan tradicionalmente con cuatro decimales, y un pip es una unidad del cuarto punto decimal: en el caso de divisas con céntimos, un pip equivale a la centésima parte de un céntimo. Para el yen, un pip es una unidad del segundo punto decimal, porque el valor del yen está mucho más cerca de la centésima parte de las otras divisas principales. A veces se confunde un pip con la unidad de cambio más pequeña de una cotización, es decir, el tick. Los pares de divisas a menudo se cotizan con cuatro decimales, pero el tick en un mercado dado puede ser, por ejemplo, de 5 pips o medio pip.

En el mundo de las inversiones, las órdenes Stop Loss (corte de pérdidas) y Take Profit (recogida de ganancias) son órdenes de compra o venta de instrumentos financieros condicionadas a que se alcance un precio previamente fijado por el inversor. Por ejemplo, supongamos que estamos operando en el par EURUSD, el cual tiene una cotización actual de $1,2111$; creemos que este valor ascenderá en el corto plazo hasta $1,2121$ para luego derrumbarse, por lo que, para recoger las ganancias en el instante anterior a la caída de precios, colocamos en la orden de compra un Take Profit de 10 pips. Para tratar de minimizar pérdidas en el caso de que el mercado reaccionara de manera opuesta a nuestra intuición, colocamos un Stop Loss de, por ejemplo, 5 pips, y así limitamos nuestras pérdidas y nos aseguramos no seguir perdiendo más.

El último término a definir es el spread. En el mercado de divisas, siempre hay dos precios dados en un par, el precio de venta y de compra. El precio de venta es el precio al que puedes vender la divisa base, mientras que el precio de compra es el precio que usarías para comprar la divisa base. La diferencia entre ambos precios se denomina spread. La mayoría de los pares de divisas de FOREX se negocian sin comisión, pero el spread es el coste que se aplica a cualquier operación que realicemos. Destacar que, aunque existan dos precios para el mismo par de divisas, cuando hablemos de precios

nos estaremos refiriendo al precio de compra del mismo.

Una vez definidos los anteriores términos, podemos continuar con el desarrollo del backtest. Para calcular el número de operaciones favorables generadas por la estrategia en el marco temporal a testear, recorreremos la columna "action" y, para cada valor distinto de cero que ésta presente, simularemos una orden de compra/venta en el mercado (de compra para $+1$ y de venta para -1). Esta orden irá ligada a un Take Profit y un Stop Loss, de tal manera que, una vez posicionada, iremos avanzando en el tiempo hasta que la cotización del par de divisas estudiado alcance uno de los dos precios establecidos con el Take Profit y Stop Loss. En ese instante, añadiremos a nuestro contador de ganancias/pérdidas, previamente inicializado en cero, los pips ganados/perdidos en la operación. Un último factor a tener en cuenta en las simulaciones es el spread. Con el fin de reforzar lo explicado, veamos un ejemplo.
Supongamos que queremos realizar un backtest a la estrategia diseñada en el par EURUSD. Una vez dispongamos de la matriz de cierres históricos con la columna "action" añadida, empezamos a recorrer dicha columna. Imaginemos que en las primeras posiciones la columna toma el valor 0, pero llegados a una determinada posición, nos encontramos un $+1$ en la columna. Entonces, simulamos la apertura de una posición de compra con un Stop Loss y Take Profit de, por ejemplo, 50 pips, y un spread de, por ejemplo, 1 pip. Si la cotización del EURUSD en ese instante era de $1,2150$, seguiremos avanzando en el tiempo, colocando las órdenes que vayan surgiendo, hasta que el precio del par alcance, o bien los $1,2201$ para recoger 50 pips de ganancias (debido a que el spread es de 1 pip, la cotización, o precio de compra, deberá de ser de $1,2201$ para que el precio de venta sea de $1,2200$), o bien los $1,2101$ para asumir una pérdida de 50 pips (de nuevo, debido al spread, para vender a $1,2100$ la cotización deberá situarse en $1,2101$). El último paso sería incluir la ganancia/pérdida resultante de esta operación en su respectivo contador y repetir el proceso con el resto de operaciones que nos encontremos en la columna "action".
Una vez completemos todas las operaciones de la estrategia en el marco temporal que la estemos testeando, a través de los contadores de ganancias y pérdidas resultantes, calcularemos el Profit Factor obtenido de operar con ésta. El Profit Factor, o factor de beneficio, es una de las métricas más populares y utilizadas en trading, la cual se calcula dividiendo el total ganado en las operaciones positivas entre el total perdido en las operaciones negativas, por lo que, en nuestro caso, le calcularemos dividiendo el contador de ganancias totales por el de pérdidas totales. Nuestro objetivo será encontrar pares de divisas en los que nuestra estrategia alcance un Profit Factor significativamente mayor a 1, ya que, aunque un buen desempeño pasado no garantiza un buen desempeño futuro, nos otorgará una mayor confianza a la hora de operar en ellos frente a pares en los que obtengamos un Profit Factor menor o igual a 1.

Una puntualización que debemos hacer es que, aunque al comienzo del capítulo hayamos supuesto que el indicador de las bandas de Bollinger sea contra tendencial, es decir, cuando el precio se aleja significativamente de la media móvil tiende a regresar a ésta, pueden existir pares de divisas que, debido a multitud de factores macroeconómicos, el indicador de las bandas de Bollinger otorgará mayor rendimiento si actúa de forma direccional, es decir, si suponemos que cuando el precio se aleja significativamente de su media entra en una tendencia, ascendente si rompe la banda de Bollinger superior y descendente si rompe la banda inferior. Este hecho debemos tenerle en cuenta a la hora de estudiar los resultados de un backtest. Por ejemplo, si realizamos un backtest a

la estrategia de las bandas de Bollinger basada en el filtro de Kalman, empleada de manera contra tendencial, durante los dos últimos años en el par EURUSD y obtenemos un Profit Factor de $0,5$, podemos afirmar que éste es un gran resultado, puesto que, aunque en las simulaciones, de cada 3 operaciones, 2 resultasen desfavorables, podemos cambiar las señales de compra por señales de venta y viceversa, es decir, emplear la estrategia tendencialmente, y obtener así un Profit Factor de 2.

El filtro de Kalman trata de solventar la principal problemática del indicador de las bandas de Bollinger basado en una media móvil, el retraso en las señales de compra/venta. Con el objetivo de estudiar la diferencia de rendimientos obtenidos de haber operado los últimos dos años en el mercado de divisas con ambas estrategias (el indicador de las bandas de Bollinger basado en una media móvil y el indicador de las bandas de Bollinger basado en el filtro de Kalman), realizaremos un backtest a cada una de ellas en los siguientes pares de divisas: EURUSD (euro - dólar estadounidense), AUDCHF (dólar australiano - franco suizo), GBPUSD (libra esterlina - dólar estadounidense), AUDUSD (dólar australiano - dólar estadounidense) y USDCAD (dólar estadounidense - dólar canadiense).

Debido a las circunstancias macroeconómicas que venimos experimentando en estos últimos 2 años, las cuales provocan grandes movimientos en los mercados, tanto alcistas como bajistas, lo más razonable es pensar que el indicador de las bandas de Bollinger predecirá, por norma general, cambios de tendencia en lugar de reversiones a la media. Un dato a tener en cuenta antes de realizar los backtest es que Australia cuenta con una economía sumamente desarrollada y es un estado rico en recursos naturales. El sector agrícola y de commodities constituyen aproximadamente el $60\,\%$ de las exportaciones del país, dentro de las cuales hay una gran variedad de commodities como el trigo, lana o metales preciosos como el oro. De esta manera el oro favorece a Australia como país exportador. Es por ello que el dólar australiano es una moneda que cuenta con una naturaleza altamente correlativa al oro, el cual, a su vez, tiene una alta correlación negativa con el dólar estadounidense, y como consecuencia, el dólar australiano (AUD) y el estadounidense (USD) están correlacionados negativamente. Por otro lado, el franco suizo es una de las monedas más beneficiadas cuando la aversión al riesgo predomina en el mercado. En tiempos de crisis e incertidumbre financiera esta divisa actúa de moneda refugio. El dólar estadounidense es otra de las principales monedas refugio, por lo que el CHF y el USD están correlacionados positivamente. Esto implica que existe una correlación negativa entre el franco suizo y el dólar canadiense. Es por esta alta correlación negativa entre las divisas AUD-CHF y AUD-USD que, a diferencia del resto de pares de divisas a estudiar, en los que operaremos de manera tendencial, en los pares AUDCHF y AUDUSD operaremos de manera contra tendencial.

A continuación se muestran los Profit Factor obtenidos en los backtest de los 5 pares de divisas mencionados para el indicador de las bandas de Bollinger basado en una media móvil con una ventana de 48 horas y un factor de 2 para las bandas. En las simulaciones hemos definido un Take Profit y un Stop Loss de 50 pips, y hemos simulado que el spread existente en las divisas es de medio pip. El marco temporal para las simulaciones han sido los dos años anteriores a la fecha de redacción de este trabajo (del 14/06/2019 al 14/06/2021).

	EURUSD	AUDCHF	GBPUSD	AUDUSD	USDCAD
Profit Factor	1,043	1,0327	1,046	0,9903	0,8857

Tabla 5.1: Bandas de Bollinger con media móvil

Del mismo modo, hemos realizado un backtest para cada uno de los 5 pares, pero esta vez operando en ellos con la estrategia de las bandas de Bollinger basada en el filtro de Kalman diseñado. Para los parámetros del filtro hemos tomado dimensión $n = 2$ y $\sigma_{measure} = 10^{-3}$ con el fin de obtener, cada 24 horas, el σ_{model} óptimo en comparación con una media móvil centrada de 48 horas. El resto de parámetros les hemos tomado iguales que el anterior caso: el factor de las bandas de Bollinger $k = 2$, el Take Profit y Stop Loss de 50 pips, el spread de medio pip y el marco temporal de 2 años.
A continuación se muestran los resultados de las simulaciones con esta estrategia.

	EURUSD	AUDCHF	GBPUSD	AUDUSD	USDCAD
Profit Factor	1,1388	1,1755	1,0574	1,2263	0,9232

Tabla 5.2: Bandas de Bollinger con el filtro de Kalman

Destacar que, tanto para la media móvil como para el filtro de Kalman, la estrategia ha ejecutado en cada par de divisas, de media, en torno a 2000 operaciones, las cuales nos proporcionan cierto grado de confianza a la hora de extraer conclusiones de los resultados.
En primer lugar, mencionar que, al contrario de lo que nos decía nuestra intuición, el indicador de las bandas de Bollinger, en los últimos 2 años, se ha comportado mejor contra tendencialmente que tendencialmente para operar en el par USDCAD, puesto que, con ambas estrategias, hemos obtenido un Profit Factor significativamente inferior a 1. De esta forma, reemplazando las órdenes de compra por órdenes de venta y viceversa obtendríamos un Profit Factor mayor a 1. En efecto, obtenemos un Profit factor de $1,0736$ para la estrategia basada en la media móvil y de $1,0311$ para la del filtro de Kalman. Podemos ver así que en el par USDCAD el filtro de Kalman no nos aporta ninguna ventaja frente a la media móvil. En cuanto al resto de divisas se refiere, vemos cómo para la estrategia basada en el filtro de Kalman todos los Profit Factor son significativamente mayores que 1 mientras que para la estrategia basada en la media móvil los Profit Factor son todos bastante cercanos a 1. Esto nos deja ver que la estrategia basada en la media móvil es inconsistente para los pares de divisas EURUSD, AUDCHF, GBPUSD y AUDUSD, mientras que la estrategia basada en el filtro de Kalman acaba con esta inconsistencia mejorando los Profit Factor de estos 4 pares. Podemos concluir así que, gracias a el filtro de Kalman, hemos mejorado significativamente la eficacia del indicador de las Bandas de Bollinger para predecir movimientos del mercado, ya sean reversiones a la media o cambios de tendencia.
Antes de finalizar esta sección realizaremos un breve estudio de la evolución del Profit Factor a lo largo de las simulaciones.
En la imagen 5.8 y 5.9 podemos apreciar cómo el Profit Factor, independientemente del par de divisas en el que operemos, va a tender cada vez más a 1. Es por este motivo que no hemos realizado el backtest a un periodo mucho más amplio, por ejemplo, 20 años. Sería erróneo pensar

que las condiciones macroeconómicas que afectan a las divisas de un par no van a cambiar en esos 20 años, y, por lo tanto, la rentabilidad de la estrategia no verse afectada.

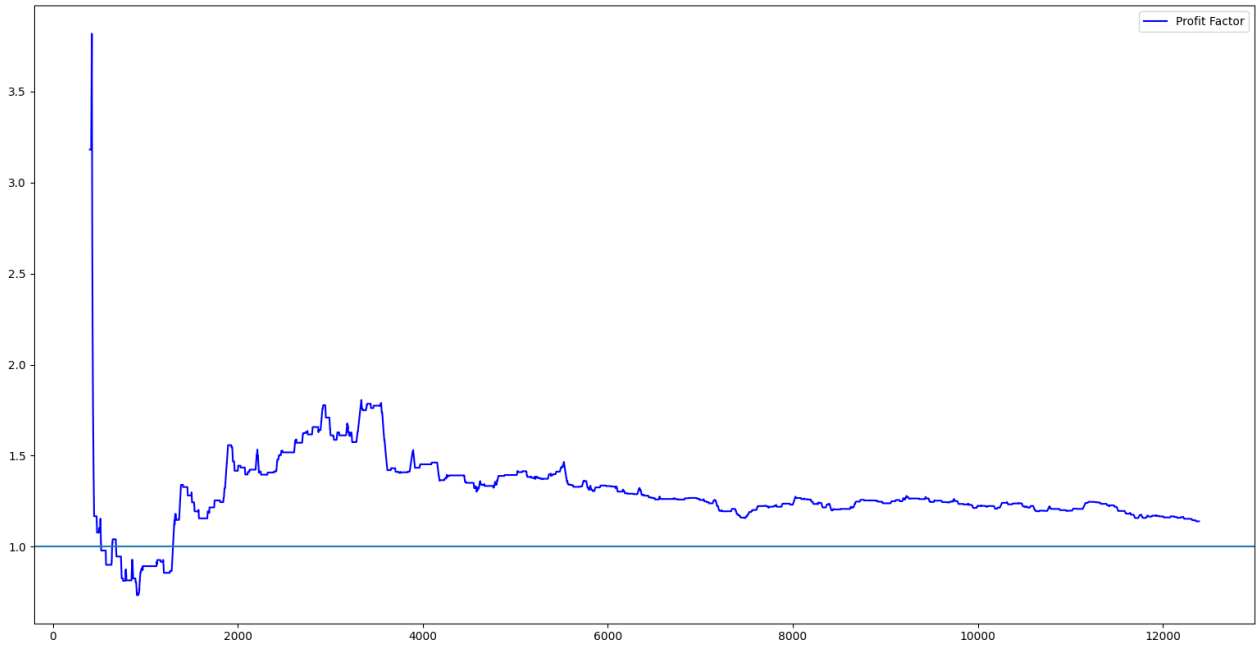


Figura 5.8: Evolución del Profit Factor, EURUSD

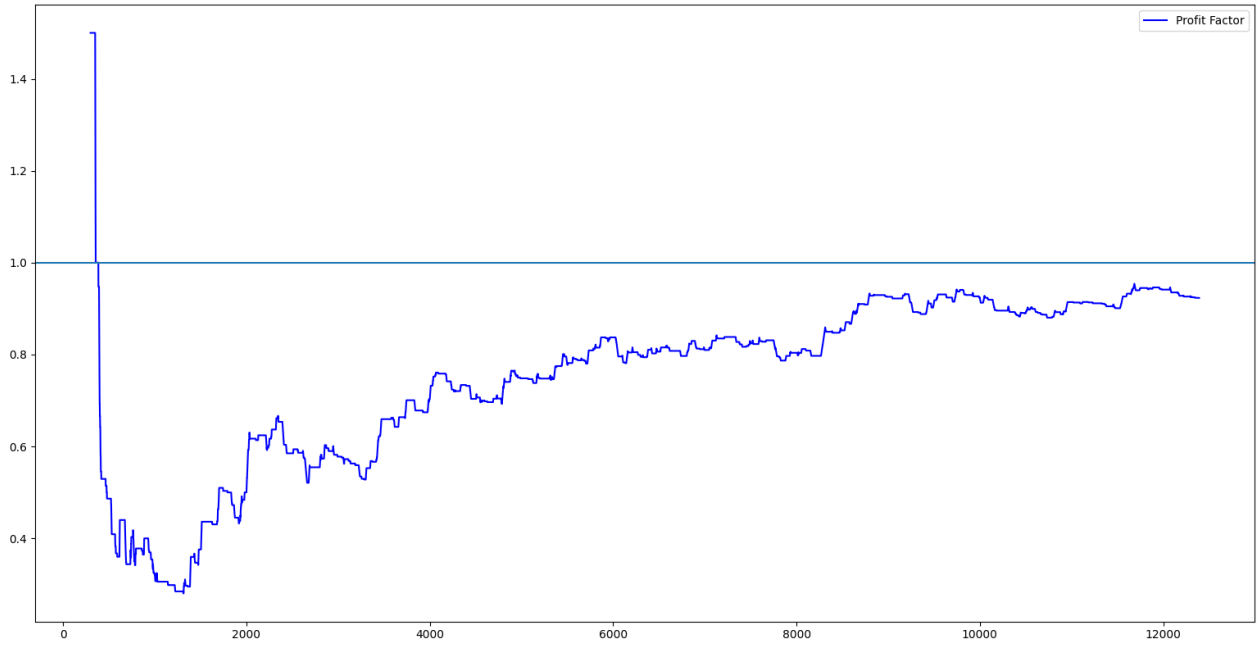


Figura 5.9: Evolución del Profit Factor, USDCAD

Por último, comentar que también hemos realizado estas mismas simulaciones optimizando los parámetros del filtro de Kalman cada hora en vez de cada 24, y los resultados arrojados durante los 2 últimos años apenas difieren en los 5 pares de divisas. Es por esto que, a la hora de operar en tiempo real, recomendamos seguir optimizando los parámetros del filtro con una frecuencia diaria y no horaria, para mejorar la eficiencia del algoritmo, además de no caer en un sobre entrenamiento

del modelo.

Todos los scripts que hemos desarrollado para la obtención de los resultados anteriores, así como los necesarios para obtener los resultados expuestos en el resto de capítulos se adjuntan en el Apéndice A.

5.4. Conclusiones

Dentro del campo del análisis técnico, desde su introducción en la década de los 80, las bandas de Bollinger se ha posicionado como uno de los indicadores técnicos preferidos por inversores y analistas de mercados. En este trabajo hemos solventado, gracias al filtro de Kalman, uno de los mayores conflictos con el que debemos lidiar al operar en base a las señales de compra/venta generadas por este indicador: el retraso en las mismas. Aunque el hecho de que la estrategia haya dado buenos resultados en los últimos 2 años no es garantía de que vaya a seguir dando buenos resultados en el futuro, la estrategia fundamentada en el filtro de Kalman nos aporta una mayor seguridad a la hora de operar en el mercado que la estrategia fundamentada en una media móvil. De los 5 pares de divisas en los que hemos simulado el comportamiento de la estrategia diseñada en este capítulo, en 4 de ellos, EURUSD, AUDCHF, GBPUSD y AUDUSD, la estrategia basada en el filtro de Kalman ha superado en rendimiento, con bastante margen, a la estrategia fundamentada en una media móvil. Por lo tanto, nuestra recomendación de actuación sería operar en estos 4 pares con el filtro de Kalman del mismo modo que en las simulaciones, vigilando las constantes macroeconómicas que pueden influir en el comportamiento de los pares, transformando éstos de tendenciales a contra tendenciales y viceversa.

Todavía quedan muchos resultados por descubrir. Por ejemplo, podríamos aplicar el filtro de Kalman que hemos diseñado a instrumentos financieros pertenecientes a otros mercados, como lo pueden ser el mercado de valores o el mercado de criptomonedas, y estudiar su comportamiento en ellos. Otra opción sería experimentar con distintos filtros de Kalman, puesto que la teoría de este filtro va mucho mas allá de la estudiada en este trabajo. Existen numerosos filtros de Kalman extendidos, en los cuales cambia alguna de las hipótesis del espacio de trabajo, por ejemplo, que el sistema a filtrar deja de ser lineal o que el vector del ruido de la observación no sigue una distribución normal. Podríamos probar a sustituir en nuestra estrategia de trading el filtro de Kalman que hemos diseñado por un filtro de Kalman extendido y comprobar la eficacia de los filtros de Kalman extendidos en el mercado de divisas u otros mercados financieros.

El filtro de Kalman es una herramienta que tradicionalmente se ha empleado en el campo de la orientación de vehículos, como coches autónomos o naves espaciales, o en el campo de la robótica, pero nos abre un enorme abanico de posibilidades cuando hablamos de aplicaciones en economía. Es por resultados como los expuestos a lo largo de este trabajo por lo que el filtro de Kalman está ganando cada vez más relevancia dentro del mundo de las finanzas, más concretamente, dentro del mundo de las inversiones.

Bibliografía

[1] Roger R Labbe Jr, *Kalman and Bayesian Filters in Python.*
https://elec3004.uqcloud.net/2015/tutes/Kalman_and_Bayesian_Filters_in_Python.pdf (pdf)
https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python (GitHub)

[2] Eric Benhamou, *Trend without hiccups - A Kalman filter approach.*
https://www.researchgate.net/publication/314934305_Trend_Without_Hiccups_-_A_Kalman_Filter_Approach

[3] https://www.mql5.com/en/articles/3886

[4] https://en.wikipedia.org/wiki/Kalman_filter

[5] https://www.kalmanfilter.net

[6] https://en.wikipedia.org/wiki/Covariance

[7] https://en.wikipedia.org/wiki/Expected_value

[8] https://en.wikipedia.org/wiki/Matrix_calculus

[9] https://en.wikipedia.org/wiki/Tikhonov_regularization

Apéndice A

Código en Python

A.1. El filtro de Kalman

```
from numpy import dot, eye, zeros, arange, array, outer
from scipy.linalg import inv
from math import factorial

## Clase del filtro de Kalman
class KalmanFilter():

        # Etapa de prediccion
        def predict(self):
                self.x = dot(self.F, self.x)
                self.P = dot(self.F,self.P).dot(self.F.T) + self.Q

        # Etapa de actualizacion
        def update(self, Z, n):
                y = Z - dot(self.H, self.x)
                S = dot(self.H, self.P).dot(self.H.T) + self.R
                K = dot(self.P, self.H.T).dot(inv(S))
                self.x = self.x + dot(K, y)
                I_KH = eye(n) - dot (K, self.H)
                self.P = dot(I_KH,self.P)

## Funcion para crear el filtro de Kalman
def KF(zs, sigma_meas, sigma_model, dt=0.1, n=2, Id=False):

        # Creamos la matriz F
        F = eye(n)
        for i in range(1,n):
                r = arange(n-i)
                F[r, r+i] = dt ** i / factorial(i)

        # Creamos la matriz H
        H = zeros((1,n)) ; H[0,0] = 1

        # Creamos la matriz Q
        if Id:
                Q = eye(n) * sigma_model ** 2
```

```
    else:
            w_k = []
            for n_i in range(n):
                    w_k.append(dt ** (n - n_i) / factorial(n - n_i))
            Q = outer(array(w_k), array(w_k).T) * sigma_model ** 2

    # Creamos la matriz R
    R = sigma_meas ** 2

    # Creamos el punto inicial
    x_ini = zeros((n, 1)) ; x_ini[0, 0] = zs[0]

    # Creamos la matriz P
    P = eye(n) * 0.001

    # Creamos el filtro de Kalman con las matrices dadas
    Kalman_filter = KalmanFilter()
    Kalman_filter.x = x_ini
    Kalman_filter.P = P
    Kalman_filter.Q = Q
    Kalman_filter.R = R
    Kalman_filter.F = F
    Kalman_filter.H = H

    # Recorremos el vector de mediciones y calculamos el estado del sistema
    us = []
    for t in range(len(zs)):
            # Etapa de prediccion
            Kalman_filter.predict()
            # Etapa de actualizacion
            Kalman_filter.update(zs[t],n)
            # Agregamos el valor filtrado al vector output
            us.append(Kalman_filter.x[0,0])

    return us
```

A.2. Optimización de los parámetros del filtro de Kalman

```
from scipy.optimize import minimize_scalar
from scipy.linalg import norm
from utils.kalman_filter import KF
from numpy import diff

## Funcion objetivo con regularizacion de Tikhonov
def fun_obj(sigma_model, sigma_meas, window_cma, df_train, delta, order):

    # Obtenemos los vectores kf y cma
    cma = df_train['close'].rolling(window=window_cma, center=True).mean()
    kf = KF(zs=df_train['close'], sigma_meas=sigma_meas, sigma_model=sigma_model)
    # Obtenemos la diferencia entre ambos vectores reemplazando nan por cero
    diff1 = (cma - kf).fillna(0)
    # Obtenemos la diferencia entre posiciones consecutivas del vector kf
```

```
    diff2 = diff(kf)

    return norm(diff1, ord=order) + delta * norm(diff2, ord=order)

## Funcion para calcular el sigma_model optimo
def optimize(df_train, window_cma, sigma_meas, delta=0, order=2):

    # Calculamos el sigma_model optimo en el intervalo (1e-5, sigma_meas)
    sigma_model = minimize_scalar(fun_obj, args=(sigma_meas, window_cma, df_train,
    delta, order), bounds=(1e-5, sigma_meas), method='bounded').x

    return sigma_model
```

A.3. Las banda de Bollinger

```
from numpy import where
import matplotlib.pyplot as plt
from utils.kalman_filter import KF

## Estrategia usando las bandas de Bollinger con kf y ma
def bollinger(df, column='close', sigma_meas=1e-3, sigma_model=1e-5, window_ma=21,
window_std=21, factor=2, contradir=True, plot_kf=False, plot_ma=False, signal=False):

    # Obtenemos las bandas de Bollinger
    df['KF'] = KF(zs=df[column], sigma_meas=sigma_meas, sigma_model=sigma_model)
    df['MA'] = df[column].rolling(window=window_ma).mean()
    df['STD'] = df[column].rolling(window=window_std).std()
    df['BOLLINGER_UP_KF'] = df['KF'] + factor * df['STD']
    df['BOLLINGER_UP_MA'] = df['MA'] + factor * df['STD']
    df['BOLLINGER_DOWN_KF'] = df['KF'] - factor * df['STD']
    df['BOLLINGER_DOWN_MA'] = df['MA'] - factor * df['STD']

    # Obtenemos la decision de la estrategia (contradireccional)
    df['action_kf'] = where(df[column] > df['BOLLINGER_UP_KF'], -1, 0)
    df['action_kf'] = where(df[column] < df['BOLLINGER_DOWN_KF'], 1, df['action_kf'])
    df['action_ma'] = where(df[column] > df['BOLLINGER_UP_MA'], -1, 0)
    df['action_ma'] = where(df[column] < df['BOLLINGER_DOWN_MA'], 1, df['action_ma'])

    # Si la estrategia es direccional, cambiamos compras por ventas y viceversa
    if not contradir:
        df['action_kf'] = -1 * df['action_kf']
        df['action_ma'] = -1 * df['action_ma']

    # Representamos los datos
    if plot_kf:
        plt.plot(df[column], c='grey', label='Market')
        plt.plot(df['KF'], c='blue', label='Kalman filter')
        plt.plot(df['BOLLINGER_UP_KF'], c='red', label='Bollinger up')
        plt.plot(df['BOLLINGER_DOWN_KF'], c='red', label='Bollinger down')
        if signal:
            plt.plot(df.loc[df['action_kf'] == 1, 'close'], '^g')
            plt.plot(df.loc[df['action_kf'] == -1, 'close'], 'vr')
```

```
        plt.title('Bollinger Bands KF') ; plt.show()
    if plot_ma:
        plt.plot(df[column], c='grey', label='Market')
        plt.plot(df['KF'], c='blue', label='Moving average')
        plt.plot(df['BOLLINGER_UP_MA'], c='red', label='Bollinger up')
        plt.plot(df['BOLLINGER_DOWN_MA'], c='red', label='Bollinger down')
        if signal:
            plt.plot(df.loc[df['action_ma'] == 1, 'close'], '^g')
            plt.plot(df.loc[df['action_ma'] == -1, 'close'], 'vr')
        plt.title('Bollinger Bands MA') ; plt.show()

    # Devolvemos la ultima decision tomada disponible para las 2 estrategias
    return df['action_kf'][len(df) - 1], df['action_ma'][len(df) - 1]
```

A.4. Backtesting

```
from utils.backtesting import BackTesting
from utils.utils import conexion, preprocesado
from utils.optimization import optimize
from utils.bollinger_bands import bollinger
from warnings import filterwarnings
from datetime import datetime, timedelta
import MetaTrader5 as mt5
import matplotlib.pyplot as plt

## Ignoramos los avisos y establecemos conexion con el servidor
filterwarnings('ignore')
conexion()

## Definimos la divisa y el timeframe
symbol = 'EURUSD'
timeframe = mt5.TIMEFRAME_H1

## Definimos los parametros de la estrategia de las bandas de Bollinger con KF
sigma_meas = 1e-3
window_std = 48
factor = 2
contradir = False

## Definimos los parametros que ajustan sigma_model
window_cma = 48
order = 2
delta = 20

## Definimos las variables del backtest
TP = 50
SL = 50
pip = mt5.symbol_info(symbol).point * 10
spread = 5

## Definimos la duracion del periodo de entrenamiento
hours_train = 14 * 24
```

```
## Definimos el periodo de backtest
date_to = datetime(2021,6,14,23,0,0)
date_from = datetime(2019,6,14,23,0,0)

## Descargamos el dataset
date_from_total = date_from - timedelta(hours=hours_train)
df = preprocesado(symbol, timeframe, date_from=date_from_total, date_to=date_to,
method='range')

## Comenzamos las iteraciones
df['action_kf'] = 0 ; df['action_ma'] = 0 ; iter = 0
for date in df['time'].loc[df['time'] > format(date_from)]:

    iter += 1
    print('ITER: {}, DATE: {}'.format(iter, date))

    # Obtenemos los historicos de entrenamiento
    date_from_train = date - timedelta(hours=hours_train)
    df_train_kf = df.loc[(df['time'] > date_from_train) & (df['time'] < date)]
    .copy().reset_index(drop=True)

    # Actualizamos sigma_model cada 24 horas
    #(al cierre del mercado, el cual es tras el cierre de la vela de las 23:00)
    if df_train_kf['time'][len(df_train_kf) - 1].hour %23 == 0 or iter == 1:
        sigma_model = optimize(df_train=df_train_kf, window_cma=window_cma,
        sigma_meas=sigma_meas, delta=delta, order=order)

    # Representamos cada 24 horas
    #(al cierre del mercado, el cual es tras el cierre de la vela de las 23:00)
    plot = False ; signal = True
    if df_train_kf['time'][len(df_train_kf) - 1].hour %23 == 0:
        plot = True

    # Obtenemos la posicion de la estrategia de las bandas de bollinger
    act_kf, act_ma = bollinger(df=df_train_kf.copy(), sigma_meas=sigma_meas,
    sigma_model=sigma_model, window_ma=window_cma, factor=factor,
    window_std=window_std, contradir=contradir, plot_kf=plot,
    plot_ma=plot, signal=signal)
    df.loc[df['time'] == date, 'action_kf'] = act_kf
    df.loc[df['time'] == date, 'action_ma'] = act_ma

## Creamos el dataframe para el backtest de la estrategia
df_backtest = df.loc[df['time'] > format(date_from)].copy().reset_index(drop=True)

## Realizamos el backtest a la estrategia basada en kf
backtest_kf = BackTesting(df_market=df_backtest, SL=SL, TP=TP, pip_value=pip,
points_spread=spread, action_column='action_kf', result_colum='Trade_Result',
just_one=False, save_df=df_backtest)
print('-'*10,'BACKTEST_KF','-'*10)
result_kf = backtest_kf.execute(metrics=True, verbose=True)

## Representamos las decisiones tomadas durante el proceso
```

```
plt.plot(result_kf['close'], c='b', label='market')
plt.plot(result_kf.loc[result_kf['action_kf'] == 1, 'close'], '^g')
plt.plot(result_kf.loc[result_kf['action_kf'] == -1, 'close'], 'vr')
plt.title('Strategy KF') ; plt.legend() ; plt.show()

## Representamos la evolucion del profit factor durante el proceso
plt.plot(result_kf['Profit_Factor'][400:], c='b', label='Profit Factor')
plt.axhline(y=1)
plt.title('Profit Factor KF') ; plt.legend() ; plt.show()

## Realizamos el backtest a la estrategia basada en ma
backtest_ma = BackTesting(df_market=df_backtest, SL=SL, TP=TP, pip_value=pip,
points_spread=spread, action_column='action_ma', result_colum='Trade_Result',
just_one=False, save_df=df_backtest)
print('-'*10,'BACKTEST MA','-'*10)
result_ma = backtest_ma.execute(metrics=True, verbose=True)

## Representamos las decisiones tomadas durante el proceso
plt.plot(result_ma['close'][400:], c='b', label='market')
plt.plot(result_ma.loc[result_ma['action_ma'] == 1, 'close'], '^g')
plt.plot(result_ma.loc[result_ma['action_ma'] == -1, 'close'], 'vr')
plt.title('Strategy MA') ; plt.legend() ; plt.show()

## Representamos la evolucion del profit factor durante el proceso
plt.plot(result_ma['Profit_Factor'], c='b', label='Profit Factor')
plt.axhline(y=1)
plt.title('Profit Factor MA') ; plt.legend() ; plt.show()
```

A.5. Funciones auxiliares

```
from pandas import DataFrame, to_datetime
from datetime import datetime
import MetaTrader5 as mt5

## Funcion para establecer conexion con MetaTrader5
def conexion(server='ICMarketsSC-Demo', login=********, password='********'):

        # Establecemos la conexion y en caso de error avisamos de este
        if not mt5.initialize(server=server, login=login, password=password):
                print("Error en el inicio de sesion.")
                mt5.shutdown()
                exit()

## Funcion para descargar historicos en un rango o dado un numero de fechas
def preprocesado(symbol, timeframe, date_from=datetime.now(),
date_to=datetime.now(), method='range', start_pos=0, count=0):

        # Descargamos los historicos del servidor
        if method == 'range':
                rates_frame = DataFrame(mt5.copy_rates_range(symbol, timeframe,
                date_from, date_to))
        elif method == 'pos':
```

```
            rates_frame = DataFrame(mt5.copy_rates_from_pos(symbol, timeframe,
            start_pos, count))
        else:
            raise Exception('Metodo no compatible.')

        # Damos formato a las fechas
        rates_frame['time'] = to_datetime(rates_frame['time'], unit='s',utc=True)

        return rates_frame
```

A.6. Gráficos auxiliares

```
from datetime import datetime, timedelta
from utils.utils import conexion, preprocesado
from utils.optimization import optimize
from utils.kalman_filter import KF
import matplotlib.pyplot as plt
import numpy as np
import MetaTrader5 as mt5

## Establecemos la conexion con el servidor
conexion()

""" SENSIBILIDAD DIMENSION """

## Definimos la divisa y el timeframe
symbol = 'EURGBP'
timeframe = mt5.TIMEFRAME_H1

## Descargamos los historicos
date_from = datetime.now() - timedelta(days=5)
date_to = datetime.now()
df = preprocesado(symbol=symbol, timeframe=timeframe, date_to=date_to,
date_from=date_from)

## Establecemos los parametros de los modelos
sigma_meas = 0.0005
sigma_model = 0.0002
Id = True

## Calculamos los vectores de filtrado
n = 1
kf1 = KF(zs=df['close'], n=n, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)
n = 2
kf2 = KF(zs=df['close'], n=n, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)
n = 3
kf3 = KF(zs=df['close'], n=n, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)
n = 4
kf4 = KF(zs=df['close'], n=n, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)

## Representamos los vectores calculados
plt.figure()
```

```
plt.plot(kf1, 'b', label='Kalman Filter 1D')
plt.plot(kf2, 'g', label='Kalman Filter 2D')
plt.plot(kf3, 'r', label='Kalman Filter 3D')
plt.plot(kf4, 'cyan', label='Kalman Filter 4D')
plt.plot(df['close'], 'm—', label='Measure')
plt.legend() ; plt.axis('off') ; plt.show()

""" SENSIBILIDAD S_MODEL """

## Creamos el dataset
muestra = 100
dt = 0.1
noise = 0.03
t1 = (np.arange(muestra) - muestra / 2) * dt
z1 = np.sin(t1)
sin = []
for i in range(muestra):
    sin.append(z1[i] + np.random.normal(0, noise))
muestra = 200
dt = 0.1
noise = 0.01
t2= (np.arange(muestra)-muestra/2) * dt
z2 = 1/(1+np.exp(-3*t2))
sigm = []
for i in range(muestra):
    sigm.append(z2[i] + np.random.normal(0, noise))


## Establecemos los parametros de los modelos
n = 2
sigma_meas = 0.03

## Calculamos los vectores de filtrado (sin)
sigma_model = 0.01
kf1 = KF(zs=sin, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)
sigma_model = 0.02
kf2 = KF(zs=sin, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)
sigma_model = 0.05
kf3 = KF(zs=sin, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)

## Representamos los vectores calculados (sin)
plt.figure()
plt.plot(t1, kf1, 'b', label='Kalman Filter 2D, s = 0.01')
plt.plot(t1, kf2, 'g', label='Kalman Filter 2D, s = 0.02')
plt.plot(t1, kf3, 'r', label='Kalman Filter 2D, s = 0.05')
plt.plot(t1, sin, 'm—', label='Measure')
plt.legend() ; plt.axis('off') ; plt.show()

## Calculamos los vectores de filtrado (sigm)
sigma_model = 0.01
kf1 = KF(zs=sigm, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)
sigma_model = 0.02
kf2 = KF(zs=sigm, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)
```

```
sigma_model = 0.05
kf3 = KF(zs=sigm, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)

## Representamos los vectores calculados (sigm)
plt.figure()
plt.plot(t2, kf1, 'b', label='Kalman Filter 2D, s = 0.01')
plt.plot(t2, kf2, 'g', label='Kalman Filter 2D, s = 0.02')
plt.plot(t2, kf3, 'r', label='Kalman Filter 2D, s = 0.05')
plt.plot(t2, sigm, 'm--', label='Measure')
plt.legend() ; plt.axis('off') ; plt.show()

""" SENSIBILIDAD S_MEASURE """

## Establecemos los parametros de los modelos
n = 2
sigma_model = 0.03

## Calculamos los vectores de filtrado (sin)
sigma_meas = 0.01
kf1 = KF(zs=sin, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)
sigma_meas = 0.02
kf2 = KF(zs=sin, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)
sigma_meas = 0.05
kf3 = KF(zs=sin, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)

## Representamos los vectores calculados (sin)
plt.figure()
plt.plot(t1, kf1, 'b', label='Kalman Filter 2D, s = 0.01')
plt.plot(t1, kf2, 'g', label='Kalman Filter 2D, s = 0.02')
plt.plot(t1, kf3, 'r', label='Kalman Filter 2D, s = 0.05')
plt.plot(t1, sin, 'm--', label='Measure')
plt.legend() ; plt.axis('off') ; plt.show()

## Calculamos los vectores de filtrado (sigm)
sigma_meas = 0.01
kf1 = KF(zs=sigm, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)
sigma_meas = 0.02
kf2 = KF(zs=sigm, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)
sigma_meas = 0.05
kf3 = KF(zs=sigm, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=Id)

## Representamos los vectores calculados (sigm)
plt.figure()
plt.plot(t2, kf1, 'b', label='Kalman Filter 2D, s = 0.01')
plt.plot(t2, kf2, 'g', label='Kalman Filter 2D, s = 0.02')
plt.plot(t2, kf3, 'r', label='Kalman Filter 2D, s = 0.05')
plt.plot(t2, sigm, 'm--', label='Measure')
plt.legend() ; plt.axis('off') ; plt.show()

""" SENSIBILIDAD MATRIZ Q """

## Establecemos los parametros de los modelos
n = 2
```

```
sigma_meas = 0.01
sigma_model = 0.05

## Calculamos los vectores de filtrado (sin)
kf1 = KF(zs=sin, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=True)
kf2 = KF(zs=sin, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=False)

## Representamos los vectores calculados (sin)
plt.figure()
plt.plot(t1, kf1, 'b', label='Kalman Filter 2D, Q = Id')
plt.plot(t1, kf2, 'g', label='Kalman Filter 2D, Q = Q_cap2')
plt.plot(t1, sin, 'm--', label='Measure')
plt.legend() ; plt.axis('off') ; plt.show()

## Calculamos los vectores de filtrado (sigm)
kf1 = KF(zs=sigm, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=True)
kf2 = KF(zs=sigm, sigma_meas=sigma_meas, sigma_model=sigma_model, Id=False)

## Representamos los vectores calculados (sigm)
plt.figure()
plt.plot(t2, kf1, 'b', label='Kalman Filter 2D, Q = Id')
plt.plot(t2, kf2, 'g', label='Kalman Filter 2D, Q = Q_cap2')
plt.plot(t2, sigm, 'm--', label='Measure')
plt.legend() ; plt.axis('off') ; plt.show()

""" BOLLINGER BANDS """

## Definimos la divisa y el timeframe
symbol = 'EURGBP'
timeframe = mt5.TIMEFRAME_H1

## Descargamos los historicos
date_from = datetime(2021,5,19,0,0,0)
date_to = datetime(2021,5,30,0,0,0)
df = preprocesado(symbol=symbol, timeframe=timeframe, date_to=date_to,
date_from=date_from)

## Establecemos los parametros de los modelos
window = 24
sigma_meas = 1e-3
sigma_model = 1e-3

## Calculamos los vectores de filtrado
df['ma'] = df['close'].rolling(window=window).mean()
df['cma'] = df['close'].rolling(window=window, center=True).mean()
df['kf'] = KF(zs=df['close'], sigma_meas=sigma_meas, sigma_model=sigma_model)
df['std'] = df['close'].rolling(window=window).std()
df['up_ma'] = df['ma'] + 2 * df['std']
df['down_ma'] = df['ma'] - 2 * df['std']
df['up_cma'] = df['cma'] + 2 * df['std']
df['down_cma'] = df['cma'] - 2 * df['std']
df['up_kf'] = df['kf'] + 2 * df['std']
df['down_kf'] = df['kf'] - 2 * df['std']
```

```
## Representamos los vectores calculados
plt.plot(df['close'], c='grey', linestyle='—', label='market')
plt.plot(df['ma'], c='green', label='moving_average')
plt.plot(df['up_ma'], c='purple')
plt.plot(df['down_ma'], c='purple')
plt.legend() ; plt.axis('off') ; plt.show()
plt.plot(df['close'], c='grey', linestyle='—', label='market')
plt.plot(df['cma'], c='red', label='centered_moving_average')
plt.plot(df['up_cma'], c='purple')
plt.plot(df['down_cma'], c='purple')
plt.legend() ; plt.axis('off') ; plt.show()
plt.plot(df['close'], c='grey', linestyle='—', label='market')
plt.plot(df['kf'], c='blue', label='kalman_filter')
plt.plot(df['up_kf'], c='purple')
plt.plot(df['down_kf'], c='purple')
plt.legend() ; plt.axis('off') ; plt.show()

""" KF vs CMA """

## Definimos la divisa y el timeframe
symbol = 'EURUSD'
timeframe = mt5.TIMEFRAME_H1

## Descargamos los historicos
date_from = datetime(2021,3,30,18,0,0)
date_to = datetime(2021,4,15,18,0,0)
df = preprocesado(symbol=symbol, timeframe=timeframe, date_to=date_to,
date_from=date_from)

## Establecemos los parametros de los modelos
window = 24
sigma_meas = 0.002
sigma_model = 0.001

## Calculamos los vectores de filtrado
sigma_model_1 = optimize(df_train=df, window_cma=window, sigma_meas=sigma_meas,
order=2, delta=0)
sigma_model_2 = optimize(df_train=df, window_cma=window, sigma_meas=sigma_meas,
order=2, delta=10)
df['kf_1'] = KF(zs=df['close'], sigma_meas=sigma_meas, sigma_model=sigma_model_1)
df['kf_2'] = KF(zs=df['close'], sigma_meas=sigma_meas, sigma_model=sigma_model_2)
df['cma'] = df['close'].rolling(window=window, center=True).mean()

## Representamos los vectores calculados
plt.plot(df['close'], c='grey', linestyle='—', label='market')
plt.plot(df['kf_1'], c='blue', label='kalman_filter')
plt.plot(df['cma'], c='red', label='centered_moving_average')
plt.legend() ; plt.axis('off') ; plt.show()
plt.plot(df['close'], c='grey', linestyle='—', label='market')
plt.plot(df['kf_2'], c='blue', label='kalman_filter')
plt.plot(df['cma'], c='red', label='centered_moving_average')
plt.legend() ; plt.axis('off') ; plt.show()
```