UNIVERSIDAD DE GRANADA

TRABAJO FIN DE GRADO

Doble Grado en Ingeniería Informática y Matemáticas

USO DE TÉCNICAS DE ANÁLISIS DE DATOS EN LA DETERMINACIÓN DE VARIABLES SIGNIFICATIVAS EN LA EVOLUCIÓN DE LA EPIDEMIA COVID 19

Autor

Ignacio Garach Vélez

Tutor

Juan Campos Rodríguez

Facultad de Ciencias

E.T.S. de Ingenierías Informática y de Telecomunicación

Granada, a 26 de junio de 2023

Declaración de originalidad

D. Ignacio Garach Vélez

Declaro explícitamente que el trabajo presentado como Trabajo de Fin de Grado (TFG), correspondiente al presente curso académico 2022-2023, es original, entendido en el sentido de que no se han utilizado para la elaboración del trabajo fuentes sin citarlas debidamente.

En Granada a 26 de junio de 2023
Fdo: Ignacio Garach Vélez

Agradecimientos

Quiero agradecer a mi familia y amigos todo su apoyo durante estos 5 años. Sin él no habría sido posible llegar hasta aquí. Siempre hemos sabido encontrar buenos momentos que me han llenado de fuerza cuando faltaba la motivación.

A mi tutor, Juan Campos, que ha sabido aconsejarme y trabajar conmigo desarrollando este trabajo, agradecerle su tiempo y disponibilidad. De igual forma, al resto de profesores que me han enseñado durante estos 5 años a disfrutar las matemáticas y la informática.

Finalmente a mis compañeros de clase, que desde el primer día hemos sabido ayudarnos y compartir nuestro tiempo de estudio y de clase y, por supuesto, también fuera de ella.

Resumen

La pandemia de COVID-19 es uno de los mayores desafíos a los que la sociedad se ha enfrentado en tiempos modernos. Ha tenido un gran impacto en nuestras vidas, afectando la economía global, la educación, el empleo y la salud mental. Para hacerle frente, muchos países han implementado medidas de distanciamiento social y cierre de negocios, lo que ha llevado a una crisis económica mundial.

En este contexto, se han recopilado gran cantidad de variables relacionadas con la pandemia de COVID-19, desde el número de casos y muertes hasta las medidas gubernamentales, datos de movilidad de nuestros dispositivos y de disponibilidad de recursos médicos y vacunas. En este trabajo nuestro objetivo se centrará en utilizar el análisis de componentes principales para explorar y comprender mejor los patrones y las relaciones entre estos datos. Concretamente, trataremos de extraer una variable nueva que recoja de la mejor forma posible el fenómeno intrínseco que describen en conjunto todos los datos con el objetivo de obtener un indicador valioso sobre el estado global de propagación e impacto de la pandemia en cada momento.

De forma adicional, se va a implementar una herramienta software en Python que permita la exploración de los datos tratados, así como la aplicación del método de análisis de forma simple y visual. Se podrá aplicar el método libremente a los rangos temporales deseados y llevar a cabo comparaciones de la variable indicadora extraída con el resto de datos. Igualmente se podrán exportar los datos resultantes para posteriores usos.

PALABRAS CLAVE: *fenómeno intrínseco, COVID-19, componentes principales, patrones, Python*

Extended Abstract

The SARS-CoV-2 pandemic is one of the greatest challenges that society has faced in modern times. The pandemic has had a huge impact on our lives, affecting the global economy, education, employment, and mental health. To cope with the pandemic, many countries have implemented social distancing measures and business closures, leading to a global economic crisis. In this context, a large number of variables related to the COVID-19 pandemic have been collected, from the number of cases and deaths to government measures, mobility data from our devices and the availability of medical resources and vaccines.
The present work focuses on the application of Principal Component Analysis (PCA) to analyze the impact of the COVID-19 pandemic in Spain and its theoretical development. In addition, we aim to develop a web application that implements the PCA method, allowing users to apply it to different temporal ranges, making easier the analysis of the pandemic's different waves.

This final project begins with an introduction that provides an overview of the COVID-19 pandemic and the importance of analyzing its impact. It also highlights the need for a comprehensive approach that considers not only health-related variables but also social, economic, climatic, and government measures. The introduction concludes by presenting the research questions and the organization of the chapters.

Chapter 2 provides a theoretical foundation by explaining concepts that next chapters will need to develop the method. It covers topics such as random variables and their properties leading into the concept of covariance matrix. It also introduces geometrical background, focusing on quadratic forms, eigenvalues and eigenvectors (diagonalization).

Chapters 3, 4 & 5 develop the method of Principal Component Analysis through the idea of maximizing variance of a new crafted *summary* variable that collects the intrinsic phenomenon of the pandemics. We link the variance of this variable with the values of a quadratic form associated to the covariance matrix. Then, the focus shifts to optimizing the quadratic form on the unit sphere so variance of the *summary* variable is also optimized. The chapter explores the need to restrict the quadratic form to ensure the existence of a maximum and discusses the use of compactness, specifically the unit sphere, as a restriction to guarantee a well-defined maximum while leaving as posibilities all directions of the space. It derives a theorem on extreme values and establishes its application to the PCA problem. This connection between optimizing the quadratic form and traditional PCA lays the foundation for further analysis and application in subsequent chapters. The chapter also discusses the benefits and limitations of PCA and its relevance in the context of the COVID-19 analysis.

Chapter 6 presents the data collection process, which combines data from various sources, including the COVID-19 Open Data Repository [29] by Google Health and the OWID COVID repository [14]. The chapter describes the types of variables considered for the analysis, such as health-related data (daily cases, deaths, hospitalizations), social and government measures (quarantine levels, transportation restrictions), and mobility data (gathered by Google). It explains how the data was cleaned, processed, and integrated into a coherent dataset suitable for PCA.

Chapter 7 presents the results and findings obtained from the method of analysis. We discuss the implications of the findings in terms of understanding the dynamics of the pandemic, identifying key factors and comparing the trends of the principal component with the rest of variables.

Chapter 8 focuses on the methodology adopted for the development of the web application. The incremental development methodology is described, highlighting its advantages in providing functional prototypes and accommodating changes during the development process. The chapter outlines the steps involved in building the application, including conceptual design, implementation of PCA algorithm, and integration of additional datasets. The chosen technologies and tools for the development are also discussed.

Chapter 9 is a user guide for the web application. It provides an overview of the user interface design, focusing on its usability and intuitiveness. The chapter explains how users can select different temporal ranges for analysis, choose variables of interest, and interpret the results obtained from the method. The integration of visualizations is also discussed to enhance the understanding of the data.

Chapter 10 provides a description of the process followed to deploy the application on the internet. We discuss benefits of using a Cloud Computing Provider as Google Cloud and the steps to deploy the Dash application at Google Cloud Platform.

Finally, we gather all the results showed at the chapters to reach the conclusions and discuss whether initial objectives have been fullfilled.

KEYWORDS: *intrinsic phenomenon, COVID-19, principal component analysis, covariance, Python*

Índice

1 Introducción y objetivos del trabajo

La pandemia de COVID-19 es uno de los mayores desafíos a los que la sociedad se ha enfrentado en tiempos modernos. Ha tenido un gran impacto en nuestras vidas, afectando la economía global, la educación, el empleo y la salud mental. Para hacerle frente, muchos países han implementado medidas de distanciamiento social y cierre de negocios, lo que ha llevado a una crisis económica mundial.

La enfermedad por COVID-19 es causada por un virus altamente contagioso conocido como SARS-CoV-2, que se transmite de persona a persona, principalmente a través de las gotículas respiratorias expulsadas por alguien infectado. Aunque los síntomas pueden variar en su gravedad la enfermedad puede ser particularmente peligrosa para las personas mayores y aquellas con afecciones preexistentes.

En este contexto, se han recopilado gran cantidad de variables relacionadas con la pandemia de COVID-19, desde el número de casos y muertes hasta las medidas gubernamentales y la disponibilidad de recursos médicos y vacunas. En este trabajo nuestro objetivo se centrará en utilizar el análisis de componentes principales para explorar y comprender mejor los patrones y las relaciones entre estos datos. Concretamente trataremos de extraer una variable nueva que recoja de la mejor forma posible el fenómeno intrínseco que describen en conjunto todos los datos con el objetivo de obtener un indicador valioso sobre el estado global de propagación e impacto de la pandemia en cada momento.

Lo que hoy en día se conoce como método de análisis de componentes principales se atribuye en primera instancia al matemático británico Karl Pearson en su artículo de 1901 *On lines and planes of closest fit to systems of points in space* [21]. Aunque en muchas ocasiones también se atribuye a Howard Hotelling que aportó una formulación más parecida a la actual y acuñó el término *componentes principales* en *Analysis of a complex of statistical variables into principal components* [7] .

Vamos a centrar nuestro interés en extraer la primera componente principal dado un conjunto de variables aleatorias $X_1, ..., X_n$ en el que cada una de ellas representa una variable de interés de la pandemia a lo largo de observaciones diarias. La primera componente principal será una nueva variable aleatoria combinación lineal de las anteriores de modo que recoja la máxima información y variabilidad de todas las demás. Es decir una variable resumen X_r que maximize la varianza.

Para lograr formular este problema de teoría de probabilidad en términos geométricos, debemos olvidar el uso descriptivo clásico de las principales pro-

piedades de las variables aleatorias: su media y su varianza. Estandarizaremos las variables de partida para evitar que aquellas con mayor escala influyan en mayor medida en la variable resumen. De esta forma, las variables estandarizadas podrían considerarse vectores de un espacio vectorial.

Consideraremos la matriz de covarianzas de este nuevo conjunto de variables estandarizadas para cuantificar las relaciones que existen entre ellas. Definiremos una forma cuadrática asociada a las covarianzas y para poder optimizarla la restrigiremos a un espacio compacto donde podremos deducir un teorema de valores extremos. Finalmente, demostraremos que el máximo se alcanza en el vector propio asociado al mayor valor propio y, de hecho, el máximo coincide con dicho valor. Por completitud, finalizaremos con un teorema espectral que proporciona un método para obtener el resto de componentes principales.

La variable resumen extraída contendrá la máxima variabilidad posible de los datos y por tanto, podría usarse para ajustar diferentes modelos de epidemiología, como el modelo SIR, o incluso para ajustar modelos de predicción futura como el modelo autoregresivo ARIMA. Esta opción queda abierta para un trabajo futuro.

En nuestro caso, vamos a aplicar el método para extraer la variable resumen de los datos correspondientes a nuestro país. Para ello se han recopilado datos sobre el número de casos, pruebas realizadas, hospitalizaciones, ingresos en UCI, etc... Y otras métricas sobre movilidad, clima o sobre medidas gubernamentales que amplían el ámbito del análisis. Se ha encontrado un problema en los datos de salud, pues existe un sesgo en la toma de datos, los fines de semana en la mayoría de zonas no se actualizaban los datos y se agregaban al dato de los lunes. De esta forma aparece una periodicidad que hace que las tendencias esten distorsionadas. Discutimos varias posibles soluciones, algunas sencillas como agrupar semanalmente, otras más complejas como el uso de medias móviles o el método de descomposición *Seasonal Trend Decomposition* propuesto en [2].

Para finalizar las secciones matemáticas del trabajo, se analizan los resultados y se realizan comparaciones con las variables de partida. Un resultado interesante que se obtiene es que la tendencia de la variable resumen está altamente relacionada con la curva de infectados, esto sugiere una posible relación con el modelo epidemiológico SIR. De igual forma, la variable también está relacionada con el número de fallecidos pero con un desfase al pasado de en torno a 7 días. También se realizan análisis interesantes comparando con las tasas de cambio en la movilidad.

De forma adicional se va a implementar una herramienta software que permita la exploración de los datos tratados así como la aplicación del método de

análisis de forma simple y visual. Se podrá aplicar el método libremente a los rangos temporales deseados y llevar a cabo comparaciones de la variable indicadora extraída con el resto de datos. Igualmente se podrán exportar los datos resultantes para posteriores usos.

Como se va a desarrollar software, por las características del proyecto, se ha escogido una metodología de desarrollo incremental y se ha realizado todo el proceso de planificación, presupuesto, análisis de requisitos, casos de uso y modelado. Se describen las herramientas utilizadas donde destaca el framework Dash. Esta herramienta permite crear aplicaciones web interactivas con gráficas para visualizar datos de forma sencilla. Permite combinar código Python, HTML y CSS para crear paneles de control y aplicaciones interactivas que respondan a las acciones del usuario mediante un sistema de callbacks.

Finalmente, se estudiará el despliegue de la aplicación web en la plataforma de computación en la nube de Google (GCP). Se ha tomado esta decisión principalmente porque no se dispone de infraestructura local estable para el despliegue. Además, en la actualidad, el modelo de computación en la nube es ampliamente adoptado por su facilidad de uso, escalabilidad y bajo coste.

Por todo ello, los objetivos de este trabajo son los siguientes:

- Estudiar y desarrollar el método de análisis de componentes principales.
- Recopilar y limpiar series temporales con datos de la epidemia en España.
- Aplicar el método a dichos datos y realizar un análisis comparativo.
- Desarrollar una aplicación web para facilitar la ejecución y difusión del método.
- Desplegar la aplicación mediante un modelo de computación en la nube.

2 Preliminares

2.1 Variables aleatorias

Para poder definir con rigor el concepto de variable aleatoria necesitamos introducir conceptos básicos de teoría de probabilidad. Comenzaremos con los elementos necesarios para considerar un espacio de probabilidad y sus axiomas principales y a partir de ellos podremos definir las variables aleatorias y sus propiedades fundamentales. Toda esta subsección se ha desarrollado en base a [23].

Vamos a considerar un conjunto Ω no vacío que llamaremos espacio de sucesos elementales. Sus elementos o sucesos elementales los notamos como ω_i.

Definición 1 *Sea $\mathcal{A}$ un conjunto compuesto por subconjuntos de Ω que cumple las siguientes propiedades:*

1. *Si $A \in \mathcal{A}$ entonces $\Omega \setminus A \in \mathcal{A}$*
2. *Si $A_1, A_2, \ldots$ es un conjunto numerable de subconjuntos pertenecientes a $\mathcal{A}$, entonces $\bigcup_n A_n \in \mathcal{A}$*

El conjunto $\mathcal{A}$ se conoce como $\sigma-$álgebra de sucesos y sus elementos se llaman sucesos.

A partir de las dos propiedades anteriores, se puede deducir que toda $\sigma-$álgebra de sucesos es un conjunto de subconjuntos de un espacio de sucesos elementales Ω que contiene para cada elemento a su complemento y para cualquier conjunto numerable de sus elementos, a su unión y a su intersección. Además, el propio Ω y su complemento $\emptyset$ también son sucesos.

Asociada a esta definición, en 1933 el matemático Kolmogorov propuso la siguiente axiomática que se mantiene en la actualidad:

- **Axioma I.** A cada suceso A se le puede asociar un número no negativo $P(A)$ llamado probabilidad del suceso A
- **Axioma II.** La probabilidad total es $P(\Omega) = 1$
- **Axioma III.** Si $A_1, A_2, \ldots$ es un conjunto numerable de sucesos incompatibles (su intersección es el suceso imposible) dos a dos, entonces

$$P(\bigcup_i A_i) = \sum_i P(A_i)$$

En el lenguaje del análisis real, la probabilidad P es una medida positiva que además cumple que $P(\Omega) = 1$. Por todo ello, definimos el concepto de espacio de probabilidad:

Definición 2 *La terna $(\Omega, \mathcal{A}, P)$ donde P es una probabilidad definida para cualquier suceso de la $\sigma-$álgebra $\mathcal{A}$ que verifica el sistema de axiomas de Kolmogorov, se llama espacio de probabilidad. Un espacio de probabilidad es un espacio medible $(\Omega, \mathcal{A})$ con una medida no negativa P tal que $P(\Omega) = 1$.*

Teniendo como punto de partida esta axiomática, se pueden demostrar una larga lista de propiedades elementales de los espacios de probabilidad que no introducimos para no alargar en exceso el desarrollo. Necesitamos una última definición relativa a sucesos:

Definición 3 *Consideramos dos sucesos cualesquiera A y B de un espacio de probabilidad $(\Omega, \mathcal{A}, P)$. Diremos que los sucesos A y B son independientes si*

$$P(A \cap B) = P(A)P(B)$$

Intuitivamente lo serán si el hecho de que A haya ocurrido no afecta a la probabilidad de que ocurra B y viceversa.

Ya estamos preparados para definir el concepto central de esta teoría:

Definición 4 *Consideramos un espacio de probabilidad $(\Omega, \mathcal{A}, P)$. Llamaremos variable aleatoria a una función $X = X(\omega)$ con valores reales definida en el espacio de sucesos elementales Ω verificando que*

$$\{\omega \in \Omega : X(\omega) \leq x\} \in \mathcal{A} \qquad \forall x \in \mathbb{R} \tag{2.1}$$

Aunque esta definición parezca muy abstracta, en términos analíticos se simplifica, una función $X(\omega)$ que cumple la condición de arriba es una función medible. Entonces una variable aleatoria no es más que una función real y medible de los sucesos elementales. Podemos escribir la condición en función de conjuntos borelianos de $\mathbb{R}$:

$$\{\omega \in \Omega : X(\omega) \in B\} \in \mathcal{A} \tag{2.2}$$

para cualquier boreliano de puntos de la recta real.

Asociada al concepto de variable aleatoria podemos definir la función de distribución. Como el conjunto $\{\omega \in \Omega : X(\omega) \leq x\}$ es un suceso, está definida la probabilidad $P(\{\omega : X(\omega) \leq x\})$ para todo $x \in \mathbb{R}$ que notaremos por simplicidad $P(X \leq x)$.

Definición 5 *Llamaremos función de distribución de una variable aleatoria X a la función real de variable real*

$$F(x) = P(X \leq x)$$

Por otro lado, si X es una variable aleatoria y B un boreliano de puntos de $\mathbb{R}$ está definida la probabilidad $P(X \in B) = P(\{\omega : X(\omega) \in B\})$.

Definición 6 *Llamaremos función de probabilidad de una variable aleatoria X a la función de conjunto*

$$P_X(B) = P(X \in B)$$

Algunas propiedades interesantes de la función de distribución son:

- **Propiedad I.** Se verifica que $0 \leq F(x) \leq 1$ para todo x real.
- **Propiedad II.** Si $a < b$, entonces $P(a < X \leq b) = F(b) - F(a)$
- **Propiedad III.** $F(x)$ es no decreciente

2.1.1 Variable aleatoria discreta

Definición 7 *Diremos que una variable aleatoria X tiene una distribución discreta, si existe un conjunto numerable $E_X \subset \mathbb{R}$ tal que*

$$P(X \in E_X) = 1 \tag{2.3}$$

Y, por tanto $P(X \in E_X^c) = 1 - P(X \in E_X) = 0$, luego

$$P(X = x) = 0 \qquad \forall x \notin E_X \tag{2.4}$$

probando que E_X es el conjunto de valores de la variable.

Por esta definición, es fácil intuir que necesitamos conocer la probabilidad de que la variable tome cada uno de sus valores para poder describir su comportamiento. Para ello definimos la función masa de probabilidad:

Definición 8 *Si X es una variable aleatoria discreta que toma valores en $E_X \subset \mathbb{R}$ se puede definir su función masa de probabilidad como*

$$\begin{aligned} p_X \colon E_X &\longrightarrow [0,1] \\ x &\longmapsto p_X(x) = P_X(\{x\}) = P(X = x) \end{aligned}$$

Así definida, es fácil deducir las siguientes propiedades:

- **Propiedad I.** No negativa $\Rightarrow p_X(x) \geq 0 \quad \forall x \in E_X$
- **Propiedad II.** La suma de sus valores es la unidad $\Rightarrow \sum_{x \in E_X} p_X(x) = 1$

Ya estamos en condiciones de especificar la expresión de la función de distribución ya que $P(X \leq x) = \sum_{k:x_k \leq x} P(X = x_k)$ y por ello:

$$F(x) = \sum_{k:x_k \leq x} p_X(x_k) \tag{2.5}$$

donde la forma del gráfico de $F(x)$ es constante en los intervalos delimitados por cada dos valores adyacentes que toma la variable aleatoria y en cada valor hay puntos de salto de tamaño p_k en cada punto x_k.

2.1.2 Vectores aleatorios

Podemos considerar un conjunto de variables aleatorias $X_1, X_2, \ldots, X_n$ definidas en un espacio de probabilidad $(\Omega, \mathcal{A}, P)$.

Definición 9 *El vector $X = (X_1, X_2, \ldots, X_n)$ se conoce como vector aleatorio o variable aleatoria n-dimensional. Y se puede definir su función de distribución conjunta*

$$F(x_1, x_2, \ldots, x_n) = P(X_1 \leq x_1, \ldots, X_n \leq x_n) \tag{2.6}$$

y siendo $p_X(x_1, x_2, \ldots, x_n)$ su función masa de probabilidad conjunta puede calcularse como:

$$F\left(x_1, \ldots, x_n\right) = \sum_{x \leq x_1} \cdots \sum_{x \leq x_n} p_X\left(x_1, \ldots, x_n\right) \tag{2.7}$$

Hemos introducido los vectores aleatorios, entre otras cosas, para poder definir el concepto de independencia de variables aleatorias.

Definición 10 *Diremos que $X_1, X_2, \ldots, X_n$ son variables aleatorias independientes si se verifica que*

$$F(x_1, x_2, \ldots, x_n) = F_1(x_1) \ldots F_n(x_n) \tag{2.8}$$

para todo $x_1, x_2, \ldots, x_m \in \mathbb{R}$

Para variables aleatorias discretas, podemos caracterizar el concepto mediante la función masa de probabilidad:

Proposición 1 *Las variables $X_1, X_2, \ldots, X_n$ son independientes si y sólo si se verifica, que siendo $x_{k_1}, x_{k_2}, \ldots$ los valores que toma cada variable aleatoria X_k*

$$p(x_{1m_1}, x_{2m_2}, \ldots, x_{nm_n}) = P(X_1 = x_{1m_1}, X_2 = x_{2m_2}, \ldots, X_n = x_{nm_n}) = p(x_{1m_1}) \ldots p(x_{nm_n})$$

para naturales $m_1, m_2, \ldots, m_n$ arbitrarios.

Definiremos a continuación dos conceptos que serán centrales en nuestro trabajo y que son indicadoras importantes del carácter de una variable aleatoria:

2.1.3 Esperanza matemática y Varianza

Comenzamos con la definición de esperanza matemática, una forma de representar el valor medio de una variable aleatoria.

Definición 11 *Sea X una variable aleatoria discreta, con función masa $p_X(x)$ y sea $g(x)$ una función continua. Entonces se define la esperanza de $g(X)$ como*

$$E[g(X)] = \sum_k g(x_k)p_X(x_k) \tag{2.9}$$

y existe si la serie $\sum_k |g(x_k)|p_X(x_k)$ es absolutamente convergente. En particular se define la esperanza matemática de X como

$$E[X] = \sum_k x_k p_X(x_k) \tag{2.10}$$

Si X toma una cantidad finita de valores, la esperanza matemática existe dado que hay un número finito de sumandos en la serie. En particular tendremos:

$$E[X] = \sum_{k=1}^{m} x_k p_X(x_k) \tag{2.11}$$

Nótese que si $p(x_1) = \ldots = p(x_m) = \frac{1}{m}$ se tiene que la esperanza matemática no es más que el promedio aritmético.

$$E[X] = \frac{1}{m}\sum_{k=1}^{m} x_k \tag{2.12}$$

Enunciamos algunas propiedades que utilizaremos:

- **Propiedad I.** Dada una variable aleatoria X con esperanza y dos números reales a y b, entonces $E[aX + b] = aE[x] + b$
- **Propiedad II.** Dadas dos variables aleatorias X e Y con esperanza, entonces $E[X + Y] = E[X] + E[Y]$
- **Propiedad III.** Dadas dos variables aleatorias independientes X e Y con esperanza, entonces $E[XY] = E[X]E[Y]$

Ahora consideraremos el concepto de varianza, que servirá como punto de partida para nuestro estudio de las diferentes variables asociadas a la epidemia, aunque no de una forma clásica.

Definición 12 *Sea X una variable aleatoria con esperanza $E[X]$. Podemos definir su varianza como la esperanza del cuadrado de las distancias de la variable X a su esperanza, es decir*

$$Var[X] = E[(X - a)^2] \tag{2.13}$$

Además, al valor positivo de la raíz de la varianza lo llamamos desviación estándar de la variable X, $\sigma_X = \sqrt{Var[X]}$

Utilizando la terminología de la definición de esperanza, podemos tomar $g(x_k) = (x_k - a)^2$ y escribir la varianza como

$$Var[X] = \sum_k (x_k - a)^2 p_X(x_k) \tag{2.14}$$

si la serie anterior es convergente, de nuevo, en el caso finito, lo será.

Para calcularla tenemos varias opciones, la sencillez de la siguiente identidad hace que se utilize bastante:

$$Var[X] = E[X^2 - 2E[X]X + E[X]^2] = E[X^2] - 2E[X]^2 + E[X]^2 = E[X^2] - E[X]^2 \tag{2.15}$$

Ahora damos algunas propiedades de la varianza que necesitaremos en nuestro desarrollo:

- **Propiedad I.** Dada una variable aleatoria X su varianza es no negativa. Además, la varianza es nula si, y sólo si P(X = c) = 1 para algún $c \in \mathbb{R}$.
- **Propiedad II.** Dada una variable aleatoria X con $Var[X]$ y dos constantes $a, b \in \mathbb{R}$, entonces $Var[aX + b] = a^2 Var[X]$.
- **Propiedad III.** Dadas dos variables aleatorias independientes X e Y con varianza, entonces $Var[X + Y] = Var[X] + Var[Y]$.

La esperanza matemática y la varianza son las dos principales características numéricas de una variable aleatorio, aunque se engloban en toda una familia conocida como momentos que no necesitaremos utilizar. La esperanza representa el centro o la posición representativa de la variable y la varianza caracteriza el grado de concentración o dispersión de los valores de la variable en torno a la esperanza.

Finalmente, estudiamos como se relacionan las variables aleatorias entre sí y varias formas de cuantificar esa relación.

2.1.4 Covarianza y correlación

En esta sección, de nuevo, consideraremos un espacio de probabilidad $(\Omega, \mathcal{A}, P)$ y dos variables aleatorias X e Y:

Definición 13 *Definimos la covarianza entre X e Y de la siguiente forma:*

$$Cov[X, Y] = E[(X - E[X])(Y - E[Y])] \tag{2.16}$$

que desarrollando la expresión facilita su cálculo mediante la identidad

$$Cov[X, Y] = E[XY] - E[X]E[Y] \tag{2.17}$$

Directamente relacionado con este concepto, pero con la ventaja de tomar valores acotados, tenemos el coeficiente de correlación:

Definición 14 *El coeficiente de correlación entre las variables aleatorias X e Y al que notaremos $\rho_(X,Y)$, se define como*

$$\rho(X,Y) = \frac{Cov[X,Y]}{\sqrt{Var[X]Var[Y]}} \tag{2.18}$$

Dadas 2 variables aleatorias independientes, como vimos, la esperanza del producto coincide con el producto de las esperanzas, por lo que $\rho(X,Y) = Cov[X,Y] = 0$ y las variables son incorreladas, el recíproco no es cierto en general.

Como dijimos, una gran ventaja es que toma valores en $[-1,1]$

Teorema 1 *Sean X, Y variables aleatorias. Entonces su coeficiente de correlación cumplen*

$$-1 \leq \rho(X,Y) \leq 1 \tag{2.19}$$

y la igualdad ocurre si, y sólo si Y es combinación lineal de X con probabilidad 1.

Cuando tenemos un conjunto de variables aleatorias $X_1, X_2, \ldots, X_n$ podemos considerar su covarianza que tiene forma de matriz:

Definición 15 *La matriz de covarianza de un vector aleatorio codifica las covarianzas entre todas las variables en una matriz $n \times n$ de la siguiente forma:*

$$\Sigma = \begin{bmatrix} Var[X_1] & Cov[X_1,X_2] & \cdots & Cov[X_1,X_n] \\ Cov[X_1,X_2] & Var[X_2] & \cdots & Cov[X_2,X_n] \\ \vdots & \vdots & \ddots & \vdots \\ Cov[X_1,X_n] & Cov[X_2,X_n] & \cdots & Var[X_n] \end{bmatrix} \tag{2.20}$$

Es sencillo ver que la matriz es simétrica, lo que sugiere considerar la forma cuadrática asociada, lo haremos en el siguiente capítulo.

2.2 Productos escalares y ortogonalidad

En la descripción del método se tratará de trasladar un problema en principio del ámbido de la teoría de probabilidad al ámbito de la geometría y del análisis. Para ello, necesitaremos conceptos ligados a los espacios vectoriales dotados de un producto escalar y a la noción de ortogonalidad. Para esta sección se ha utilizado como referencia [15].

Definición 16 *Sea V un espacio vectorial, un producto escalar en V es una aplicación:*

$$\langle \cdot, \cdot \rangle : V \times V \Longrightarrow \mathbb{R} \tag{2.21}$$

verificando las siguientes propiedades:

1. $\langle u, v \rangle = \langle v, u \rangle \qquad \forall u, v \in V$
2. $\langle u + v, w \rangle = \langle u, w \rangle + \langle v, w \rangle \qquad \forall u, v, w \in V$
3. $\langle au, v \rangle = a\langle u, v \rangle \qquad \forall u, v \in V \quad \forall a \in \mathbb{R}$
4. $\forall u \in V \quad \langle u, u \rangle \geq 0$ *y* $\langle u, u \rangle = 0 \iff u = 0$

Rápidamente, a partir de la definición se tienen estas 2 propiedades:

5. $\forall u \in V \quad \langle 0, u \rangle = \langle u, 0 \rangle = 0$
6. $\langle \sum_i a_i u_i, \sum_j b_j v_j \rangle = \sum_{i,j} a_i b_j \langle u_i, v_j \rangle$ *para cualesquiera* $a_i, b_j \in \mathbb{R}$ *y* $u_i, v_j \in V$

Esto induce la siguiente definición:

Definición 17 *Sea V un espacio vectorial con un producto escalar bien definido, entonces se define la norma de un vector como* $||u|| = \sqrt{\langle u, u \rangle}$

Es fácil probar las propiedades de la norma a partir de las del producto escalar. Conviene introducir junto a estas definiciones, las siguientes desigualdades clásicas de este tipo de espacios:

Proposición 2 *(Desigualdad de Schwartz) Sea V un espacio vectorial dotado de un producto escalar. Para cada $x, y \in V$ se verifica:*

$$|\langle x, y \rangle| \leq ||x||\,||y|| \tag{2.22}$$

Proposición 3 *(Desigualdad triangular de Minkowski) Sea V un espacio vectorial dotado de un producto escalar. Para cada $x, y \in V$ se verifica:*

$$||x + y|| \leq ||x|| + ||y|| \tag{2.23}$$

A raíz de la desigualdad de Schwartz, se verifica que

$$-1 \leq \frac{\langle x, y \rangle}{||x||\,||y||} \leq 1 \tag{2.24}$$

Lo que nos lleva a introducir el siguiente concepto que proviene de la definición clásica del producto escalar en $\mathbb{R}^2$:

Definición 18 *Llamaremos ángulo entre los vectores x e y al único* $\alpha \in \mathbb{R}$ *con* $\alpha \in [0, \pi]$ *de forma que:*

$$cos(\alpha) = \frac{\langle x, y \rangle}{||x||\,||y||} \tag{2.25}$$

En nuestro contexto, este ángulo tendrá un significado especial relacionado con la similitud entre las variables.

Definición 19 *Se dice que los vectores* $x, y \in V$ *son ortogonales y escribimos* $x \perp y$ *si* $\langle x, y \rangle = 0$

Aunque no vamos a necesitar de forma explícita trabajar con bases, si utilizaremos las matrices asociadas a las mismas, lo que nos lleva a introducir estos tipos especiales de bases.

Definición 20 *Se dice que una base de un espacio vectorial V es ortogonal si los vectores que la forman son ortogonales 2 a 2. Además, una base ortogonal es ortonormal si todos los vectores que la forman tienen norma unitaria.*

Definición 21 *Una matriz A es ortogonal si su inversa coincide con su traspuesta. Es decir si* $A^{-1} = A^T$. *Además, la matriz de cambio de base entre 2 bases ortonormales es ortogonal.*

Podemos establecer que en cualquier espacio vectorial dotado de un producto escalar existen bases ortonormales y además dar un proceso para construirlas.

Teorema 2 *(Proceso de Gram-Schmidt) Sea V un espacio vectorial con producto escalar y sea* $\mathcal{B} = \{u_1, \ldots, u_k\}$ *una base de vectores de V, entonces existe una base ortonormal* $\{e_1, \ldots, e_k\}$ *tal que para cada k se tiene que* $L(\{u_1, \ldots, u_k\}) = L(\{e_1, \ldots, e_k\})$

El procedimiento es el siguiente:

$e_1 = u_1$

$e_2 = u_2 - \frac{\langle u_2, e_1 \rangle}{||e_1||^2} e_1$

$e_n = u_n - \frac{\langle u_n, e_1 \rangle}{||e_1||^2} e_1 - \cdots - \frac{\langle u_n, e_{n-1} \rangle}{||e_{n-1}||^2} e_{n-1}$

Es decir, para $i = 1, \ldots, n$: $e_i = u_i - \lambda_{i,1} e_1 - \cdots - \lambda_{i,i-1} e_{i-1}$ con $\lambda_{i,j} = \frac{\langle u_i, e_j \rangle}{||e_j||^2}$

Una simple inducción prueba el resultado y con las propiedades de la definición de producto escalar se puede probar la ortogonalidad. Obtenemos el siguiente corolario que nos será de utilidad.

Corolario 1 *En un espacio vectorial dotado de un producto escalar, todo conjunto de vectores no nulos y ortogonales dos a dos $\{u_1, \ldots, u_r\}$ puede ampliarse a una base ortogonal de V usando el proceso de Gram-Schmidt.*

Finalmente necesitaremos utilizar conceptos de ortogonalidad y subespacios.

Definición 22 *Dado un vector $x \in V$ y un subespacio U de V se dice que x es ortogonal a U si es ortogonal a todos los vectores de U, es decir $x \perp U \iff x \perp y \quad \forall y \in U$. Equivalentemente se puede decir lo mismo si cumple esto último para un conjunto de generadores de U.*

Definición 23 *Dado un subespacio U de un espacio vectorial V. Entonces el conjunto*

$$U^{\perp} = \{x \in V | x \perp U\} \tag{2.26}$$

es un subespacio vectorial conocido como complemento ortogonal y $V = U \bigoplus U^{\perp}$

Finalizamos esta sección con un resultado clásico debido a Pitágoras:

Teorema 3 *(Teorema de Pitágoras) Sea V un espacio vectorial dotado de un producto escalar y sean $x, y \in V$ ortogonales, entonces*

$$||x + y||^2 = ||x||^2 + ||y||^2 \tag{2.27}$$

La demostración es sencilla escribiendo las normas con la definición 2.

2.3 Formas bilineales simétricas y formas cuadráticas

Comenzamos definiendo el concepto de forma bilineal, aunque lo necesitaremos sólo colateralmente. Nuestro interés se encuentra en las formas cuadráticas, ya que en nuestro estudio posterior definiremos una de estas. Continuamos basándonos en [15] y [16].

Definición 24 *Sean $\mathbb{K}$ un cuerpo y V un $\mathbb{K}$-espacio vectorial, una forma bilineal es una aplicación:*

$$f\colon V \times V \longrightarrow \mathbb{K}$$

verificando:

1. $f(u_1 + u_2, v) = f(u_1, v) + f(u_2, v)$
2. $f(u, v_1 + v_2) = f(u, v_1) + f(u, v_2)$
3. $f(au, v) = af(u, v)$
4. $f(u, av) = af(u, v)$

para cualesquiera $a \in \mathbb{K}$, $u, v, u_1, u_2, v_1, v_2 \in V$

Tiene propiedades muy similares a las dadas para el producto escalar, de hecho, es una generalización del mismo. Un producto escalar es una forma bilineal simétrica y definida positiva. Veamos que significa esto. Podemos asociar una matriz M a una forma bilineal f, dada una base de V como $M = (a_{ij}) = (f(e_i, e_j))$ obteniendo la ecuación matricial de f:

$$f(x,y) = x^T My = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \begin{bmatrix} f(e_1,e_1) & f(e_1,e_2) & \cdots & f(e_1,e_n) \\ f(e_2,e_1) & f(e_2,e_2) & \cdots & f(e_2,e_n) \\ \vdots & \vdots & \ddots & \vdots \\ f(e_n,e_1) & f(e_n,e_2) & \cdots & f(e_n,e_n) \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \tag{2.28}$$

Definición 25 *Una forma bilineal es simétrica si verifica $f(x, y) = f(y, x)$ para todo $x, y \in V$*

Definición 26 *Una forma bilineal es semidefinida positiva si verifica $f(x, x) \geq 0$ para todo $x \in V$.*

A partir de estos conceptos, podemos introducir las formas cuadráticas.

Definición 27 *Sea V un $\mathbb{K}$-espacio vectorial y sea $f\colon V \times V \longrightarrow \mathbb{K}$ una forma bilineal. Se llama forma cuadrática asociada a la forma bilineal f a la aplicación*

$$\Phi\colon V \longrightarrow \mathbb{K}$$

definida por

$$\Phi(x) = f(x, x)$$

De las que podemos deducir algunas propiedades a partir de las de las formas bilineales:

- **Propiedad I.** $\Phi(0) = 0$.
- **Propiedad II.** $\Phi(\lambda x) = \lambda^2 \Phi(x)$.
- **Propiedad III.** $\Phi(x + y) = \Phi(x) + \Phi(y) + f(x, y) + f(y, x)$.

Cada forma cuadrática está asociada a una única forma bilineal simétrica:

Proposición 4 *Dada una forma cuadrática Φ en V, existe una única forma bilineal simétrica f_p cuya forma simétrica asociada es Φ.*

Finalmente, como se tratará la optimización de formas cuadráticas, introducimos una notación para referirnos al $u \in V$ tal que una forma cuadrática Φ alcanza en él su valor máximo:

$$u = \arg\max_{v \in V} \Phi(v), \tag{2.29}$$

2.4 Diagonalización

La diagonalización es un proceso utilizado ampliamente en la implementación de métodos númericos y de cálculo para optimizar operaciones con matrices. Aunque en nuestro caso la utilizaremos de otra forma, en concreto para calcular el óptimo de una función. El siguiente desarrollo y algunos resultados sucesivos se toman de [15] y [16]. Para comenzar el desarrollo debemos definir la siguiente relación entre matrices:

Definición 28 *Dos matrices cuadradas A, B sobre un cuerpo son semejantes si existe una matriz regular P tal que $B = P^{-1}AP$*

Necesitamos algunos resultados sobre la traza de este tipo de matrices:

Proposición 5 *Si A y B son dos matrices cuadradas de tamaño n,entonces $tr(AB) = tr(BA)$*

Podemos escribir el producto por entradas de la siguiente forma:

$$(AB)_{ij} = \sum_{k=1}^{n} (A)_{ik}(B)_{kj} \tag{2.30}$$

entonces la traza se escribe como:

$$tr(AB) = \sum_{i=1}^{n} (AB)_{ii} = \sum_{i=1}^{n}\sum_{k=1}^{n} (A)_{ik}(B)_{ki} \tag{2.31}$$

y por la asociatividad del producto de matrices:

$$tr(AB) = \sum_{k=1}^{n}\sum_{i=1}^{n} (A)_{ik}(B)_{ki} = \sum_{k=1}^{n}\sum_{i=1}^{n} (B)_{ki}(A)_{ik} = \sum_{k=1}^{n} (BA)_{kk} = tr(BA) \tag{2.32}$$

Corolario 2 *La traza es un invariante entre matrices semejantes, es decir si A y B son semejantes, entonces $tr(A) = tr(B)$*

Con la proposición anterior y la definición de semejanza es casi trivial:

$$tr(B) = tr(P^{-1}AP) = tr((P^{-1}A)P) = tr(P(P^{-1}A)) = tr((PP^{-1})A) = tr(A) \tag{2.33}$$

El objetivo de la diagonalización de una matriz A es encontrar una matriz D diagonal (con entradas no nulas sólo en la diagonal principal) y una matriz de paso P que haga semejantes a D y A.

Definición 29 *Una matriz A es diagonalizable si es semejante a una matriz diagonal.*

La formulación habitual que se utiliza se escribe en términos de endomorfismos de un espacio vectorial de dimensión n. Esto es debido a que existe un isomorfismo entre las matrices cuadradas y dicho espacio de endomorfismos. Para nuestro objetivo, esto no tiene especial interés, por lo que nos limitamos a las definiciones de valores propios y vectores propios y a dar un criterio general de diagonalización.

Definición 30 *Sea A una matriz cuadrada de dimensión n y $v \in \mathbb{C}^n$ un vector no nulo cumpliendo que*

$$Av = \lambda v \tag{2.34}$$

para algún $\lambda \in \mathbb{C}$. Entonces λ se llama valor propio de la matriz A y v vector propio de A asociado a λ. Para cada λ valor propio de A se puede definir el subespacio propio asociado, que no es más que el conjunto

$$V_\lambda = \{u \in \mathbb{C}^n | Au = \lambda u\}$$

Definimos la multiplicidad geométrica para cada λ como la dimensión de su subespacio propio asociado.

De esta definición surge naturalmente la siguiente, que no proviene mas que de tratar de encontrar los valores λ que son valores propios.

Definición 31 *El polinomio característico de una matriz A cuadrada de tamaño n se define a partir de la definición de valor propio de la siguiente forma:*

$$det(A - \lambda I_n) = 0$$

Además, tiene grado n y sus raíces son los valores propios de la matriz A. Se define la multiplicidad algebraica de cada λ como el número de veces que aparece como raíz de la ecuación.

Finalmente, vamos a enunciar un criterio general de diagonalización que nos bastará para nuestros propósitos:

Teorema 4 *Sea $A \in \mathbb{R}^{n\times n}$ una matriz cuadrada y sean $\lambda_1, \lambda_1, \cdots, \lambda_r$ sus distintos valores propios. Entonces A es diagonalizable si, y sólo si se verifican las siguientes condiciones:*

1. *La suma de las multiplicidades algebraicas de los valores propios es n.*
2. *Para cada λ_i las multiplicidades algebraica y geométrica coinciden.*

3 Descripción del método

Vamos a modelar los datos como un conjunto de variables aleatorias $X_1, ..., X_n$ en el que cada una de ellas representa una variable de interés de la pandemia a lo largo de observaciones diarias. Nuestro objetivo es extraer una nueva variable aleatoria combinación lineal de las anteriores de modo que recoja la máxima información y variabilidad de todas las demás. Es decir una variable resumen X_r.

$$X_r = a_1X_1 + a_2X_2 + ... + a_nX_n$$

En primer lugar, vamos a estandarizar cada variable, esto es una técnica muy habitual para el análisis de datos pues conseguimos hacer comparables datos que tienen diferentes escalas de medida. Obtenemos variables normalizadas, independientes de la escala y con la misma media y varianza. Es decir, pasamos a tener un conjunto de variables $X_1^N, X_2^N, ..., X_N^N$ que proceden de restarles su media muestral a las originales y dividirlas entre su desviación típica:

$$X_i^N = \frac{X_i - \bar{X}_i}{\sqrt{Var[X_i]}}$$

De este modo tenemos que $E[X_i^N] = 0$ y $Var[X_i^N] = 1$. Vamos a comenzar a estudiar la variabilidad de los datos, en concreto como se relacionan entre sí las variables, para ello definamos su matriz de covarianza. Normalmente la covarianza entre un par de variables se define como $Cov[X, Y] = E[(X - E[X])(Y - E[Y])]$ pero en nuestro caso como las variables estan centradas se reduce a la $E[XY]$.

De hecho si consideramos las variables aleatorias centradas como vectores de un espacio vectorial, la covarianza es un producto escalar válido. Las variables centradas forman un espacio vectorial considerando 0 del espacio a la variable aleatoria que vale 0 con probabilidad 1. Además, las combinaciones lineales de variables con media 0 también tienen media 0. El resto de propiedades se deducen fácilmente.

Es muy importante considerar este enfoque en el que reconocemos el espacio vectorial generado por las variables aleatorias centradas. Esto nos va a permitir alejarnos del campo de la estadística y poder formular nuestro problema en términos de análisis y geometría.

Cuando se comparan 2 vectores en un espacio vectorial, se suele utilizar como medida de similaridad el coseno del ángulo que forman. Cuando vale 1, son colineales, cuando vale 0 ortogonales y cuando vale -1 son colineales pero con direcciones opuestas. Este valor coincide con el coeficiente de correlación que

hemos definido en los preliminares de este trabajo y es igual al cociente de la covarianza entre las desviaciones típicas de las variables, en nuestro caso, como hemos estandarizado las variables, coincide con la covarianza:

$$\rho_{X,Y} = \frac{Cov[X,Y]}{\sqrt{Var[X]Var[Y]}} = E[XY] \tag{3.1}$$

En nuestro caso de interés, como las variables estan centradas, todas las covarianzas coinciden con la esperanza del producto de cada par de variables.

Consideremos de nuevo nuestra variable resumen objetivo X_r, como hemos centrado las variables sabemos por linealidad de la esperanza que

$$E[X_r] = E[a_1X_1^N + a_2X_2^N + \cdots + a_nX_n^N] = a_1E[X_1^N] + a_2E[X_2^N] + \cdots + +a_nE[X_n^N] = 0 \tag{3.2}$$

Luego sin más remedio nuestro objeto central de estudio debe ser la varianza de X_r que ya adelantamos que deseamos que describa en su conjunto la mayor cantidad posible de información del resto de variables. Vamos a relacionar la varianza de X_r con los conceptos que hemos introducido en los preliminares de este trabajo:

Proposición 6 *Sea X_r una variable aleatoria combinación lineal de $X_1^N, X_2^N ..., X_n^N$ y sea $v = (a_1, a_2, ..., a_n) \in \mathbb{R}^N$ el vector de coeficientes de dicha combinación lineal. Consideremos también la matriz de covarianza Σ_{X_r} del conjunto de variables $X_1^N, X_2^N ..., X_n^N$. Entonces:*

$$Var[X_r] = v^T \Sigma_{X_r} v \tag{3.3}$$

Desarrollemos la expresión de la varianza:

$$\begin{aligned} Var[X_r] = E[X_r^2] &= E[(a_1X_1^N + a_2X_2^N + \cdots + a_nX_n^N)^2] \\ &= a_1^2E[X_1^{N2}] + a_2^2E[X_2^{N2}] + \cdots + a_n^2E[X_n^{N2}] + \sum_{\substack{i,j=1 \\ i \neq j}}^{n} a_ia_jE[X_i^NX_j^N] \\ &= \sum_{i,j=1}^{n} a_ia_jE[X_i^NX_j^N] \end{aligned} \tag{3.4}$$

Y veamos que coincide con la otra expresión:

$$
\begin{aligned}
v^T\Sigma_{X_r}v &= \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix} \begin{bmatrix} Var[X_1^N] & Cov[X_1^N, X_2^N] & \cdots & Cov[X_1^N, X_n^N] \\ Cov[X_1^N, X_2^N] & Var[X_2^N] & \cdots & Cov[X_2^N, X_n^N] \\ \vdots & \vdots & \ddots & \vdots \\ Cov[X_1^N, X_n^N] & Cov[X_2^N, X_n^N] & \cdots & Var[X_n^N] \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} \\
&= \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix} \begin{bmatrix} E[X_1^{N^2}] & E[X_1^N X_2^N] & \cdots & E[X_1^N X_n^N] \\ E[X_1^N X_2^N] & E[X_2^{N^2}] & \cdots & E[X_2^N X_n^N] \\ \vdots & \vdots & \ddots & \vdots \\ E[X_1^N X_n^N] & E[X_2^N X_n^N] & \cdots & E[X_n^{N^2}] \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} \\
&= \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix} \begin{bmatrix} a_1E[X_1^{N^2}] + a_2E[X_1^N X_2^N] + \cdots + a_nE[X_1^N X_n^N] \\ a_1E[X_1^N X_2^N] + a_2E[X_2^{N^2}] + \cdots + a_nE[X_2^N X_n^N] \\ \vdots \\ a_1E[X_1^N X_n^N] + a_2E[X_2^N X_n^N] + \cdots + a_nE[X_n^{N^2}] \end{bmatrix} \\
&= a_1^2E[X_1^{N^2}] + a_1a_2E[X_1^N X_2^N] + \cdots + a_1a_nE[X_1^N X_n^N] \\
&+ a_1a_2E[X_1^N X_2^N] + a_2^2E[X_2^{N^2}] + \cdots + a_2a_nE[X_2^N X_n^N] \\
&+ \dots \\
&+ a_1a_nE[X_1^N X_n^N] + a_2a_nE[X_2^N X_n^N] + \cdots + a_n^2E[X_n^{N^2}] \\
&= \sum_{i,j=1}^{n} a_ia_jE[X_i^N X_j^N]
\end{aligned}
\tag{3.5}
$$

Corolario 3 *La varianza de las distintas combinaciones lineales* $X_r = a_1X_1^N + a_2X_2^N + ... + a_nX_n^N$ *define una forma cuadrática en* $\mathbb{R}^n$

$$
\begin{aligned}
\mathcal{V}\colon \mathbb{R}^n &\longrightarrow \mathbb{R} \\
v &\longmapsto v^T\Sigma_{X_r}v = \langle \Sigma_{X_r}v, v \rangle
\end{aligned}
$$

Se deduce de la simetría de Σ_{X_r} y de la correspondencia biyectiva existente entre las formas cuadráticas y las matrices simétricas.

Nuestro objetivo en el próximo capítulo será encontrar los coeficientes de la combinación lineal que maximizen la varianza de la variable resumen. Para ello deberemos imponer alguna condición sobre el dominio, ya que tal y como lo hemos definido, la forma cuadrática no está acotada.

Proposición 7 *La forma cuadrática $\mathcal{V}$ es semidefinida positiva y no está acotada superiormente.*

Recordemos la relación con la varianza de X_r y el hecho de que toda variable aleatoria tiene varianza no negativa.

$$\mathcal{V}(v) = Var[X_r] \geq 0 \qquad \forall v \in \mathbb{R}^n \setminus 0 \tag{3.6}$$

No podemos afirmar que sea definida positiva, porque puede ocurrir que la variable X_r sea degenerada. Suponiendo que no fuera degenerada, si aumentamos los coeficientes de la combinación lineal, aumenta la varianza sin límite, es decir:

$$\lim_{||x|| \to \infty} \mathcal{V}(x) = +\infty \tag{3.7}$$

Admitiendo el caso degenerado, se puede probar que existen direcciones en las que ocurre lo mismo y la varianza crece sin límite.

4 Optimización de la forma cuadrática $\mathcal{V}$

El siguiente problema radica en encontrar la combinación lineal que maximice la varianza, lo cual se puede abordar formulando un problema de maximización. Sin embargo la forma cuadrática actual no está acotada. Para garantizar que este máximo exista, necesitamos restringir la forma cuadrática. Es razonable restringirla a un compacto para garantizar la existencia del máximo. Para ello deberemos obtener un teorema de existencia de valores extremos que podremos derivar de resultados clásicos de topología sobre compactos.

La restricción a la esfera unidad parece ser la mejor opción. Además de ser esencial por su compacidad para garantizar que la forma cuadrática tenga un máximo bien definido, tiene un significado geométrico importante, ya que limita las direcciones de las combinaciones lineales a puntos en la superficie de la esfera, es decir, sigue existiendo cierta libertad de elección en todas las direccciones del espacio.

Deduzcamos un teorema de valores extremos para aplicarlo a nuestro caso, para ello nos ayudaremos del Teorema de Heine-Borel (se toma la demostración de [11]) y de un lema básico sobre de aplicaciones continuas:

Teorema 5 *(Teorema de Heine-Borel) Un subconjunto K de $\mathbb{R}^n$ es compacto si, y sólo si, es cerrado y acotado.*

Sea K un compacto de $\mathbb{R}^n$. Como $\mathbb{R}^n$ es un espacio Hausdorff, K es cerrado. Consideramos el siguiente recubrimiento por abiertos de K: $\{B(0,m) \cap K \mid m \in \mathbb{N}\}$. Como K es compacto existe un subrecubrimiento finito. Entonces existe un $M \in \mathbb{N}$ tal que coincide con el radio máximo de las bolas del subrecubrimiento, luego $K \subset B(0,M)$ y K es acotado. Recíprocamente si $K \subset \mathbb{R}^n$ es cerrado y acotado. Existe un $N \in \mathbb{N}$ suficientemente grande como para que el producto de intervalos $[-N,N]^n$ que es un compacto, contenga a K. Y un cerrado contenido en un compacto también lo es.

Lema 1 *La compacidad se conserva por aplicaciones continuas.*

Estamos en condiciones de probar el siguiente teorema de valores extremos:

Teorema 6 *Sea $\mathcal{V}\colon \mathbb{S}^{n-1} \longrightarrow \mathbb{R}$ una función real continua definida en el compacto $\mathbb{S}^{n-1}$. Entonces $\mathcal{V}$ alcanza un valor máximo.*

Es decir, existe un $a \in \mathbb{S}^{n-1}$ tal que $\mathcal{V}(a) \geq \mathcal{V}(x) \qquad \forall x \in \mathbb{S}^{n-1}$

Como $\mathbb{S}^{n-1}$ es compacto, la imagen $\mathcal{V}(\mathbb{S}^{n-1})$ es un compacto de $\mathbb{R}$, por el teorema de Heine-Borel, es cerrado y acotado. Por ser cerrado, el supremo α de la imagen $\mathcal{V}(\mathbb{S}^{n-1})$ que es un punto adherente pertenece al conjunto $\alpha \in Im\mathcal{V}$. Entonces existe $a \in \mathbb{S}^{n-1}$ tal que $\mathcal{V}(a) = \alpha$ siendo este el máximo.

Con este teorema aplicado en la esfera unitaria a nuestra forma cuadrática asociada a la varianza de X_r garantizamos la existencia de la variable objetivo que maximiza la varianza y por tanto recoge la máxima cantidad de información del resto de variables. Probada la existencia, ahora necesitamos obtener una forma de calcularlo, para ello acudiremos a la teoría de Sylvester. Comenzamos con algunos lemas asociados a los valores y subespacios propios de una matriz simétrica cuya idea es la misma y se han tomado de [16].

Lema 2 *Si λ_1, λ_2 son valores propios distintos de una matriz simétrica A entonces sus subespacios propios asociados son ortogonales.*

Sean v_1, v_2 vectores de los subespacios propios asociados a λ_1 y λ_2. Calcularemos el producto escalar $v_1 \cdot Av_2$ de 2 formas distintas gracias a la simetría de A:

$$v_1 \cdot Av_2 = v_1 \cdot (\lambda_2 v_2) = \lambda_2 (v_1 \cdot v_2) \tag{4.1}$$

$$v_1 \cdot Av_2 = v_1^T Av_2 = (A^T v_1)^T v_2 = (Av_1)^T v_2 = \lambda_1 (v_1 \cdot v_2) \tag{4.2}$$

Pero como los valores propios son distintos, para que se de la igualdad los vectores deben ser ortogonales.

Lema 3 *Todos los valores propios de una matriz real simétrica son reales.*

Sean $p \pm iq$ una pareja de valores propios conjugados de A con vectores propios correspondientes $v \pm iw$ y calculemos el siguiente producto, también de 2 formas:

$$\begin{aligned} (v+iw)^T A(v-iw) &= (v+iw)(p-iq)(v-iw) \\ &= (p-iq)(v+iw)(v-iw) \\ &= (p-iq)(||v||^2 + ||w||^2) \end{aligned} \tag{4.3}$$

$$\begin{aligned} (v+iw)^T A(v-iw) &= (A^T(v+iw))^T (v-iw) \\ &= (A(v+iw))^T (v-iw) \\ &= (p+iq)(v+iw)^T (v+iw) \\ &= (p+iq)(||v||^2 + ||w||^2) \end{aligned} \tag{4.4}$$

Claramento eso implica que $q = 0$ y por tanto los valores propios son reales.

Ahora comprenderemos porque este interés en los valores propios, veremos que el máximo de la forma cuadrática se alcanza en un vector propio y su valor es justamente el mayor valor propio. Este último lo denominaremos varianza explicada por la variable resumen X_r. Esta demostración se ha tomado de [4], al igual que la idea de la construcción de el método de esta forma en lugar que de la manera habitual con optimización de Lagrange que se puede consultar en [8].

Proposición 8 *Dada una matriz simétrica $A \in \mathbb{R}^{n\times n}$ existe un vector $u_1 \in \mathbb{S}^{n-1}$ tal que maximiza la forma cuadrática $\mathcal{V}(x) = x^T Ax$ en la esfera unitaria. Además u_1 es un vector propio, asociado al valor propio real $\lambda_1 = u_1^T Au_1$, es decir:*

$$u_1 = \arg \max_{\|x\|_2=1} x^T Ax \tag{4.5}$$

$$\lambda_1 = \max_{\|x\|_2=1} x^T Ax \tag{4.6}$$

Sea $u_1 \in \mathbb{S}^{n-1}$ el vector en el que se alcanza el máximo que nos brinda el teorema 6. Descompongamos Au_1 en 2 componentes ortogonales, la primera en la dirección del vector u_1 y la otra en el plano tangente a la esfera, de la siguiente forma

$$Au_1 = u_1^T Au_1 u_1 + x_\perp \tag{4.7}$$

$$x_\perp = Au_1 - u_1^T Au_1 u_1 \tag{4.8}$$

Queremos probar que como en u_1 se alcanza el máximo, entonces $x_\perp = 0$ en cuyo caso

$$Au_1 = u_1^T Au_1 u_1 = \lambda_1 u_1 \tag{4.9}$$

Lo haremos por reducción al absurdo, supongamos que $x_\perp \neq 0$ y definamos el siguiente vector para algún $\epsilon > 0$ a escoger más tarde.

$$y = u_1 + \epsilon x_\perp \tag{4.10}$$

Como $x_\perp \perp u_1$ podemos aplicar el teorema de Pitágoras:

$$\begin{aligned} ||y||_2^2 &= ||u_1||_2^2 + \epsilon^2 ||x_\perp||_2^2 \\ &= 1 + \epsilon^2 ||x_\perp||_2^2 \end{aligned} \tag{4.11}$$

Entonces si normalizamos el vector y, tenemos un vector de la esfera unitaria en el que se alcanza un valor mayor que en u_1

$$\begin{aligned}\left(\frac{y}{\|y\|_2}\right)^T A\frac{y}{\|y\|_2} &= \frac{y^TAy}{\|y\|_2^2}\\ &= \frac{u_1^TAu_1 + 2\epsilon x_\perp^TAu_1 + \epsilon^2 x_\perp^TAx_\perp}{1+\epsilon^2\|x_\perp\|_2^2}\\ &= \frac{u_1^TAu_1 + 2\epsilon\|x_\perp\|_2^2 + \epsilon^2 x_\perp^TAx_\perp}{1+\epsilon^2\|x_\perp\|_2^2}\\ &= u_1^TAu_1 + \frac{2\epsilon\|x_\perp\|_2^2 + \epsilon^2\left(x_\perp^TAx_\perp - \|x_\perp\|_2^2 u_1^TAu_1\right)}{1+\epsilon^2\|x_\perp\|_2^2}\\ &> u_1^TAu_1\end{aligned} \tag{4.12}$$

siempre que escojamos un ϵ tal que

$$\epsilon < \frac{2\|x_\perp\|_2^2}{\left|x_\perp^TAx_\perp\right| + \|x_\perp\|_2^2\left|u_1^TAu_1\right|} \tag{4.13}$$

Lo que entra en contradicción con que u_1 sea un máximo para la forma cuadrática $\mathcal{V}$. Por lo que $x_\perp = 0$ y por tanto el máximo se alcanza en u_1 que es un vector propio asociado al valor propio $\lambda_1 = u_1^TAu_1$.

Ya hemos probado que el máximo se alcanza en un vector propio, ahora veamos que toda matriz asociada a una forma cuadrática diagonaliza y por tanto cada cada valor propio es el máximo de la forma cuadrática en su respectivo subespacio ortogonal. Esta demostración se ha tomado de [3]

Proposición 9 *Toda matriz simétrica $A \in \mathbb{R}^{n\times n}$ es diagonalizable por congruencias. Es decir existen una matriz diagonal de valores propios $D \in \mathbb{R}^{n\times n}$ y una matriz ortogonal $O \in \mathbb{R}^{n\times n}$ tal que*

$$A = O^TDO \tag{4.14}$$

Utilizaremos el primer principio de inducción finita sobre el tamaño de la matriz A. El caso base es sencillo, ya que para tamaño uno, toda matriz es simétrica y de hecho diagonal. Supongamos como hipótesis de inducción que toda matriz simétrica de tamaño $n-1$ es diagonalizable por congruencias y tratemos de probarlo para tamaño n:

Comenzamos seleccionando un valor propio $\lambda \in \mathbb{R}$ y su correspondiente vector propio v_1 normalizado. Ahora completaremos v_1 a una base ortonormal de $\mathbb{R}^n$, sabemos que esto puede hacerse para todo espacio vectorial euclídeo, por ejemplo, mediante el proceso de Gram-Schmidt. Colocamos estos vectores por columnas en una matriz P que es por tanto ortogonal. Para simplificar la notación llamemos V a la matriz $n \times (n-1)$ que hemos concatenado a v_1 para

definir P. Entonces:

$$P^TAP = \begin{bmatrix} v_1^T \\ V^T \end{bmatrix} A \begin{bmatrix} v_1 & V \end{bmatrix} = \begin{bmatrix} v_1^TAv_1 & v_1^TAV \\ V^TAv_1 & V^TAV \end{bmatrix} = \begin{bmatrix} \lambda_1 v_1^Tv_1 & \lambda_1 v_1^TV \\ \lambda_1 V^Tv_1 & V^TAV \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & V^TAV \end{bmatrix} \tag{4.15}$$

Donde V_TAV es una matriz simétrica $(n-1)\times(n-1)$ que por hipótesis de inducción es diagonalizable ortogonalmente, es decir, existe $\Delta \in \mathbb{R}^{(n-1)\times(n-1)}$ diagonal y $Q \in \mathbb{R}^{(n-1)\times(n-1)}$ ortogonal tal que:

$$\Delta = Q^T(V^TAV)Q \tag{4.16}$$

Concluimos que la matriz ortogonal $O = [v_1 \quad VQ]$ diagonaliza ortogonalmente A:

$$O^TAO = \begin{bmatrix} v_1^T \\ Q^TV^T \end{bmatrix} A\,[v_1VQ] = \begin{bmatrix} \lambda_1 & 0 \\ 0 & Q^TV^TAVQ \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \Delta \end{bmatrix} \tag{4.17}$$

Aunque nuestro interés finaliza con la consecución del máximo, por completitud vamos a analizar el comportamiento de la forma cuadrática en es subespacio ortogonal al generado por el vector u_1. El razonamiento de la demostración de la proposición anterior nos brinda una base ortogonal del subespacio ortogonal al generado por v_1, con ello basta aplicar reiteradamente la proposición 7 teniendo en cuenta que todo vector se puede normalizar para llevarlo a la esfera unitaria.

Para finalizar, de la semipositividad de la forma cuadrática se puede concluir lo mismo de sus valores propios:

Proposición 10 *Si una forma cuadrática es semidefinida positiva, entonces los valores propios λ_i de su matriz asociada son no negativos.*

La demostración es sencilla, basta considerar la siguiente cadena de desigualdades para cada valor propio λ_i y su vector propio asociado normalizado x_i:

$$0 \leq \mathcal{V}(x_i) = x_i^TAx_i = x_i^T\lambda_i x_i = \lambda_i||x_i||^2 = \lambda_i \tag{4.18}$$

Podemos resumir todo nuestro trabajo en el siguiente teorema:

Teorema 7 *(Teorema espectral de matrices simétricas y maximización de formas cuadráticas) Toda matriz simétrica $A \in \mathbb{R}^{n\times n}$ que define la forma cuadrática $\mathcal{V}(x) = x^TAx$ tiene una diagonalización por congruencias de la siguiente forma:*

$$A = \begin{bmatrix} v_1 & v_2 & \cdots & v_n \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \tag{4.19}$$

con valores propios reales $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$ y vectores propios reales, unitarios y ortogonales $v_1, v_2 \cdots v_n$. Además, optimizan la forma cuadrática en los subespacios:

$$\begin{aligned}
\lambda_1 &= \max_{\|x\|_2=1} x^T A x, \\
v_1 &= \arg \max_{\|x\|_2=1} x^T A x, \\
\lambda_k &= \max_{\|x\|_2=1, x \perp v_1, \ldots, v_{k-1}} x^T A x, \quad 2 \leq k \leq n-1, \\
v_k &= \arg \max_{\|x\|_2=1, x \perp v_1, \ldots, v_{k-1}} x^T A x, \quad 2 \leq k \leq n-1, \\
\lambda_n &= \max_{\|x\|_2=1, x \perp v_1, \ldots, v_{k-1}} x^T A x, \\
v_n &= \arg \max_{\|x\|_2=1, x \perp v_1, \ldots, v_{k-1}} x^T A x.
\end{aligned} \tag{4.20}$$

4.1 Propiedades y casos particulares

Estudiamos ahora algunas propiedades de los valores propios que optimizan la forma cuadrática, así como algunos casos concretos que trataremos de caracterizar. Estos resultados nos llevarán a detectar el significado del valor propio principal:

Comenzamos estudiando la relación entre los valores propios y la traza de la matriz de covarianzas.

Proposición 11 *La traza de la matriz de covarianzas de las variables estandarizadas coincide con el número de variables de partida. Además coincide con la suma de los valores propios de la matriz.*

$$tr(\Sigma_{\tilde{X}}) = n = \sum_{i=1}^{n} \lambda_i \tag{4.21}$$

Como las variables están normalizadas, su varianza es uno, además, la traza es un invariante para matrices semejantes (o congruentes), esto se probó en el corolario 2 de la sección 2.4 de preliminares:

$$tr(\Sigma_{\tilde{X}}) = \sum_{i=1}^{n} Var[X_i^N] = n = \sum_{i=1}^{n} \lambda_i \tag{4.22}$$

La consecuencia inmediata es que como los valores propios son no negativos, han de sumar N y existe un valor propio máximo, al menos este debe ser mayor o igual que uno.

Veamos ahora algunos casos particulares de valores propios:

Proposición 12 *Si $\lambda_1 = 1$ entonces la matriz de covarianzas es la identidad $\Sigma = I_n$ y todas las variables de partida son incorreladas. Además la forma cuadrática $\mathcal{V}$ es constantemente 1.*

Si el valor propio $\lambda_1 = 1$ y hemos demostrado que es mayor o igual que todos los demás y la suma de los valores propios ha de ser n por la proposición anterior, tenemos que todos los valores propios son $\lambda_i = 1$ y la diagonalización de la matriz de covarianzas que sabemos que existe es la identidad. Por último, como la diagonal de la matriz de covarianzas Σ de las variables estandarizadas ya estaba formada por unos, debe ser que $\Sigma = I_n$ y por tanto:

$$Cov[X_i, X_j] = 0 \qquad \forall X_i, X_j \text{con } i \neq j \tag{4.23}$$

lo que significa que las variables son mutuamente incorreladas.

Es trivial ver que la forma cuadrática es constante:

$$\mathcal{V}(x) = x^T \Sigma x = x^T x = ||x||^2 = 1 \qquad \forall x \in \mathbb{S}^{n-1} \tag{4.24}$$

Finalmente, vamos a ver otro interesante resultado que proviene de la teoría de Perron sobre matrices positivas, para lo cuál necesitamos introducir algunos conceptos:

Definición 32 *Una matriz cuadrada $A \in \mathbb{R}^{n \times n}$ es positiva cuando todas sus entradas son positivas, es decir, $a_{ij} > 0$ y lo denotamos $A \gg 0$.*

Y también conceptos relacionados con el llamado espectro de una matriz u operador:

Definición 33 *El conjunto de valores propios de una matriz A se conoce como espectro de A y se denota $\sigma(A)$. Al valor propio de mayor módulo se le conoce como radio espectral y se le denota $\rho(A)$. También se le conoce como valor propio dominante.*

$$\rho(A) = \max_{\lambda_i \in \sigma(A)} \{|\lambda_i|\} \tag{4.25}$$

Este tipo de matrices son muy comunes en gran variedad de aplicaciones, tanto en física como en teoría de grafos, etc... Esto motiva el estudio de sus propiedades que llevo a cabo en profundidad el matemático alemán Oskar Perron durante el siglo XX. Para nuestro estudio, bastarán algunos resultados básicos sobre el espectro de las matrices positivas. En concreto, el llamado teorema de Perron-Frobenius cuya extensa demostración se omite por alejarse de los objetivos de este escrito y puede consultarse en [17]. En concreto, es suficiente con el siguiente enunciado de dicho teorema

Teorema 8 *Sea $\Sigma \in \mathbb{R}^d$ una matriz positiva $\Sigma \gg 0$. Entonces Σ tiene un valor propio λ_1 que es positivo y dominante. Además, el vector propio asociado v_1 tiene todas sus entradas positivas, es decir, $v \gg 0$.*

Este resultado conlleva la siguiente definición:

Definición 34 *El valor propio que cumple las condiciones del teorema anterior se conoce como valor propio de Perron.*

Aplicamos el teorema a nuestro estudio, teniendo en cuenta que 2 variables están correladas si su covarianza es positiva, obtenemos el siguiente resultado.

Corolario 4 *Si para las variables aleatorias $X_1, X_2, \ldots, X_n$ se tiene que*

$$\Sigma = \{Cov[X_i, X_j]\}_{i,j\in 1,\ldots,N} \gg 0 \tag{4.26}$$

entonces la matriz de covarianzas es positiva (todos los pares de variables están correladas) y λ_1 es un valor propio de Perron. Además, su vector propio asociado es positivo y por tanto, todos los pesos que generan la variable resumen $X_r = a_1 X_1^N + \ldots + a_n X_n^N$ son positivos. En particular, la variable resumen X_r correla positivamente con X_i para todo $i = 1, 2, \ldots, n$

Todo lo recién enunciado es consecuencia directa del Teorema de Perron, conviene justificar el hecho de que la correlación de X_r sea positiva con todas las variables de partida. En efecto para cada variable X_j con $j = 1, 2, \ldots, N$:

$$Cov[X_r, X_j] = Cov[\sum_i a_i X_i, X_j] = \sum_i a_i Cov[X_i, X_j] > 0 \tag{4.27}$$

ya que $v = (a_1, a_2, \ldots, a_N)$ es un vector asociado al valor propio de Perron y las covarianzas entre las variables son todas positivas por hipótesis.

Veamos por último que ocurriría si en esta situación, el valor propio de Perron es n y por tanto, la variable resumen recoge toda la variabilidad del conjunto de variables $X_1, X_2, \ldots, X_n$.

Proposición 13 *Sea $\Sigma = \{Cov[X_i, X_j]\}_{i,j\in 1,\ldots,N} \gg 0$ y supongamos que $\lambda_1 = N$ es un valor propio de Σ. Entonces la matriz Σ tiene unos en todas sus entradas y por tanto, las variables $X_i = X_j$ en el espacio de probabilidad.*

Para probarlo, comenzamos obteniendo por la propiedad de la traza que el resto de valores propios son 0. Además, el teorema de Perron Frobenius nos asegura que el vector propio asociado a λ_1 tiene todas sus entradas positivas $v \gg 0$.

Definamos la matriz $C = \Sigma - Nvv^T$ y amplíamos $\{v\}$ a una base de vectores propios de $\mathbb{R}^n$ añadiéndole los vectores propios asociados al valor propio 0.

Comprobemos que $C = 0$ para ello la multiplicamos con cada vector de la base de vectores propios de $\mathbb{R}^n$, comenzamos con v y luego estudiamos el caso del resto conjuntamente:

$$Cv = \Sigma v - Nvv^T v = Nv - Nv = 0 \tag{4.28}$$

$$Cv_i = \Sigma v_i - Nvv^T v_i = 0v_i - Nv\langle v, v_i\rangle = 0 \text{ para i} =, 2, ..., N \tag{4.29}$$

Luego, como C es nula sobre una base, $C = 0$ y $\Sigma = Nvv^T$, fijémonos en los elementos de la diagonal:

$$\Sigma = Nvv^T = N \begin{bmatrix} v_1^2 & & & \\ & v_2^2 & & \\ & & \ddots & \\ & & & v_n^2 \end{bmatrix} \tag{4.30}$$

Como las variables están normalizadas, conocemos que la diagonal de Σ tiene unos debido a las varianzas unitarias y despejando obtenemos que

$$v = \begin{bmatrix} \frac{1}{\sqrt{N}} \\ \vdots \\ \vdots \\ \frac{1}{\sqrt{N}} \end{bmatrix} \tag{4.31}$$

donde hemos usado que el vector propio v es positivo por el teorema de Perron y, por tanto, debemos utilizar la raíz cuadrada postiva. Ya podemos despejar la expresión final de Σ:

$$\Sigma = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ 1 & 1 & \ddots & 1 \\ 1 & 1 & \cdots & 1 \end{bmatrix} \tag{4.32}$$

Por tanto, el coeficiente de correlación lineal es 1

$$Cov[X_i, X_j] = 1 = \sqrt{Var(X_i)Var(X_j)} \tag{4.33}$$

y si calculamos la recta de regresión minimo cuadráticas de cada par de variables que tiene la siguiente expresión para dos variables aleatorias X e Y:

$$Y - E[Y] = \frac{Cov[X, Y]}{Var[X]}(X - E[X]) \tag{4.34}$$

observamos que como las variables están centradas $X_i = X_j$ para $i, j \leq n$.

Observamos que en este caso, no tendría sentido aplicar el método a variables idénticas. Para terminar veamos que tampoco tiene sentido aplicar el método a variables independientes.

Proposición 14 *Sean $X_1, X_2, \ldots, X_n$ un conjunto de variables aleatorias independientes 2 a 2. Entonces las variables son incorreladas y la matriz de covarianzas es la identidad.*

Es fácil ver que por independencia, las esperanzas son multiplicativas y como las variables están centradas, tenemos que $E[X_i^N X_j^N] = E[X_i^N]E[X_j^N] = 0$ y por tanto la matriz de covarianzas es la identidad. Por la proposición 12, el valor propio que maximiza la varianza también es unitario.

Después de realizar el desarrollo teórico de esta variable X_r que maximiza la varianza y comprobar en esta sección algunos casos particulares, es natural preguntarse cuanta información se conserva de las variables de partida (conjunto de datos original).

El primer y más representativo indicador es el valor propio más grande.

Definición 35 *El valor propio λ_1 que es el máximo valor que toma la forma cuadrática $\mathcal{V}$ en la esfera unitaria se denomina varianza explicada. Representa el número de variables del conjunto de partida que X_r consigue representar.*

Podemos entonces definir el porcentaje de varianza explicada mediante un cociente:

Definición 36 *El porcentaje de varianza explicada por la variable resumen viene dado por el cociente entre el mayor valor propio λ_1 y el número de variables de partida. Representa el porcentaje de variables del conjunto original que es capaz de resumir.*

$$\%VarianzaExp = \frac{Var[X_r]}{\sum_{i=1}^{n} Var[X_i^N]} \cdot 100 = \frac{\lambda_1}{n} \cdot 100 \tag{4.35}$$

5 Análisis de componentes principales

Recordemos el objetivo principal del capítulo anterior, maximizar la varianza de la variable resumen X_r. Todo el desarrollo teórico nos permite afirmar que por el teorema espectral, el máximo valor de dicha varianza se alcanza en un vector propio de la matriz de covarianzas, la dirección de máxima varianza. Además su valor coincide con el mayor valor propio de dicha matriz. Podemos reescribir el último teorema adaptándolo al contexto:

Teorema 9 *Sea $\tilde{X} = (X_1, \cdots, X_n)$ un vector aleatorio con matriz de covarianzas $\Sigma_{\tilde{X}}$ y un vector de coeficientes $u = (a_1, \cdots, a_n)$ tal que podemos escribir la variable resumen como*

$$X_r = a_1 X_1^N + a_2 X_2^N + ... + a_n X_n^N$$

Consideremos también λ_1 y v_1 el máximo valor propio de $\Sigma_{\tilde{X}}$ y su vector propio asociado normalizado. Entonces:

$$\begin{aligned} \lambda_1 &= \max_{\|u\|_2=1} \operatorname{Var}\left(\langle u, \tilde{X}\rangle\right), \\ v_1 &= \arg \max_{\|u\|_2=1} \operatorname{Var}\left(\langle u, \tilde{X}\rangle\right), \end{aligned} \tag{5.1}$$

Definición 37 *Llamamos direcciones principales de un conjunto de datos a las direcciones asociadas a los vectores propios de la matriz de covarianzas. Las componentes de un vector aleatorio en cada dirección principal definen las componentes principales de dicho vector.*

$$\widetilde{PC}_i := u_i^T \tilde{X}, \quad 1 \leq i \leq n \tag{5.2}$$

La variable X_r buscada no es más que la primera componente principal, aquella que maximiza la varianza y que resume la máxima información posible del conjunto de variables originales. Una propiedad fundamental de esta componente en relación con las demás es la siguiente.

Proposición 15 *Las componentes principales son incorreladas*

$$\begin{aligned} E[\widetilde{PC}_i \widetilde{PC}_j] &= E[u_i^T \tilde{X} u_j^T \tilde{X}] \\ &= u_i^T E[\tilde{X}^2] u_j \\ &= u_i^T \Sigma_{\tilde{X}} u_j \\ &= u_i^T \lambda_j u_j \\ &= \lambda_j u_i^T u_j = 0 \end{aligned} \tag{5.3}$$

La última igualdad proviene de la ortogonalidad de vectores propios distintos.

Para finalizar, podemos resumir todo nuestro desarrollo de forma simple algorítmicamente.

Algoritmo 1 *(Análisis de componentes principales) Dado un conjunto de datos de tamaño m con n variables* $X_1, \cdots, X_n$ *donde* $m > n$

1. *Estandarizamos cada variable para que tengan media 0 y varianza 1.*
2. *Calculamos la matriz de covarianzas de la muestra* $\Sigma_{X_1,\cdots,X_n}$.
3. *Diagonalizamos la matriz para calcular los coeficientes que definen cada componente.*
4. *Calculamos las componentes como es producto de los coeficientes por las variables.*

5.1 Comparación con las variables originales

Habíamos estudiado en la sección que se describe el método que la variable X_r cumple que

$$E[X_r] = E[a_1 X_1^N + a_2 X_2^N + \cdots + a_n X_n^N] = 0 \tag{5.4}$$

pero como hemos probado en el apartado de optimización, no conserva varianza unitaria como el conjunto de variables de partida. De hecho, nuestro objetivo hasta ahora ha sido maximizar esta varianza restringiendo los pesos a la esfera unitaria.

Si terminado el proceso queremos comparar alguna variable de partida con la variable X_r debemos de dotar a X_r de la media y la varianza de la variable a comparar. Para ello, podemos realizar 2 simples pasos.

En primer lugar, normalizamos la componente principal X_r dividiendo por su desviación típica:

$$X_r^N = \frac{X_r}{\sqrt{Var[X_r]}} \tag{5.5}$$

A continuación la dotamos de la media y varianza de la variable X_i con la que queremos comparar, para ello multiplicamos por su desviación típica y le sumamos su media:

$$X_r^{X_i} = \sqrt{Var[X_i]} X_r^N + E[X_i] \tag{5.6}$$

y como vimos en los preliminares se comprueba que obtenemos las siguientes media y varianza:

$$E[X_r^{X_i}] = \sqrt{Var[X_i]} E[X_r^N] + E[X_i] = E[X_i]$$

$$Var[X_r^{X_i}] = Var[X_i] Var[X_r^N] = Var[X_i]$$

De esta forma tenemos un escalado que nos permite comparar las tendencias del resto de variables con la de la componente principal.

6 Búsqueda y limpieza de datos

Una vez hemos construido el modelo, el siguiente paso es recopilar datos para aplicarlo. Se ha decidido utilizar datos de nuestro país, España. El motivo fundamental es la cercanía y el conocimiento previo del contexto y la situación del país durante el desarrollo de las distintas olas. A la hora de la limpieza y del análisis posterior, tendremos ventaja al conocerlo ya que podremos encontrar posibles motivos de los problemas o de los resultados que se obtengan.

Para la obtención de los mismos se ha investigado en distintas fuentes, ya sea el Instituto Nacional de Estadística o las distintas administraciones públicas. Aunque la disponibilidad y facilidad de exportación de los datos de salud es muy buena, no era suficiente ya que el objetivo de nuestro estudio es realizar el análisis sobre variables mucho más generales, ya sean sociales, económicas, climáticas, de actuaciones del gobierno, etc... Finalmente se descubrió la plataforma COVID-19 Open Data Repository de Google Health [29]. COVID-19 Open Data proporciona una amplia colección de conjuntos de datos actualizados sobre la pandemia de COVID-19 en todo el mundo, en concreto también para España. Esta fuente de datos contiene información precisa y confiable sobre el número de casos, pruebas realizadas, hospitalizaciones y otras métricas sobre movilidad, clima o socioeconómicas que amplian el ámbito del análisis como se deseaba. Además de su amplitud, los datos de Open COVID-19 Data estan bien estructurados, lo que facilita su procesamiento y análisis. Pudimos extraer fácilmente las series temporales de las variables deseadas.

No se contemplaban algunos datos que nos resultaban importantes para el estudio como el número de tests realizados y los datos de vacunación, se han tomado de otro dataset abierto de la organización Our World In Data [14]. Para integrarlos, como estaban en formato csv, se han podido integrar de forma sencilla con funciones de la librería Pandas [19] y su clase por excelencia, el DataFrame.

Comenzamos exponiendo los distintos tipos de variables que se van a contemplar en el análisis:

- Salud: Datos básicos sobre el número diario de infectados, fallecidos, hospitalizados o ingresados en la unidad de cuidados intensivos. Tambien incluye divisiones de las mismas por grupos de edad y por género. Finalmente, datos del número de test realizados diariamente y a partir de enero de 2021 también datos de vacunación.
- Sociales y sobre medidas gubernamentales: Datos sobre niveles de las medidas de contención impuestas por el gobierno. Niveles de cuarentena, restricciones de transporte, cancelación de eventos, cierre de escuelas, etc...
- Movilidad: Datos recopilados por Google [29] mediante densidad de dispo-

sitivos en lugares de trabajo, lugares de ocio, parques, comercios, estaciones de transporte y zonas residenciales.

Se muestran las variables junto a su descripción en la siguiente tabla:

Variables	**Descripción**
new confirmed	Nuevos infectados detectados
new deceased	Nuevos fallecidos
new hospitalized patients	Nuevos hospitalizados
new intensive care patients	Nuevos ingresados UCI
new tested	Número de test realizados
new persons vaccinated	Nuevos vacunados
new persons fully vaccinated	Nuevas personas con pauta completa
new confirmed age 0-8	Nuevos infectados por grupos de edad 0-8 (9)
new deceased age 0-8	Nuevos fallecidos por grupos de edad 0-8 (9)
new hospitalized patients age 0-8	Nuevos hospitalizados por grupos de edad 0-8 (9)
new intensive care patients age 0-8	Nuevos ingresados UCI por grupos de edad 0-8 (9)
new confirmed male	Nuevos infectados hombres
new confirmed female	Nuevas infectadas mujeres
new deceased male	Nuevos fallecidos hombres
new deceased female	Nuevas fallecidas mujeres
new hospitalized patients male	Nuevos hospitalizados hombres
new hospitalized patients female	Nuevas hospitalizadas mujeres
new intensive care patients male	Nuevos ingresados UCI hombres
new intensive care patients female	Nuevas ingresadas UCI mujeres
school closing	Grado de cierre de escuelas en el país
workplace closing	Grado de cierre de lugares de trabajo en el país
cancel public events	Grado de cancelación de eventos públicos en el país
restrictions on gatherings	Grado de restricciones en reuniones
public transport closing	Grado de cierre de transporte público
stay at home requirements	Grado de confinamiento o cuarentena
restrictions on internal movement	Grado de restricciones en viajes en el país
international travel controls	Grado de control en viajes internacionales
income support	Nivel de ayuda económica general del gobierno
debt relief	Nivel de ayudas respecto a alquileres y rentas
testing policy	Nivel de política de test de infectados
contact tracing	Nivel de trazado de contactos de infectados
investment in vaccines	Grado de inversión nacional en vacunas
facial coverings	Grado de requerimiento de uso de mascarillas
vaccination policy	Grado de política de vacunación
stringency index	Métrica combinada de variables de respuesta social
retail and recreation percent change	Tasa de cambio de movilidad de ocio
grocery and pharmacy percent change	Tasa de cambio de movilidad de comercio y farmacias
parks percent change	Tasa de cambio de movilidad en parques públicos
transit stations percent change	Tasa de cambio de movilidad en estaciones de transporte
workplaces percent change	Tasa de cambio de movilidad en lugares de trabajo
residential percent change	Tasa de cambio de movilidad en zonas residenciales

Trás llevar a cabo la integración de los datos en un DataFrame de Pandas y una limpieza de nulos (sustitución por ceros, ya que se achaca a la ausencia de medición) un breve analisis exploratorio, nos muestra que las variables de salud tienen este tipo de ruido:

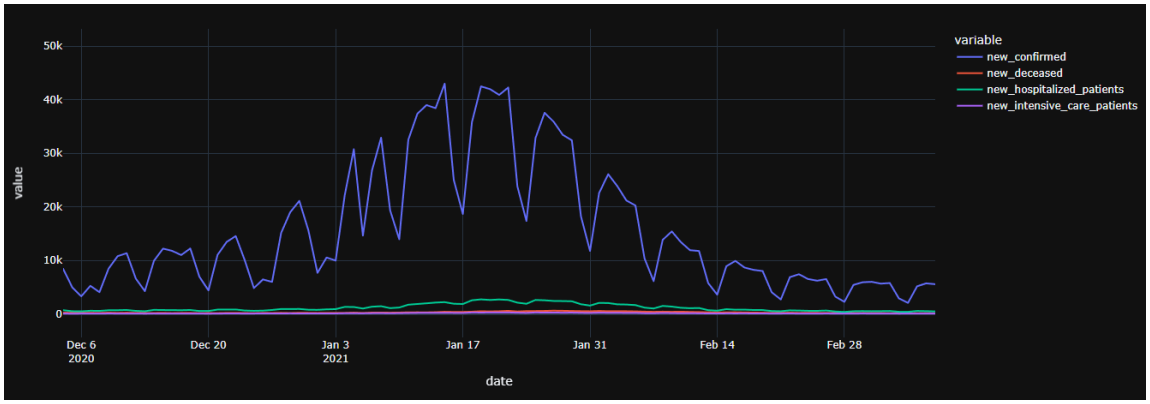

Figura 1: Ruido de medición en la variable Nuevos Confirmados

El motivo es un sesgo en la toma de datos, los fines de semana en la mayoría de zonas no se actualizaban los datos y a posteriori se agregaban al dato de los lunes. De esta forma tenemos esta periodicidad que hace que las tendencias esten distorsionadas. Una aplicación del método en estas condiciones hacía que la primera componente principal asumiera esta periodicidad ya que esta forma de medición pasa a formar parte del fenómeno.

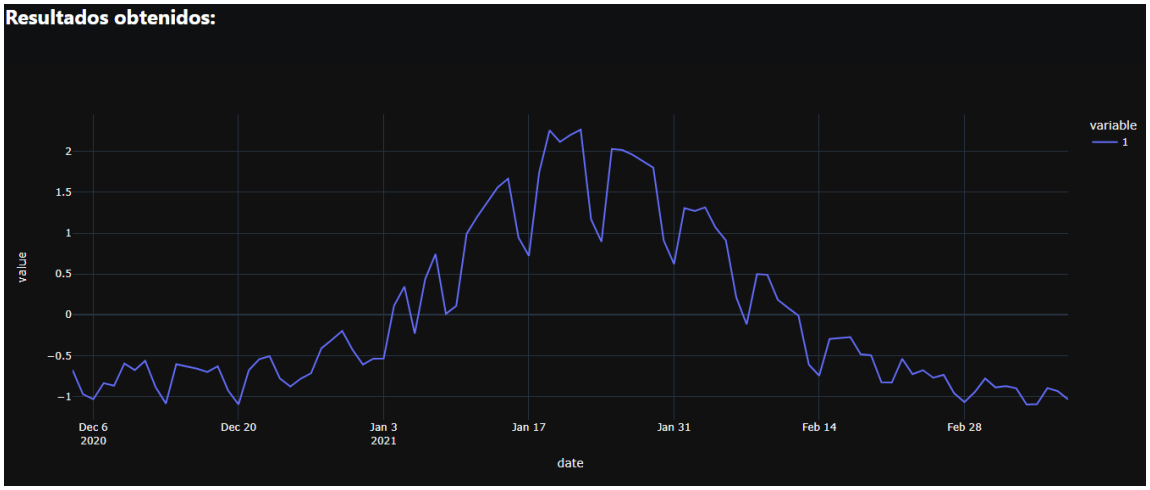

Figura 2: Ruido de medición transmitido a la variable resumen X_r

Podríamos asumir esta periodicidad, pero sabemos que los resultados están sesgados por la forma de tomar los datos y no son congruentes con la realidad de la epidemia en España, por tanto estudiaremos 3 formas distintas de solucionar este problema suavizando los datos sin hacer que varíen sus propiedades en exceso.

6.1 Promedio entre viernes y lunes

La primera idea que surge es tratar de suavizar utilizando la información que tenemos de que el ruido se produce semanalmente compensándonse los sábados y domingos inframuestreados, sobremuestreando los lunes. La idea es promediar los valores de sábados, domingos y lunes cada semana y repartir los datos en esos 3 días. Sin embargo, aunque el ruido que queda no es tan abrupto, no se termina de eliminar, en este caso tenemos pequeños valles semanalmente en los fines de semana y los lunes por el reparto que se ha hecho.

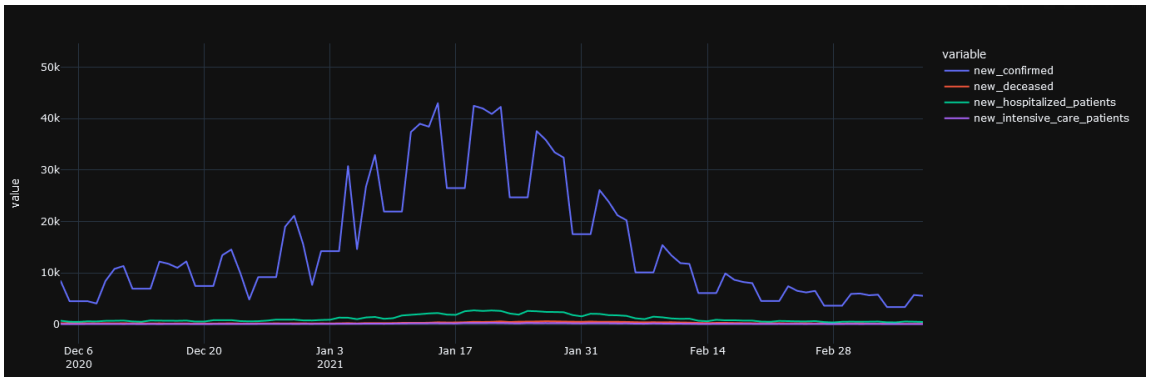

Figura 3: Ruido difuminado en valles.

Debemos buscar otras alternativas que tengan resultados más potentes, ya que de nuevo estos valles se transmiten a la variable resumen:

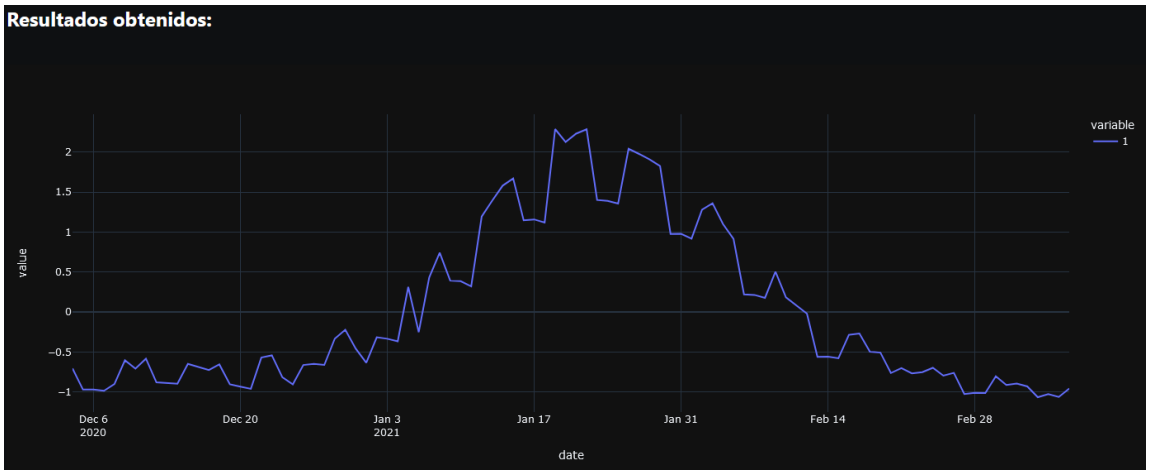

Figura 4: Ruido en valles transmitido a la variable resumen X_r

6.2 Agrupación semanal

La solución más clara sería agrupar los datos de cada variable sumándolos en grupos de N días, la cuestión es cómo elegir ese N para mejorar la calidad de los datos (sin perder o difuminar demasiado la información) y los resultados del método de Análisis de Componentes.

Podemos realizar el siguiente experimento, agrupar en periodos de n días y ver la varianza $Var[X_r]$ resultante. Se observa en la siguiente gráfica que el resultado es prácticamente monótono creciente, aumenta la varianza explicada con el número de días.

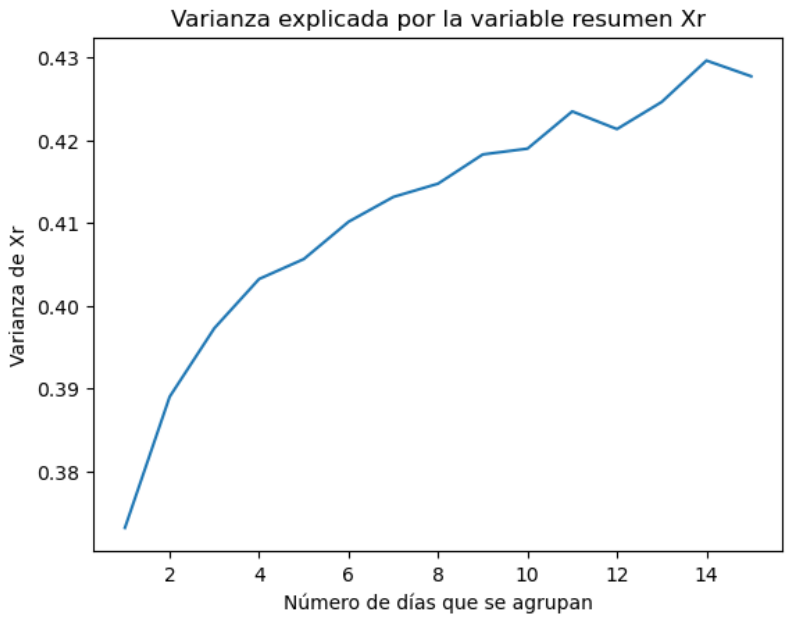

Figura 5: Valor de $Var[X_r]$ al aplicar el método en grupos de N días

Entendemos que esto es intuitivo, cuantos menos datos haya, mejor los va a poder representar el método en general. Sin embargo, cuanto mayor sea el N que tomemos, más estamos difuminando la información, de este modo, en lugar de tener una variable significativa que resuma el fenómeno de forma diaria, la obtenemos en un rango temporal cada vez más grande.

Observemos que en la figura anterior, el aumento significativo de la varianza explicada ocurre en el intervalo $[1,7]$ donde aumenta del 37 % al 41.5 % , a continuación solo aumenta al 42.8 % a costa de agrupar en grupos muy grandes. Esto unido a nuestro conocimiento de la naturaleza del ruido y de su periodicidad semanal nos hace escoger $N = 7$ y tomar grupos semanales.
Esto es algo que en epidemiología es habitual, calcular tasas tanto a 7 días como a 14 días. Como curiosidad, esto se suele hacer en cada población o región geográfica calculando una tasa por cada 100000 habitantes que hace a los datos

comparables entre regiones. Nuestro análisis será global para todos los datos de España y no usaremos estas tasas.
Esta agrupación evidentemente suaviza los datos como queríamos como se puede ver en la siguiente figura, aunque a costa sacrificar que cada dato sea una semana y no un día.

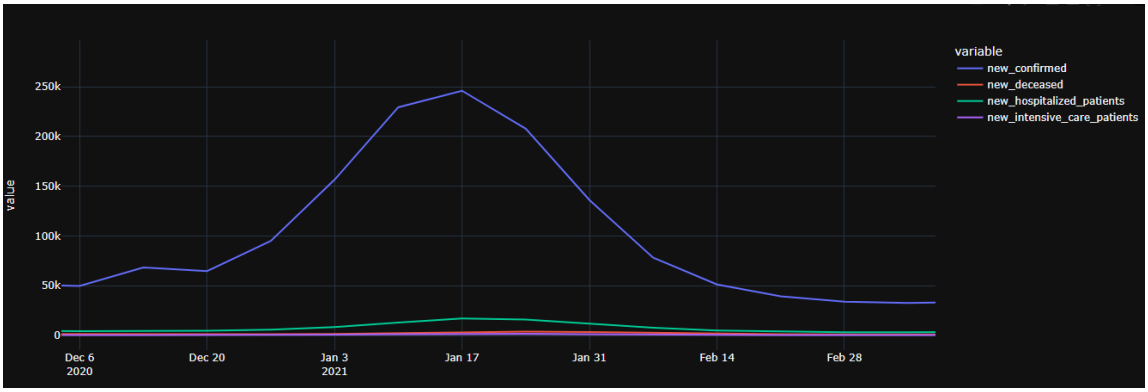

Figura 6: Suavizado de los datos por agrupación semanal

6.3 Media móvil o Seasonal Trend Decomposition

Aunque la opción anterior ya resuelve el problema y es la que se va a implementar, podemos explorar otras opciones avanzadas que son capaces de extraer una tendencia diaria suave de los datos. Lo hacen a costa de manipular ligeramente la realidad de los datos, aunque realmente esto ocurre en todo proceso de limpieza mientras que se mantenga un porcentaje muy alto de su naturaleza.

- Media móvil semanal: En cada punto, se toman datos 6 días hacia atrás y el correspondiente al propio día y se calcula la media. Esto suaviza los datos extrayendo la tendencia a largo plazo y eliminando la estacionalidad.
- Seasonal Trend Decomposition: Técnica que descompone una serie temporal en tres componentes: la tendencia, la estacionalidad y los residuos. Utiliza un enfoque iterativo para estimar y eliminar la componente estacional y los residuos, dejando únicamente la tendencia. Utiliza la técnica de regresión local, un método mixto de regresión no lineal que utiliza normalmente una función de pesos cúbica. Este complejo método se describe en [2].

7 Análisis de resultados

Los datos que vamos a analizar sobre el estado de la pandemia en España se corresponden con el periodo en el que se estuvo tomando datos de forma diaria. En concreto, la comunidad científica considera la existencia de las siguientes 6 olas [1] de la pandemia en España atendiendo a su distribución temporal.

- Primera ola: desde los primeros casos detectados a principios de marzo de 2020 hasta el 31 de junio de 2020. Está marcada por el infratesteo (no se disponía de pruebas adecuadas), la tardía entrada en vigor de las medidas de contención y la alta carga viral. Todo esto provocó un alto número de fallecidos y el colapso del sistema sanitario.

- Segunda ola: desde el 1 de julio de 2020 hasta el 30 de noviembre de 2020. En este caso, es complejo determinar cuando finaliza la segunda ola y comienza la tercera. Sus características principales son un aumento en la capacidad de test, y por ello, del número de positivos y también una respuesta más rápida con medidas de distanciamiento social e higiene respiratoria bastante estrictas.

- Tercera ola: desde el 1 de diciembre de 2020 hasta el 31 de marzo de 2021. En esta ola comienza la campaña de vacunación en los grupos más vulnerables a partir del mes de enero.

- Cuarta ola: desde el 1 de abril de 2021 hasta el 14 de junio de 2021. Es la ola con los números más bajos de contagios y fallecidos, probablemente por el adecuado progreso de la campaña de vacunación.

- Quinta ola: desde el 15 de junio de 2021 hasta el 30 de noviembre de 2021. En esta ola repuntan los casos con respecto a la cuarta, pese a que la campaña de vacunación está casi completada. Esto se asocia al surgimiento de la cepa Ómicron [28], más contagiosa pero menos letal y sospechosa de ser algo resistente a las vacunas.

- Sexta ola: desde el 1 de diciembre de 2021 hasta el 31 de marzo de 2022. Esta ola está marcada por la cepa Ómicron y por la reducción de las medidas de contención y el descenso de la conciencia social. Todo esto provocó los mayores datos de contagio de toda la pandemia y también volvió a llenar los recintos hospitalarios.

7.1 Varianza explicada por el método

A continuación, se muestra una tabla con las estadísticas de varianza explicada por la variable resumen X_r en las diferentes olas, de forma diaria o semanal.

Rango temporal	**Tipo de análisis**	**Varianza explicada por X_r**	**% Varianza Explicada**
1ª ola	Diario	43.569	59.6836
	Semanal	49.856	68.2959
2ª ola	Diario	36.862	50.4959
	Semanal	44.658	61.1753
3ª ola	Diario	35.097	48.0781
	Semanal	44.985	61.6233
4ª ola	Diario	30.806	42.2000
	Semanal	39.959	54.7384
5ª ola	Diario	31.559	43.2315
	Semanal	39.991	54.7822
6ª ola	Diario	37.903	51.9219
	Semanal	46.991	64.3712
Global	Diario	27.282	37.3726
	Semanal	30.428	41.6822

Observamos que como ya habíamos adelantado en la sección sexta, la varianza explicada siempre es mayor cuando se aplica el método semanalmente. Por otro lado, en la aplicación general a todas las olas, la varianza explicada alcanza un 40 por ciento del total de las variables, bastante menos que cuando se aplica a olas concretas. Esto es razonable, ya que la varianza del conjunto es más difícil de maximizar cuando en las distintas variables se calculo en base a distintas olas (incluso distintas cepas del virus) y aparecen nuevos fenómenos como la vacunación o los cambios en las restricciones.

En el resto de olas, se obtiene en torno a un 42 % y un 60 % de varianza explicada en el caso diario y entre un 54 % y un 68 % en la aplicación semanal. Lo que corresponde a explicar un alto número de variables, ya que se incluyen numerosas variables menos correladas como niveles de restricciones o tasas de movilidad. La forma de las curvas resultantes que se estudiara comparativamente a continuación, sugiere que en casi todos los casos (en el global se requiere discusión propia) se logra extraer la tendencia general de los datos de salud de la pandemia.

Vamos a realizar algunos análisis concretos para ciertas olas, en especial, la primera y la última, aunque son extensibles al resto de ellas. También realizaremos un análisis global, aunque ya hemos adelantado que la calidad disminuye. Realizaremos comparaciones con algunas de las variables originales y trataremos de extraer algunas conclusiones, aunque recordemos que el objetivo principal de este trabajo es de limpieza de datos y no tanto de análisis o predicción.

7.2 Análisis por olas

Comenzamos aplicando el método a la primera ola, con el siguiente resultado:

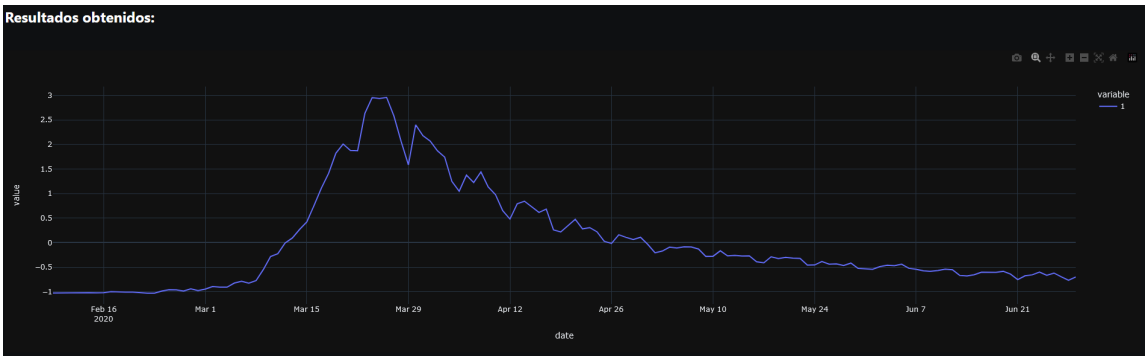

Figura 7: Variable resumen extraída en la primera ola.

Podemos comparar con la tendencia de nuevos infectados mediante el método descrito al final de la sección quinta.

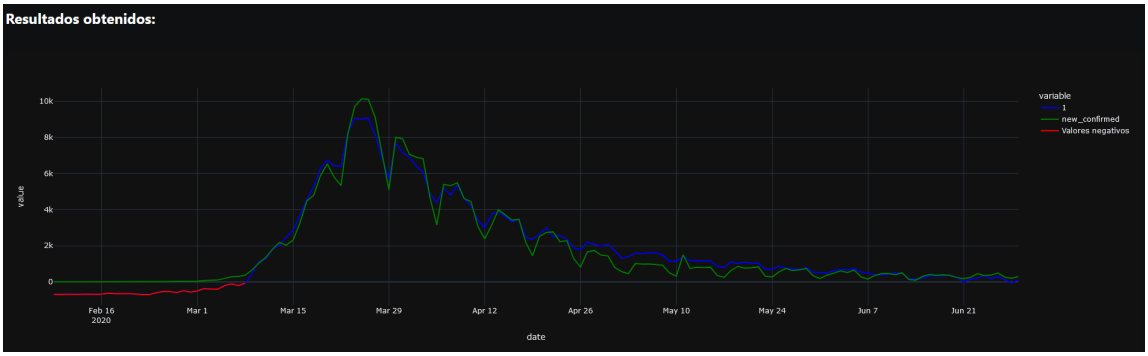

Figura 8: Variable resumen comparada con los datos de nuevos confirmados.

Podemos observar que a pesar de tratarse de un resumen de numerosas variables, la tendencia es muy similar a la de los nuevos infectados. Esto nos hace intuir que la componente principal tiene una forma muy similar a la curva de infectados del modelo epidemiológico SIR que introduciremos al final de esta sección.

Si comparamos con la variable que contiene los datos de los nuevos fallecidos, observamos un hecho interesante.

En este caso, la tendencia es muy similar pero con un desfase de 6 o 7 días entre los datos. Podría promediarse esta distancia temporal, para calcular una estimación del número de días que pasan desde que se detectan los contagios hasta que se alcanza la fase crítica [13]. Podemos repetir este análisis en la última ola de la pandemia, donde se supone predominante la variante Ómicron, mucho menos letal, aunque más contagiosa.

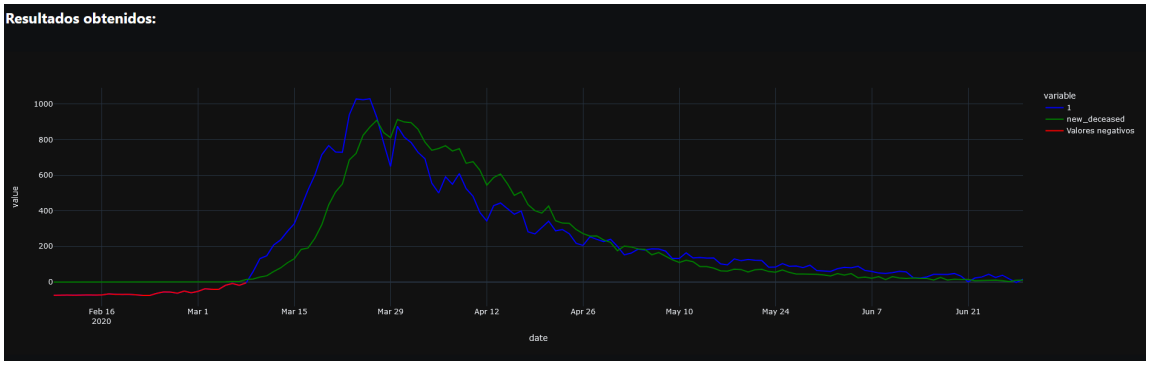

Figura 9: Variable resumen en la 1ª ola comparada con los datos de nuevos fallecidos.

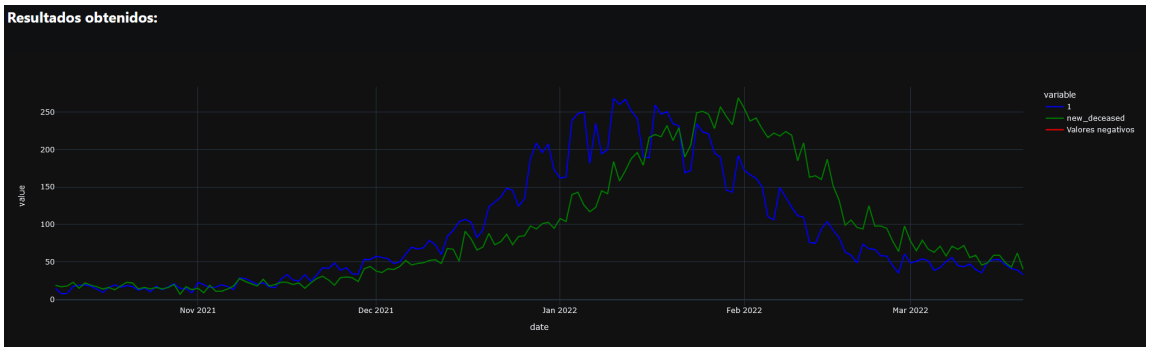

Figura 10: Variable resumen en la 6ª ola comparada con los datos de nuevos fallecidos.

El resultado es significativo, el desfase que aparece es mayor, ya que en este caso se puede interpretar que no se debe tanto al desarrollo de casos muy graves desde su infección, sino a que el número de muertes aumenta más tarde que en otras olas respecto al estado de la pandemia que indica la variable resumen. Intuitivamente, como la cepa es menos letal, necesita que la gravedad de la situación aumente durante más tiempo y se disparen los contagios, porque los fallecidos aparecen mayoritariamente en los grupos más vulnerables.

Podemos dar más certeza a este resultado si comparamos ambas olas con el número de fallecidos del grupo de edad de 50 años.

En la primera ola, las tendencias, aunque están desfasadas son mayoritariamente similares a la primera componente principal. Por el contrario en la última ola, además del desfase, las tendencias son mucho menos similares, teniendo oscilaciones mucho más fuertes y diferentes a las propias de la componente principal. Esto indica que que la primera ola afectó de forma más directa y grave a este grupo de edad y en la última lo hizo como consecuencia del altísimo número de contagios.

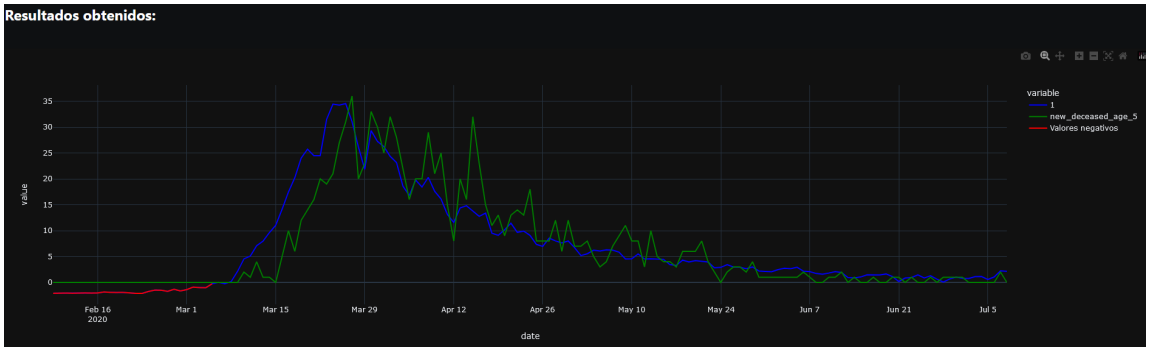

Figura 11: Variable resumen en la 1ª ola comparada con los nuevos fallecidos del grupo 50-60.

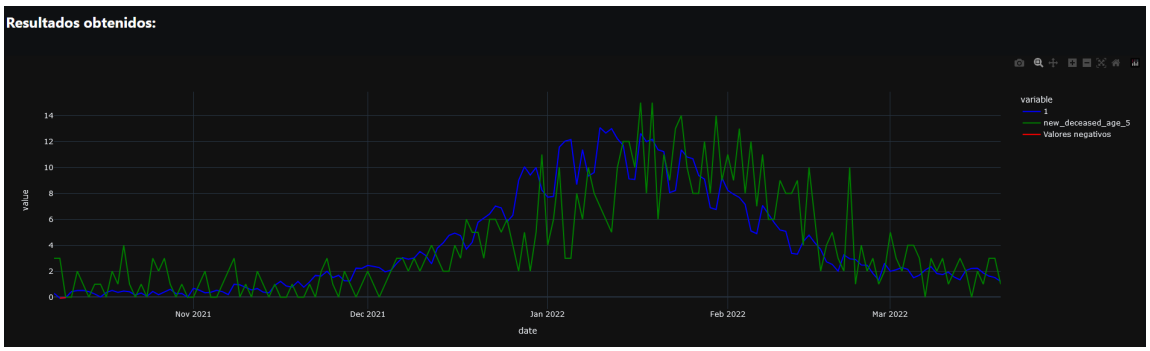

Figura 12: Variable resumen en la 6ª ola comparada con los nuevos fallecidos del grupo 50-60.

7.3 Análisis global

Comenzamos mostrando la variable resumen resultante de aplicar el método desde febrero de 2020 hasta marzo de 2022.

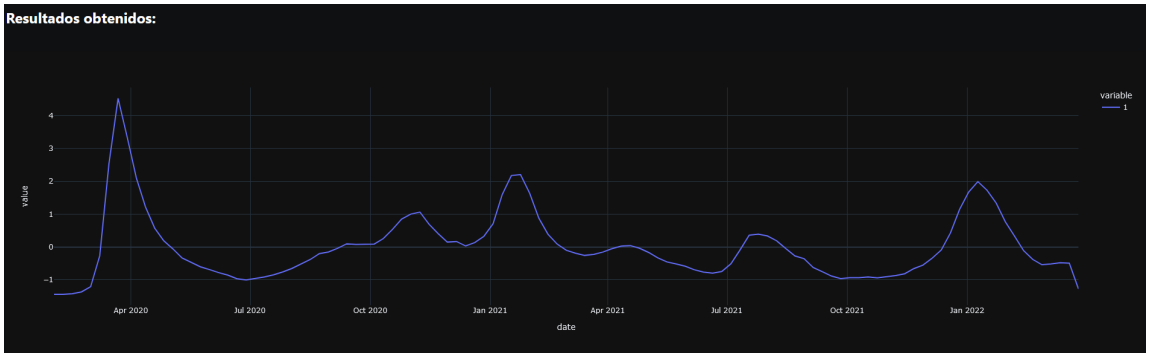

Figura 13: Variable resumen global.

Es interesante observar que al comparar con los datos de infectados, no ocurre como en la aplicación por olas, que las tendencias estaban altamente relacionadas. En este caso, la componente principal está disparada en la primera ola con respecto a los infectados y en el resto se adapta algo más, pero no tanto en escala

como en el análisis por olas.

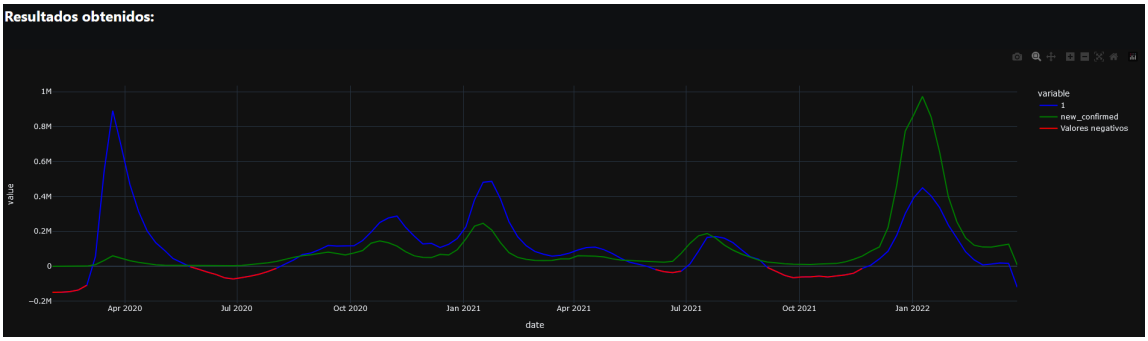

Figura 14: Variable resumen global comparada con los nuevos infectados.

Curiosamente, en este caso, la componente principal continúa teniendo una tendencia muy similar a la del número de fallecidos. Se observa el efecto de algo anteriormente comentado, existe un desfase entre la componente y los fallecidos. Este desfase comienza siendo de una semana, pero va aumentando a medida que se introducen las nuevas cepas, más contagiosas pero menos letales.

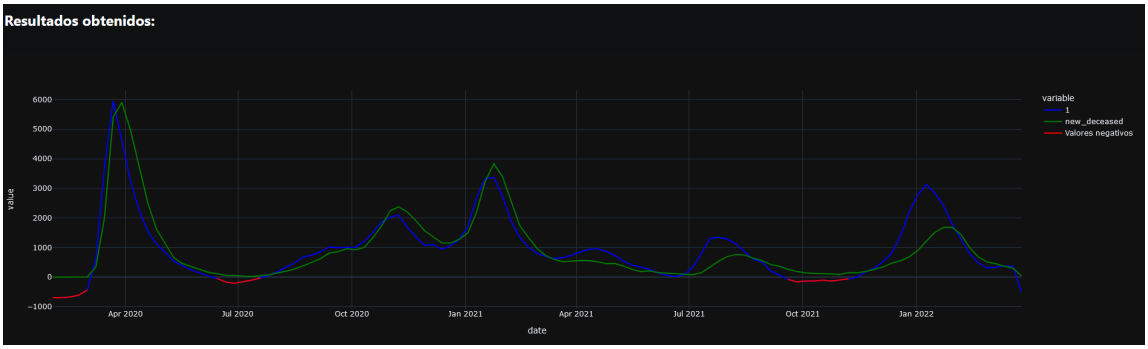

Figura 15: Variable resumen global comparada con los nuevos fallecidos.

Finalmente, reseñar que a pesar del infratesteo que ocurrió en la primera ola, de alguna forma el método detecta que la gravedad de la pandemia fue máxima en este rango temporal, asignándole un valor de X_r más alto que en ninguna otra, que está más ligado al alto número de fallecidos que a ninguna otra variable.

7.4 Comparativa con datos de vacunación, medidas y movilidad

Para finalizar nuestro análisis, vamos a comparar la variable resumen de los datos globales con algunas variables concretas que puedan mostrar la relación del estado de la pandemia con las medidas de restricción de la movilidad, higiene respiratoria y vacunación.

Comenzamos comparando con los datos de la campaña de vacunación que comienza en enero de 2021. Podemos ver que a partir de que hay más grupos de edad vacunados, la variable resumen en las últimas olas toma un valor menor en los picos de las mismas, salvo en la última, que como mencionamos, resultó grave a pesar de la campaña de vacunación por el alto número de contagios.

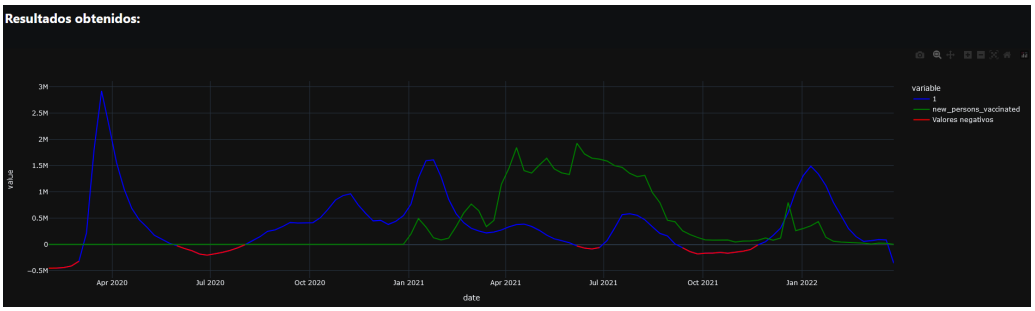

Figura 16: Variable resumen global comparada con los nuevos vacunados.

En cuanto al uso de la mascarilla, al comenzar la pandemia las directrices del gobierno eran muy laxas, insistiendo más en el distanciamiento social y en la higiene de manos, pues se creía que el virus se transmitía por contacto estrecho y por superficies más que por el aire [27]. Al poco tiempo se demostró que era al contrario, la transmisión se producía mayoritariamente por aerosoles y gotículas respiratorias. Pese a ello, no se dispuso de suficientes mascarillas hasta el final de la primera ola y la salida de la cuarentena. A partir de ahí, podemos ver que la gravedad de las olas disminuye cuando se impone el uso obligatorio de mascarilla.

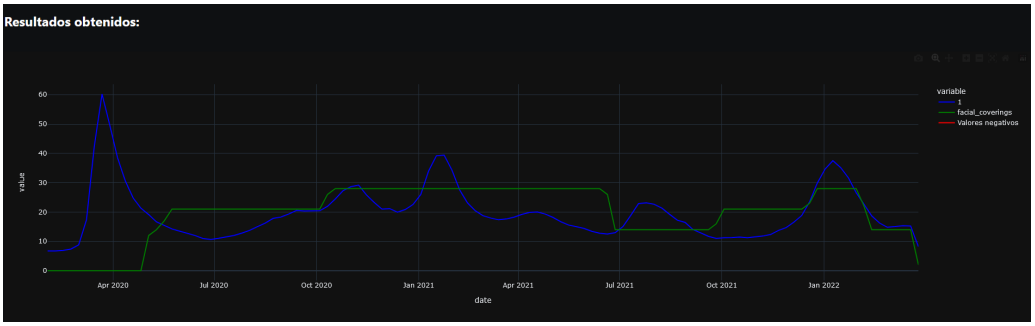

Figura 17: Variable resumen global comparada con el uso de mascarillas.

Finalmente, realizaremos una comparación con los datos de movilidad recogidos por Google. Estos datos consisten en una tasa de cambio de densidad de dispositivos en diferentes ubicaciones, con respecto a la densidad habitual calculada a lo largo de los años anteriores.

La comparación con la tasa de cambio de estancia en zonas residenciales tiene una tendencia muy similar a la de la variable resumen. En cada ola, suele reflejar las cuarentenas impuestas por el gobierno en función de la gravedad de la situación.

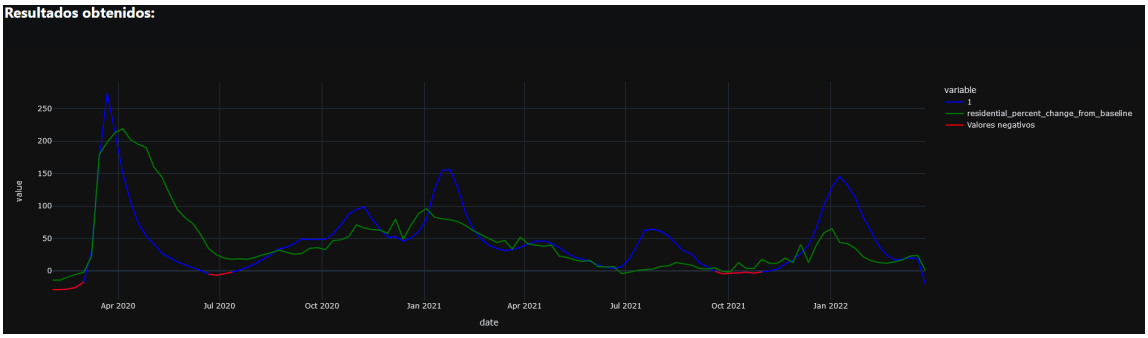

Figura 18: Variable resumen global comparada con la tasa de cambio de movilidad en zonas residenciales.

Finalmente, podemos observar el razonable efecto contrario si comparamos con ubicaciones como parques o zonas de ocio. En ambos casos, la tasa es muy baja (negativa) cuando hay restricciones y, cuando estas finalizan (normalmente con el fin de una ola) aumentan. También podemos ver que al finalizar las olas, la tasa de movilidad en parques aumenta incluso por encima de lo habitual, esto puede entenderse por la preferencia de la población por los exteriores, donde es más difícil contagiarse.

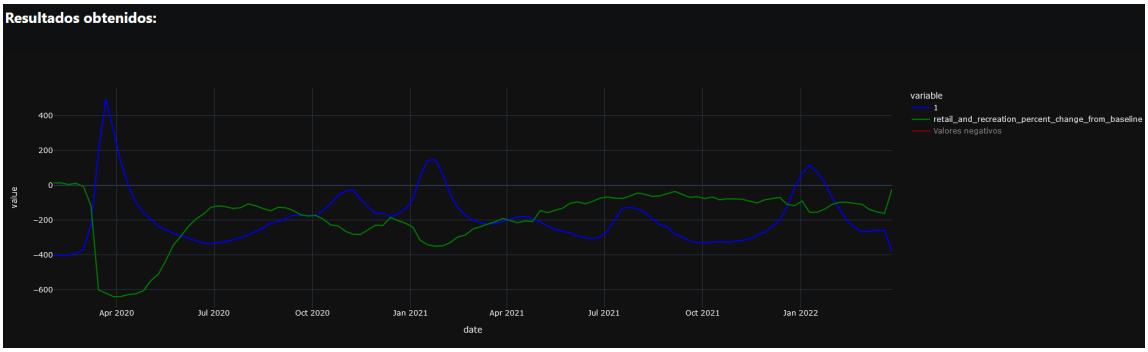

Figura 19: Variable resumen global comparada con la tasa de cambio de movilidad en lugares de ocio.

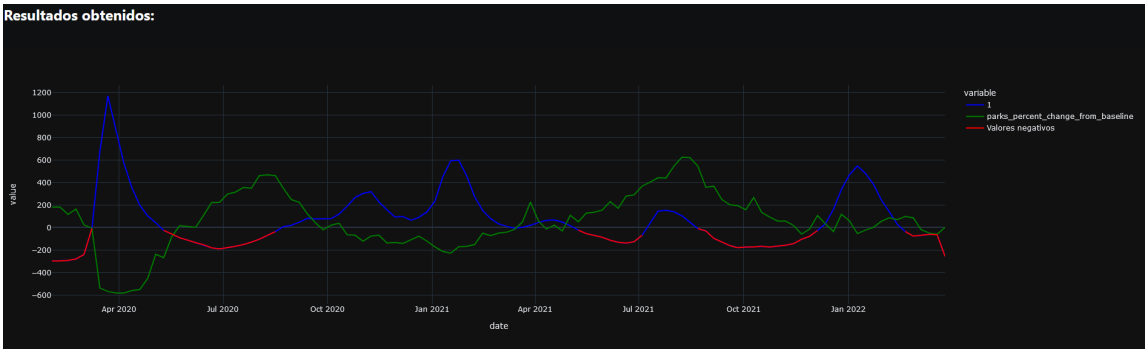

Figura 20: Variable resumen global comparada con la tasa de cambio de movilidad en parques.

7.5 Modelo SIR

Los modelos epidemiológicos [9] suelen basarse en la subdivisión de la población susceptible a la infección en un conjunto reducido de grupos. Cada grupo está compuesto por individuos que se encuentran en el mismo estado respecto a la infección. En el modelo SIR básico [30], existen tres grupos:

- Población susceptible (S): Individuos no inmunes a la infección, que pueden ser infectados si se exponen al mismo.
- Población infectada (I): Individuos que están infectados en un momento dado y pueden transmitir la infección a individuos de la población susceptible con los que tienen contacto.
- Población recuperada y fallecidos (R): Individuos inmunes a la infección o fallecidos, por lo que no afectan a la transmisión al entrar en contacto con otros individuos. En teoría de grafos, si representáramos los grupos como nodos, este sería un sumidero, ya que tiene aristas entrantes pero no salientes.

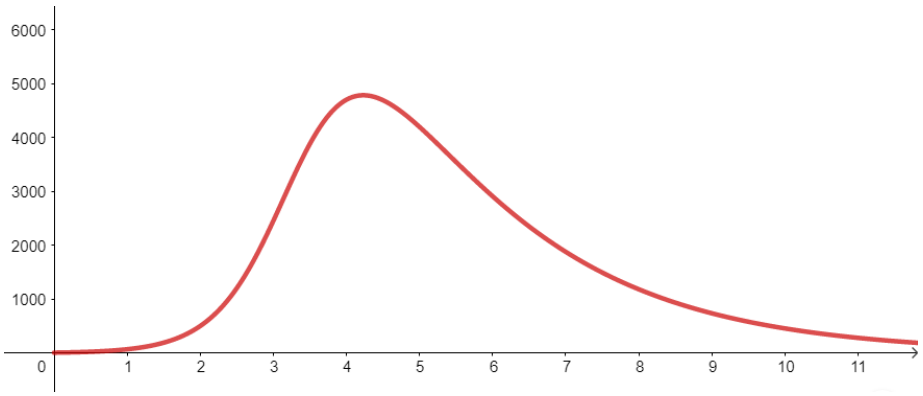

Figura 21: Curva de infectados del modelo epidemiológico SIR.

La población total es $N = S + I + R$. Una vez realizada esta agrupación, es necesario establecer ecuaciones que describan la variación temporal del número de individuos en cada uno. El perfil de la solución $I(t)$ debe ser similar a la progresión observada del número de personas infectadas. Aunque el número de individuos en cada grupo debe ser un número entero, dado el gran tamaño de la población N, las variables S, I y R pueden tomar como variables continuas. El modelo SIR se expresa mediante las siguientes ecuaciones diferenciales:

$$\begin{aligned} \frac{dS}{dt} &= -\beta SI \\ \frac{dI}{dt} &= +\beta SI - \gamma I \\ \frac{dR}{dt} &= \gamma I \end{aligned} \tag{7.1}$$

Donde, β representa la tasa de transmisión y γ la tasa de recuperación (el período medio de recuperación es $1/\gamma$).

Nosotros hemos intuido la relación con la curva de infectados, aunque los conceptos son diferentes, nuestros datos de infectados corresponden a los nuevos casos diarios y la variable $I(t)$ representa no sólo los nuevos casos, sino una estimación de los casos activos. Por tanto, puede interpretarse que la variable resumen en cada ola está relacionada con esta función $I(t)$.

8 Desarrollo de la aplicación interactiva

Una vez hemos concluido el desarrollo matemático de la teoría del análisis de componentes principales, vamos a desarrollar una aplicación web que implemente el método y facilite su utilización en un entorno No-Code que permita aplicarlo fácilmente sobre diferentes rangos temporales. Con esto se facilitará el análisis de los resultados en las diferentes olas de la epidemia.
Para el desarrollo del proyecto vamos a utilizar una metodología de la ingeniería del software conocida como desarrollo incremental [26] que procedemos a describir.

8.1 Metodología de desarrollo incremental

El desarrollo de software incremental se basa en la idea de disponer de prototipos continuamente a los que se les va incrementando la funcionalidad. El modelo aplica de forma escalonada varias secuencias de desarrollo lineales. Cada una de estas secuencias incrementa la funcionalidad del software.

Cuando se finaliza el primer incremento, el resultado es un programa que sólo contiene la funcionalidad esencial, es decir, los requisitos de funcionamiento básicos. Conforme avanzan las iteraciones, se va dotando de funcionalidad a la herramienta a la vez que se prueban y depuran las primeras versiones.

La ventajas principales de esta metodología son las siguientes:

- El software está operativo desde la primera etapa de forma temprana.
- Es un modelo muy flexible, esto reduce el coste de cambios en los requisitos y alcance de la aplicación. Por ello es una buena opción para un proyecto de investigación como el que se lleva a cabo.
- Al tener iteraciones más reducidas, se disminuyen los problemas que puedan surgir en la depuración o en el testeo
- El desarrollo es muy estable y permite actualizar funcionalidades de las primeras fases con requisitos que hayan surgido a mitad del proyecto.
- Es adecuado para pequeños proyectos como este, para proyectos muy complejos, pueden surgir problemas por las interdependencias entre hitos.
- Las fases finales de una iteración pueden solaparse con el comienzo de la siguiente, de esta manera se pueden solucionar las posibles taras en la nueva planificación.

En general, se suele entender que una iteración está compuesta por las siguientes fases:

1. Comunicación
2. Planificación
3. Modelado
4. Implementación y prueba
5. Despliegue

La siguiente ilustración es muy descriptiva del funcionamiento del modelo incremental:

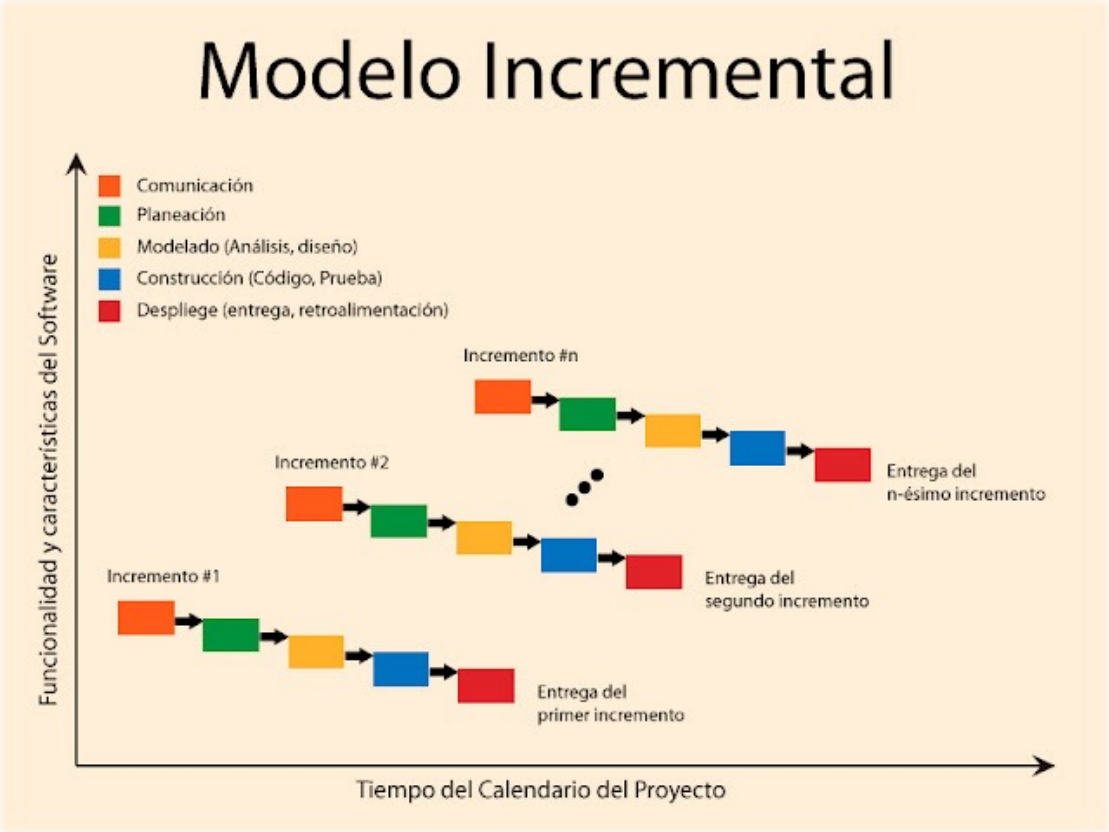

Figura 22: Desarrollo incremental. Gráfica tomada de [25] a su vez adaptada de [26]

En el caso concreto de este proyecto, la comunicación se lleva a cabo entre el tutor y el alumno discutiendo las funcionalidades a implementar. La planificación la realiza el alumno compatibilizándola con el desarrollo matemático del trabajo. Finalmente las fases de modelado, implementación y despliegue también corren a cargo del alumno.

8.2 Análisis de requisitos

El objetivo de este proyecto es disponer de una herramienta sencilla con la que estudiar el modelo desarrollado con datos reales del desarrollo de la epidemia en diferentes fases. Además, y de forma colateral, se pretende mostrar de forma interactiva las posibles relaciones entre las variables indicadoras de la pandemia clásicas y otras menos utilizadas de las que se disponen gracias a las nuevas tecnologías. Por tanto, el objetivo es doble, diseñar una herramienta para el estudio del comportamiento del modelo matemático y a su vez una herramienta de divulgación de los posibles factores injerentes en el desarrollo de la epidemia COVID-19.

Por tanto, después de todas las iteraciones del modelo incremental, la lista de requisitos es la siguiente:

8.2.1 Requisitos funcionales

- **RF1**: Se deben poder visualizar las distintas series temporales de datos, ya sea de forma diaria o semanal, con o sin normalizar.
- **RF2**: El usuario debe poder escoger los rangos temporales en los que explorar los datos.
- **RF3**: El sistema debe permitir activar o desactivar las variables que se quieren visualizar.
- **RF4**: El sistema debe permitir seleccionar grupos de variables comparables.
- **RF5**: El sistema debe permitir descargar los datos de la pandemia que se muestran.
- **RF6**: El sistema debe permitir elegir a que rango aplicar el método de análisis.
- **RF7**: Los resultados se representarán en una nueva gráfica independiente. Además, se mostrará la varianza explicada y un breve informe.
- **RF8**: El sistema debe permitir comparar el resultado con las variables de partida.
- **RF9**: El sistema debe permitir descargar la primera componente principal resultante.

8.2.2 Requisitos no funcionales

- **RNF1**: La gráficas deben ser interactivas, permitir zooms y translaciones.
- **RNF2**: Las gráficas deben actualizarse en tiempo real.

- **RNF3**: Se mostrarán unas indicaciones del funcionamiento de la aplicación.
- **RNF4**: Se debe poder utilizar en un navegador web.

8.2.3 Requisitos de información

- **RI1**: Los ficheros generados por la aplicación tanto con los datos originales como los resultantes deben estar en formato .CSV.
- **RI2**: Los gráficas deben poder exportarse en cada estado en formato .PNG.

8.3 Recursos a utilizar y estimación del coste

En esta sección, recopilaremos todos los recursos que se han utilizado durante el desarrollo del proyecto. Podemos dividirlos en recursos software y hardware y recursos humanos.

8.3.1 Recursos humanos

En principio se cuenta con 2 personas, un encargado del desarrollo y un responsable de tutorización. Sin perjuicio de posibles consultas a especialistas del ámbito del software, del cual el tutor no es especialista.

- Desarrollador: Ignacio Garach Vélez
  - Planificación y modelado del proyecto.
  - Diseño e implementación de la aplicación.
  - Testeo en cada etapa del modelo incremental.
  - Redacción de la memoria y la documentación del proyecto.
- Responsable de tutorización: Juan Campos Rodríguez
  - Concepto de partida del proyecto.
  - Solicitud de aprobación del mismo a la comisión de TFG del doble grado.
  - Apoyo en el análisis de resultados y en la redacción de la memoria.
  - Supervisar el desarrollo del proyecto.

8.3.2 Recursos software

En esta sección se describen las tecnologías software utilizadas para el desarrollo del proyecto:

- Linux: Es el sistema operativo completamente libre y gratuito por excelencia. Es altamente personalizable y configurable y existen gran variedad de distribuciones del mismo para satisfacer distintas necesidades. En nuestro caso hemos optado por utilizar Linux Mint, una distribución compacta que viene con muchas funcionalidades software preinstaladas y está basada en Ubuntu, una de las distribuciones más utilizadas.
- Python: Es un lenguaje de programación de alto nivel, interpretado y de propósito general. Es conocido por su sintaxis simple y legible. En la actualidad es uno de los lenguajes más usados y sin duda el más utilizado para análisis de datos y técnicas de aprendizaje automático por la gran cantidadd de librería específicas y la gran comunidad que trabaja con las

mismas. Se utiliza también para desarrollo web, en concreto mediante los frameworks Django o Flask. Nosotros indirectamente, utilizaremos Flask a través de Dash.

- Jupyter Notebook: Es una herramienta interactiva de programación y análisis de datos muy utilizada en la ciencia de datos y la investigación. Nos ofrece un entorno basado en web que permite la creación de documentos enriquecidos que combinan celdas de código ejecutable, texto explicativo y demás elementos. Es interesante también para desarrollos matemáticos pues se le puede integrar texto LaTeXde forma sencilla. Los Notebooks de Jupyter son muy versátiles y admiten múltiples lenguajes, incluidos Python, R y Julia. La capacidad de ejecutar y probar código de forma interactiva sobre los datos, junto con la visualización en tiempo real de los resultados ha sido crucial para elegirlo para el desarrollo de la parte de análisis de datos del proyecto.

- Dash Plotly: Es un framework de Python que permite crear aplicaciones web interactivas con gráficas para visualizar datos de forma sencilla. Permite combinar código Python, HTML y CSS para crear paneles de control y aplicaciones interactivas que respondan a las acciones del usuario (sistema de callbacks). Además, Dash tiene una documentación sólida [24] y una comunidad activa en la que es fácil resolver los problemas que surjan, lo que facilita la curva de aprendizaje, en el caso de este desarrollo, se ha partido de cero. A nivel interno, se basa en un servidor Flask que se comunica con un Front-End basado en React.js a través de JSON. Aunque proporciona muchas opciones con sus llamadas Core Components, podemos enriquecer la aplicación con HTML, CSS o incluso JavaScript.

- CSS: La hojas de estilo en cascada se utilizan para dar estilo y presentación a documentos HTML. Permite definir la apariencia del HTML con colores, fuentes, márgenes o diseños predefinidos. CSS es muy flexible, lo que facilita crear diseños atractivos y responsivos. Con CSS, podemos separar el contenido de los estilos, contribuyendo a la separación Modelo-Vista-Controlador. En nuestro caso, aunque Dash permite controlar la mayoría de los estilos, para detalles específicas permite que carguemos CSS para modificar el HTML que genera.

- Librerías Python: Numpy, Pandas y Scikit-learn son tres librerías ampliamente utilizadas en el ámbito del análisis de datos y la ciencia de datos.

    - Numpy es una librería básica de cálculo numérico en Python. Proporciona estructuras de datos eficientes, como arrays multidimensionales e implementaciones de funciones matemáticas con buen rendimiento, lo que facilita manipular y operar con grandes conjuntos de datos.
    - Pandas se centra en la manipulación y el análisis de datos estructurados. Ofrece la estructura de datos por excelencia en el análisis de

datos, los DataFrames, que permiten trabajar con datos tabulares de forma intuitiva. Tiene integradas muchas herramientas para la limpieza y manipulación de datos, lo que simplifica mucho el preprocesamiento de los mismos [19].

- Scikit-learn es una librería de aprendizaje automático que ofrece gran cantidad de algoritmos para tareas como clasificación, regresión, agrupamiento y selección de características. En nuestro caso, vamos a utilizar principalmente los módulos de estandarización y el de descomposición en componentes principales [22].

- Gunicorn: Green Unicorn es un servidor web compatible con aplicaciones escritas en Python. Se utiliza comúnmente para entornos de producción. Gunicorn tiene un funcionamiento muy simple y es capaz de manejar múltiples solicitudes simultáneamente de manera eficiente. Proporciona una interfaz sencilla para configurar y ejecutar servidores web Python y es compatible con frameworks como Flask y Django, o en nuestro caso Dash que como sabemos está basado en Flask.

- Overleaf: Interfaz online gratuita para redacción de documentos en LaTeX. Se ha utilizado para el desarrollo de la memoria.

- Git y Github: Se han utilizado como controlador de versiones y para poder desplegar versiones parciales en la nube Render.

- VsCode: Como editor de código, hemos optado por esta alternatica gratuita que es compatible con una gran variedad de lenguajes gracias a sus plugins, además puede integrarse con git.

8.3.3 Recursos hardware

Vamos a dividirlo en 2 partes, por un lado los recursos propios y por otro los que se utilizarán en Google Cloud Platform.

Como recursos propios se utilizó un ordenador de sobremesa para el desarrollo con las siguientes características principales:

- **CPU**: AMD Ryzen 7 3700X 8-core.

- **GPU**: NVIDIA GeForce 1660 SUPER.

- **RAM**: 32GB RAM DDR4 HyperX.

- **Disco duro SSD**: 500 GB.

- **Disco duro HDD**: 2TB.

- **Precio**: 1100€.

Como recursos tomados mediante App Engine en GCP, se utilizarán un máximo de 2 instancias F4:

- **CPU**: Límite de 2.4 GHz.
- **RAM**: 1GB RAM.
- **Disco duro HDD**: 2GB.
- **Precio**: Nivel gratuito hasta alcanzar límite de solicitudes.

Contando con todo ello vamos a tratar de detallar un presupuesto de gastos lo más realista posible. Los gastos en recursos humanos vamos a calcularlos en función de los créditos ECTS, como el TFG de este Doble Grado tiene asignados 18 créditos, esto corresponde a 450 horas que podemos valorar a 14 euros la hora, si suponemos un salario de alumno en prácticas. Por otro lado, calculamos que el tutor ha dedicado en promedio unas 2 horas semanales que dada su formación y su posición de catedrático podemos estimarles un coste de 40 euros la hora. Además, como las tecnologías utilizadas son de uso libre y gratuito y los gastos del despliegue en Google Cloud dada la baja carga esperada son casi despreciables (de hecho nulos, porque se está utilizando la versión de prueba), sólo queda añadir el coste por amortización del equipo de sobremesa al que se le supone una vida media de 8 años.

Tabla 1: Desglose presupuestario del proyecto.

Recursos humanos	**Coste (€)**
Alumno	6.300,00
Tutor	2.560,00
Suma	**8.860,00**
Recursos hardware	**Coste (€)**
Ordenador de sobremesa	103.12
Suma	**103.12**
Suma total	**8.963,12**
IVA (21 %)	**1882,25**
Coste final	**10.845,37**

8.4 Planificación temporal

Para el desarrollo del proyecto, aunque se ha seguido un modelo incremental para la implementación del código del modelo y de la aplicación web, se han distinguido las distintas etapas anteriores y posteriores. Las distintas partes del proyecto han sido:

- **P1. Estudio de bibliografía y comprensión del método**. Se ha consultado la bibliografía proporcionada por el profesor, aunque no era rigurosa, porque estaba centrada en la aplicación directa del método que se utiliza en el ámbito de la economía. Por tanto, se han consultado diversos libros y artículos para poder establecer con rigor los conceptos matemáticos que se utilizan en el proyecto.
- **P2. Desarrollo matemático**. Se han introducido como preliminares los conceptos necesarios sobre variables aleatorias y espacios vectoriales euclídeos. Se ha efectuado el planteamiento del problema para pasarlo del ámbito estadístico al analítico-geométrico. Finalmente se ha desarrollado una teoría paralela a la que se usa habitualmente en el análisis de componentes principales, en lugar de usar el método de optimización de Lagrange, se optimiza una forma cuadrática en un dominio compacto.
- **P3. Búsqueda y limpieza de datos**. Una vez tenemos desarrollado el método, se han recopilado gran cantidad de variables tanto de salud como socioeconómicas a las que aplicarselo. Se ha efectuado a continuación una limpieza de los mismos y un breve análisis exploratorio.
- **P4. Análisis de requisitos**. En base a la teoría obtenida en la primera parte, en esta etapa se han especificado los requisitos que el software debe cumplir, tanto el modelo matemático como la aplicación web.
- **P5. Diseño del sistema**. A la vista del listado de requisitos obtenido, se han almacenado los datos, se han escogido las tecnologías a utilizar, se ha diseñado la estructura de la aplicación y la interfaz de usuario.
- **P6. Implementación del modelo basado en la librería Scikit-Learn**. Se ha realizado la implementación del análisis de componentes principales a partir de un conjunto de datos, en resumen se normalizan y calculan las correlaciones mediante Scikit-Learn, a partir de ellas se obtienen mediante diagonalización las componentes.
- **P7. Implementación de la aplicación web Dash**. Se ha realizado la implementación de la aplicación web en base al modelado, siguiendo un desarrollo incremental en el que se han incluido nuevas funcionalidades en cada etapa.
- **P8. Aplicación del método a datos de la epidemia en España**. Con la ayuda de la aplicación desarrollada, se ha aplicado el método en conjunto y a las

distintas olas. Se ha estudiado su comportamiento y tratado de relacionar sus resultados con el modelo epidemiológico SIR.

- **P9. Despliegue de la aplicación**. Se ha desplegado la aplicación de forma pública a través del servicio App Engine de la plataforma GCP. De esta forma la aplicación ya es accesible desde cualquier lugar a través de un navegador web.
- **P10. Documentación del proyecto**. Finalmente, se ha realizado la documentación del proyecto en la presente memoria, tanto la parte del desarrollo matemático como la informática.

El desarrollo del proyecto comenzó en noviembre de 2022 tras su asignación y aprobación y se distribuyó temporalmente como se indica en el siguiente cronograma. Observamos que por la metodología de desarrollo incremental, se solapan y alargan en el tiempo las etapas de extracción de requisitos y modelado con las de implementación como tal. Notemos también que las etapas de desarrollo matemático y análisis de datos se han solapado con el desarrollo porque el producto inicial software ya era aplicable en su forma primitiva a conjuntos de datos.

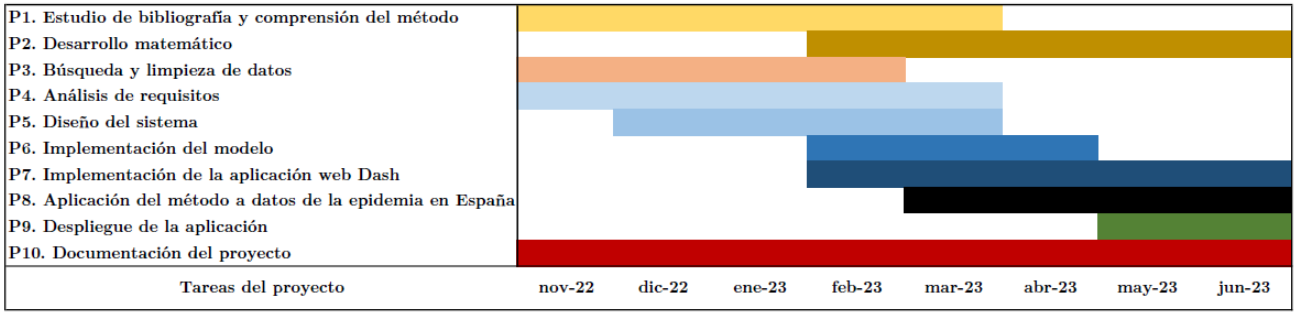

Figura 23: Diagrama con la planificación temporal de tareas

8.5 Casos de uso

Atendiendo a los requisitos del sistema se han definido los siguientes casos de uso, para los cuales se ha completado su plantilla de descripción extendida. Además, se ha generado un diagrama con los mismos.

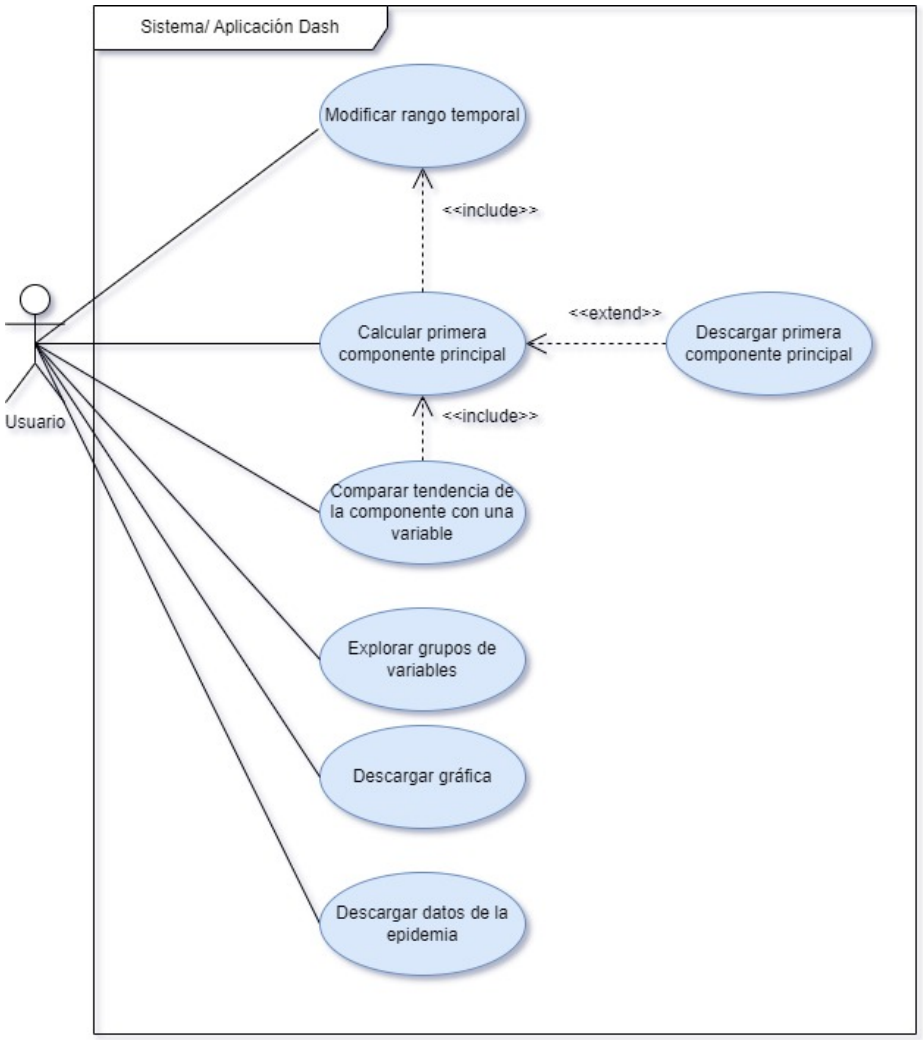

Figura 24: Diagrama de casos de uso UML

Caso de Uso	Explorar grupos de variables				CU-01
Actores	Usuario				
Tipo	Primario				
Referencias					
Precondición	La página debe estar inicializada y los datos cargados.				
Autor	Ignacio Garach	**Fecha**	14-02-23	**Versión**	1.0

Propósito
Permitir al usuario realizar un análisis exploratorio de las variables aleatorias a las que se les va a aplicar el análisis. Puede escogerse entre vista diaria o semanal y uso o no de normalización. Cambiar entre distintos grupos comparables.

Curso Normal			
1	El usuario carga la web y visualiza los distintos grupos de variables disponibles.		
2	El usuario selecciona la casilla correspondiente al grupo que desea visualizar. También selecciona el modo en que desea hacerlo.		
		3	El sistema actualiza la gráfica en tiempo real con las variables seleccionadas

Cursos Alternos	
(4)	Si el usuario desactiva una variable concreta, también se actualiza la gráfica.

Otros datos			
Frecuencia esperada	Habitual	**Rendimiento**	Medio
Importancia	Media	**Urgencia**	Alta
Estado	Funcional	**Estabilidad**	Alta

Caso de Uso	Modificar rango temporal				CU-02
Actores	Usuario				
Tipo	Secundario				
Referencias					
Precondición	La página debe estar inicializada y los datos cargados.				
Autor	Ignacio Garach	**Fecha**	14-02-23	**Versión**	1.0

Propósito
Permitir al usuario seleccionar una fecha de inicio y fin de los datos para los cuales se mostrarán los datos y se podrá aplicar el método de análisis de componentes principales.

Curso Normal			
1	El usuario introduce fechas de inicio a fin ya sea desde el selector o seleccionando un rango en la gráfica interactiva.		
		2	El sistema actualiza la gráfica en tiempo real mostrando el rango seleccionado y lo almacena para cálculos posteriores.

Cursos Alternos	
(3)	Si el usuario pulsa el botón de resetear gráfica en cualquier momento se resetean los rangos al máximo.

Otros datos			
Frecuencia esperada	Habitual	**Rendimiento**	Medio
Importancia	Alta	**Urgencia**	Alta
Estado	Funcional	**Estabilidad**	Alta

Caso de Uso	Descargar datos de la epidemia			CU-03	
Actores	Usuario				
Tipo	Secundario				
Referencias					
Precondición	La página debe estar inicializada y los datos cargados.				
Postcondición					
Autor	Ignacio Garach	**Fecha**	14-02-23	**Versión**	1.0

Propósito
Permitir al usuario descargar todas las mediciones de las distintas variables a lo largo del periodo de análisis en formato CSV.

Curso Normal			
1	El usuario pulsa el botón de descargar datos originales.		
		2	El sistema lanza la descarga y el fichero CSV correspondiente se descarga al dispositivo desde el que se realiza la petición.

Cursos Alternos	
	No se conocen

Otros datos			
Frecuencia esperada	Ocasional	**Rendimiento**	Alto
Importancia	Media	**Urgencia**	Alta
Estado	Funcional	**Estabilidad**	Alta

Caso de Uso	Calcular primera componente principal			CU-04	
Actores	Usuario				
Tipo	Primario				
Referencias	CU-02				
Precondición	La página debe estar inicializada y los datos cargados. Además debe haberse seleccionado el rango temporal en el que aplicarlo.				
Postcondición	Los resultados están disponibles para descargar.				
Autor	Ignacio Garach	**Fecha**	14-02-23	**Versión**	1.0

Propósito
Permitir al usuario calcular la primera componente principal de las variables aleatorias. Para ello ejecuta el método descrito en la sección del desarrollo matemático

Curso Normal			
1	El usuario lanza el caso de uso CU-02		
		2	El sistema ejecuta el caso de uso CU-02
3	El usuario pulsa el botón de Aplicar Método		
		4	El sistema ejecuta el método y diagonaliza la matriz. Además calcula la componente principal y la muestra en una nueva gráfica.

Cursos Alternos	
	No se conocen

Otros datos			
Frecuencia esperada	Ocasional	**Rendimiento**	Alto
Importancia	Media	**Urgencia**	Alta
Estado	Funcional	**Estabilidad**	Alta

Caso de Uso	Comparar tendencia de la componente principal con una variable.		CU-05
Actores	Usuario		
Tipo	Primario		
Referencias	CU-04		
Precondición	Debe haberse calculado la primera componente principal		
Postcondición			
Autor	Ignacio Garach	**Fecha** 14-02-23	**Versión** 1.0

Propósito

Permitir al usuario comparar las tendencias de cada variable con la componente principal en una gráfica. Esto facilita el análisis de los resultados.

Curso Normal

1	El usuario lanza el caso de uso CU-04		
		2	El sistema ejecuta el caso de uso CU-04
3	El usuario selecciona una variable en el selector para comparar		
		4	El sistema devuelve la media y varianza de la selección a la componente y la representa junto a esta en una gráfica a tiempo real.

Cursos Alternos

	No se conocen

Otros datos

Frecuencia esperada	Habitual	**Rendimiento**	Alto
Importancia	Alta	**Urgencia**	Alta
Estado	Funcional	**Estabilidad**	Alta

Caso de Uso	Descargar primera componente principal		CU-06
Actores	Usuario		
Tipo	Secundario		
Referencias			
Precondición	La primera componente principal se ha calculado.		
Postcondición			
Autor	Ignacio Garach	**Fecha** 14-02-23	**Versión** 1.0

Propósito

Permitir al usuario descargar los resultados en formato CSV para posteriores usos de predicción.

Curso Normal

1	El usuario pulsa el botón de descargar datos resultantes.		
		2	El sistema lanza la descarga y el fichero CSV correspondiente se descarga al dispositivo desde el que se realiza la petición.

Cursos Alternos

	No se conocen

Otros datos

Frecuencia esperada	Ocasional	**Rendimiento**	Alto
Importancia	Media	**Urgencia**	Alta
Estado	Funcional	**Estabilidad**	Alta

Caso de Uso	Descargar gráfica con series temporales				CU-07
Actores	Usuario				
Tipo	Secundario				
Referencias					
Precondición					
Postcondición					
Autor	Ignacio Garach	**Fecha**	14-02-23	**Versión**	1.0

Propósito
Permitir al usuario descargar en formato PNG el estado actual de cualquier gráfica.

Curso Normal			
1	El usuario pulsa el botón de descargar gráfica.		
		2	El sistema lanza la descarga y el fichero PNG correspondiente se descarga al dispositivo desde el que se realiza la petición.

Cursos Alternos	
	No se conocen

Otros datos			
Frecuencia esperada	Ocasional	**Rendimiento**	Medio
Importancia	Media	**Urgencia**	Alta
Estado	Funcional	**Estabilidad**	Alta

8.6 Sistema de callbacks

El sistema de callbacks de Dash Plotly es una de las características fundamentales que facilitan crear aplicaciones web interactivas y dinámicas con la herramienta. Los callbacks son la forma en que Dash permite la interacción en tiempo real entre los componentes de la interfaz de usuario y los datos subyacentes, así como con las funciones que definan la lógica del sistema.

El funcionamiento de los callbacks se basa en la idea de que los componentes de la interfaz de usuario (como botones, deslizadores o incluso las propias gráficas) pueden generar eventos (triggers), como hacer clic o cambiar de valor. Estos eventos pueden desencadenar la ejecución de una función específica que actualiza los componentes de salida que se muestran al usuario o realiza acciones en función de los nuevos datos o parámetros.

Para crear un callback en Dash, se debe definir una función decorada con un decorador especial llamado @app.callback [24]. Este decorador especifica qué componentes de entrada desencadenarán la función (triggers) y qué componentes de salida se actualizarán en función de los cambios. Además, en la función decorada se pueden especificar en orden los argumentos de entrada y salida de la función para acceder a los valores actuales de los componentes y actualizarlos en consecuencia. Por otro lado, también se pueden agregar en el decorador objetos de estado, a los que la función necesite acceder para efectuar sus cálculos. En estos objetos, se suele almacenar información proveniente de cálculos costosos, ya que la implementación de Dash con concurrencia es serverless, es decir, no deben utilizarse variables globales. Los objetos de tipo estado no son Inputs que puedan desencadenar el callback.

El sistema de callbacks permite una amplia gama de interacciones. Por ejemplo, se puede actualizar un gráfico en respuesta a un cambio en un deslizador, filtrar y actualizar una tabla de datos según una selección en un menú desplegable o incluso llamar a una API externa para obtener nuevos datos y mostrarlos en tiempo real.

En nuestro caso se han implementado los siguientes callbacks:

- Actualizar gráfica de exploración: A partir de las interacciones con los botones que muestran los distintos modos de visualización y grupos de variables, actualiza en tiempo real la gráfica correspondiente. También actualiza el rango temporal mostrado si se cambia el mismo a través del calendario doble selector.

- Actualizar calendario selector: A partir de las selecciones de rango hechas directamente sobre la gráfica interactiva, actualiza el calendario para mantenerlos sincronizados.

- Aplicar Método: Cuando el usuario pulsa el botón correspondiente, el sistema recopila el estado de los objetos que le indican a que rango temporal y sobre que tipo de datos aplicar el método. Después llama a la función que calcula el resultado y genera el informe, los objetos que lo muestran y habilita las diferentes componentes para análisis del resultado. (se amplía esta información en el manual de usuario). El sistema permite aplicaciones sucesivas en una misma sesión, por ello el callback está preparado para responder a llamadas sucesivas de forma distinta a la primera para mantener la lógica y la estructura de la aplicación.

- Actualizar gráfica de componentes: Al seleccionar una variable para comparar en un desplegable, se efectua el proceso de devolución de varianza descrito al final de la sección 5. Además, se muestran la variable seleccionada con la variable resumen escalada en una gráfica.

- Descarga de datos originales y de resultados: Lanzan la descarga de un fichero CSV al navegador del usuario trás pulsar el botón deseado.

En la siguiente figura se muestra como interaccionan los callbacks con los distintos objetos del sistema:

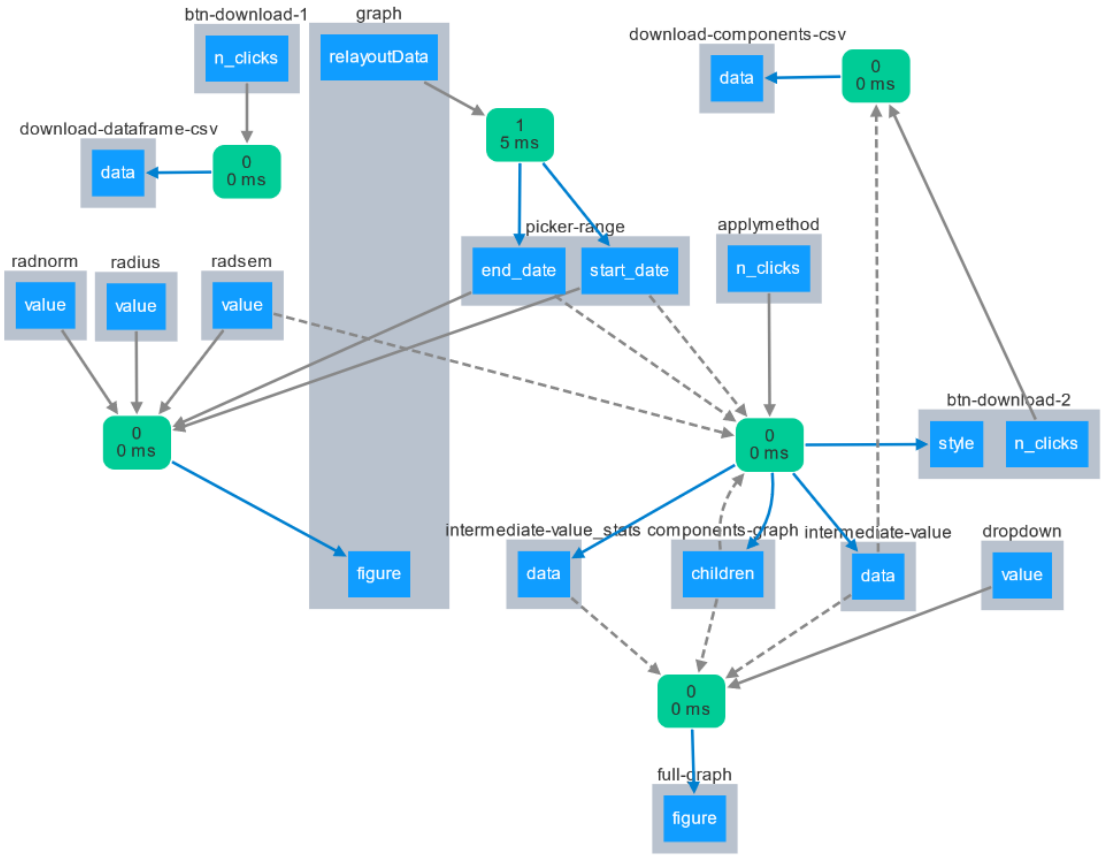

Figura 25: Diagrama de callbacks de Dash

8.7 Diseño de la interfaz

Hemos diseñado la interfaz mediante los objetos que ofrece dash en sus core components y en sus HTML components, además se han retocado algunos valores con CSS. La estructura planeada sigue el formato del siguiente boceto en un estilo Single Page Application:

Todas las secciones que aparecen en la web desde el momento de carga se generan con el siguiente código:

```
app.layout = html.Div([
html.H1(children="Analisis de componentes principales: COVID 19"),
html.H2(children='Exploracion de variables'),
html.P(children='En esta seccion puedes explorar los datos de las distintas, olas...'),
dcc.RadioItems(options = ['Basicas', 'Infectados',...], value='Basicas', ...),
html.Button("Descargar datos originales en .CSV", id="btn-download-1", className='buttons'),
dcc.Download(id="download-dataframe-csv"),
dcc.Graph(figure=fig, id='graph'),
html.P(children='Una vez hayas terminado de explorar, selecciona el rango de fechas...'),

dcc.DatePickerRange(
id='picker-range',
min_date_allowed=date(2020, 2, 1),
max_date_allowed=date(2022, 3, 28),
start_date=date(2020, 2, 1),
end_date=date(2022, 3, 28),
),
html.Button('APLICA EL METODO DE COMPONENTES PRINCIPALES', id='applymethod', n_clicks=None,...),
html.Button("Descargar componentes en .CSV", ... style={'visibility':'hidden'}),
dcc.Download(id="download-components-csv"),
html.Div([], id="components-graph"),
dcc.Store(id='intermediate-value'),
dcc.Store(id='intermediate-value_stats'),
])
```

El resto de componentes que están ocultas de base y aparecen al ejecutar el método dependen de los callbacks o manejadores de eventos de Dash. Por ejemplo la visibilidad del botón de descarga la modifica un callback, y los resultados y el resto de selectores para comparación se introducen en el Div *components-graph* mediante el callback correspondiente.

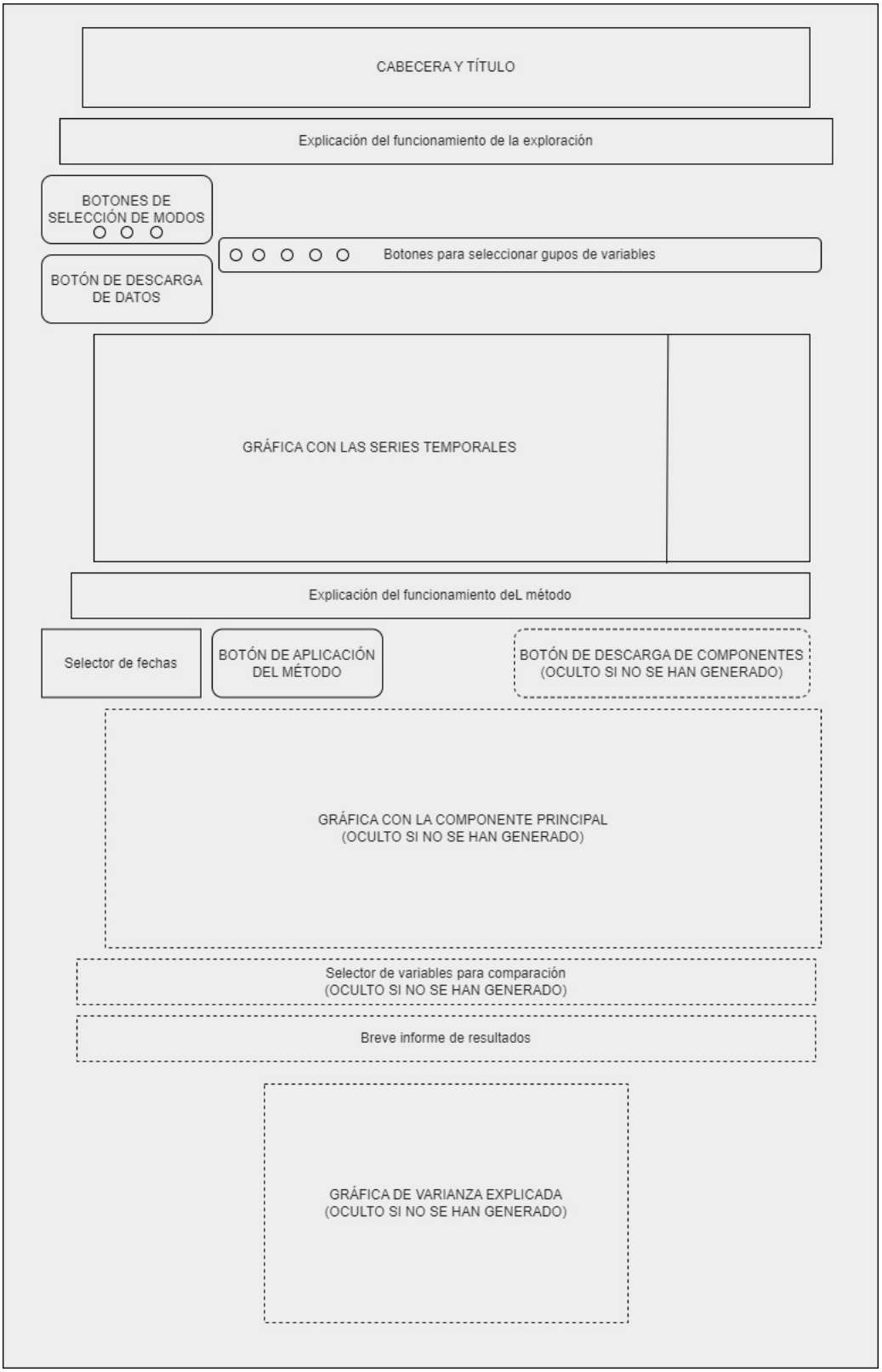

Figura 26: Boceto de la interfaz de usuario

8.8 Pruebas del sistema

De manera paralela a la implementación de nuevas funcionalidades del proyecto se han ido realizando pruebas y testeos. Esta es una de las ventajas del modelo de desarrollo incremental, como se tiene una versión preliminar del sistema pronto, se puede probar y perfeccionar en las siguientes versiones. Podemos distinguir los siguientes tipos de pruebas:

- Pruebas unitarias: Se trata de pruebas en las que se toma una pequeña parte de la aplicación, por ejemplo una función o un subsistema, se le aisla del resto del código y se prueba su comportamiento.En nuestro caso, como para el desarrollo parcial se ha utilizado Jupyter Notebook, esto se simplifica bastante por su interfaz interactiva. Hemos realizado las siguientes pruebas.
  - Pruebas numéricas: Se ha verificado que en la implementación de las ecuaciones del método, lo resultados siempre respetan el estudio teórico y que los posibles errores de aproximación no generan problemas.
  - Pruebas gráficas: Se han verificado todos los datos que se suministran a las gráficas interactivas de Plotly.
  - Pruebas de descarga: Se ha comprobado que se generan correctamente los CSV con los datos y los resultados provenientes de DataFrames.
- Pruebas de usuario: Simulan la utilización del producto software que realizará el usuario final. En nuestro caso están íntimamente relacionadas con el buen funcionamiento del sistema de callbacks.
  - Pruebas de interacción: Se ha comprobado para todos los callbacks implementados, su respuesta en todas las situaciones posibles de estado de la aplicación, que responden de la forma esperada y no generan errores lógicos ni estructurales.
  - Responsividad: Se ha comprobado la interfaz de usuario en los navegadores más usuales para comprobar que no hay problemas con el viewport ni con la responsividad de las gráficas y las funciones.
  - Prueba de aplicaciones sucesivas: Se ha comprobado que el sistema es capaz de actualizar la aplicación a tiempo real cuando se aplica el método en sucesivas ocasiones a diferentes rangos temporales.
  - Interacción con las gráficas: Las gráficas se pueden manipular de forma interactiva, resetearse y descargar su estado en cualquier momento.

9 Manual de usuario

El software consiste en una aplicación web de estilo SPA (Single Page Application) que se actualiza dinámicamente que tiene un doble objetivo. Por un lado, permite la aplicación del método de análisis de componentes principales a los datos de la epidemia COVID en España, así como la exploración de variables de salud, de movilidad o socioeconómicas. Tiene por tanto también un objetivo de divulgación sobre los datos de la pandemia y las diversas relaciones entre las distintas variables.

Para acceder a la aplicación, se debe introducir la URL en la que se encuentre desplegada en cualquier navegador web habitual, actualmente está disponible en

https://pca-tfg-igarachv.oa.r.appspot.com

9.1 Exploración de variables

Una vez cargada la página, los datos ya están disponibles y por defecto aparecen agrupados de forma semanal y sin normalizar en la gráfica.

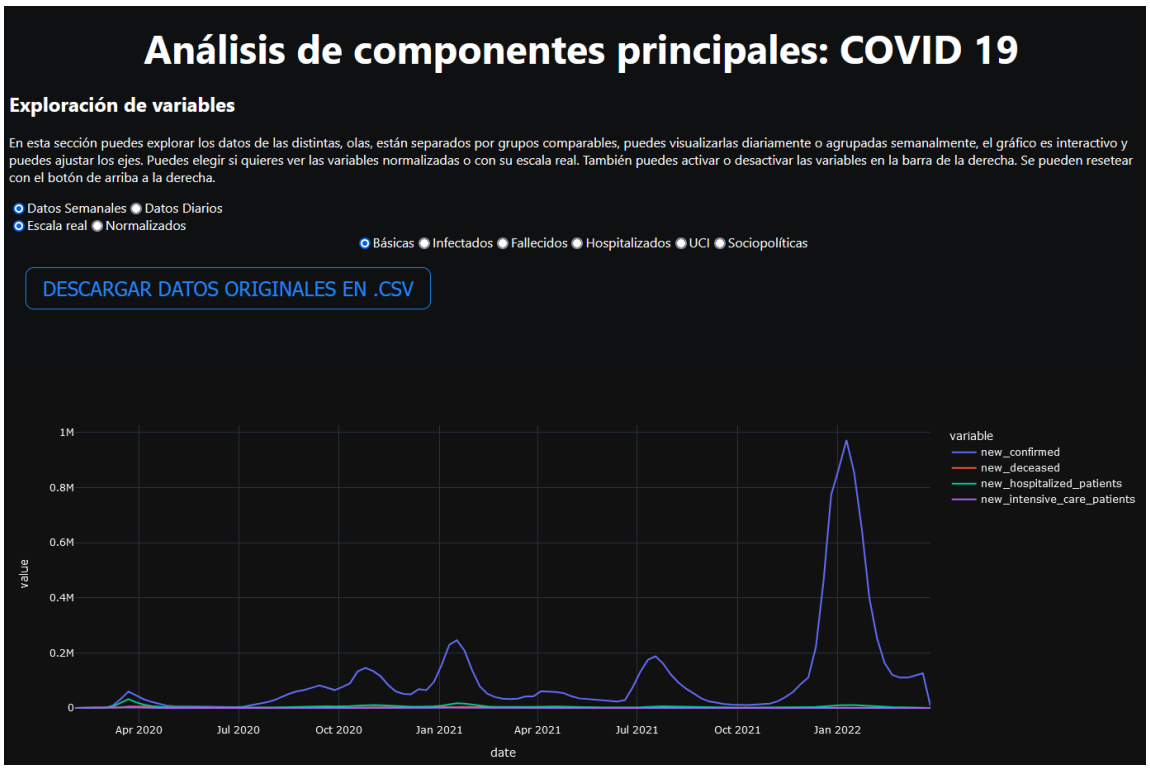

Figura 27: Aspecto inicial de la aplicación

Con las casillas de la parte izquierda podemos cambiar de forma interactiva entre las distintas posibilidades. Podemos escoger si queremos visualizar los datos normalizados o de forma diaria en lugar de semanal. Además, en las ca-

sillas de arriba podemos escoger que grupo de variables comparables queremos visualizar en la gráfica.

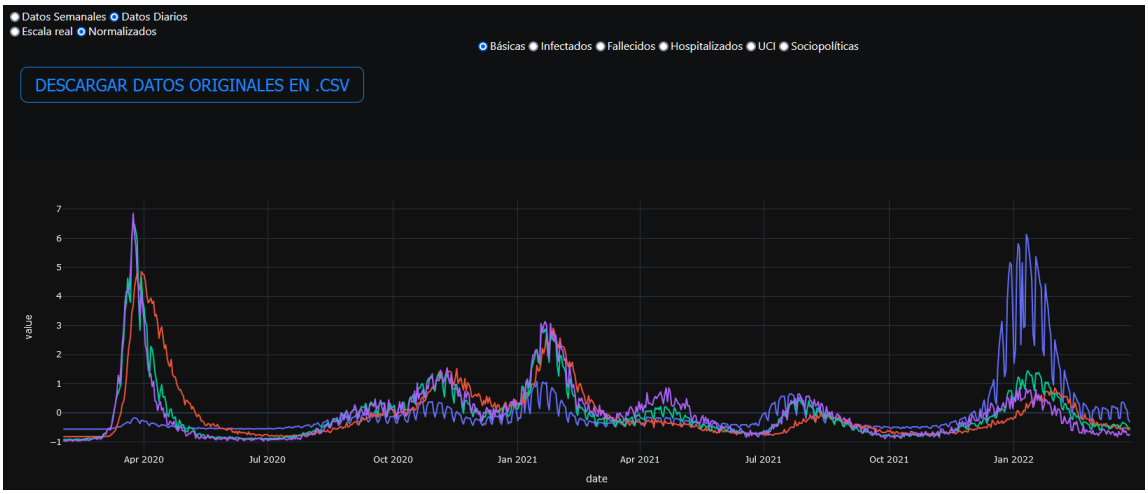

Figura 28: Selectores de opciones y grupos de variables

Finalmente, en la barra de la derecha se pueden activar o desactivar las series temporales correspondientes a cada variable para que se muestren en la gráfica.

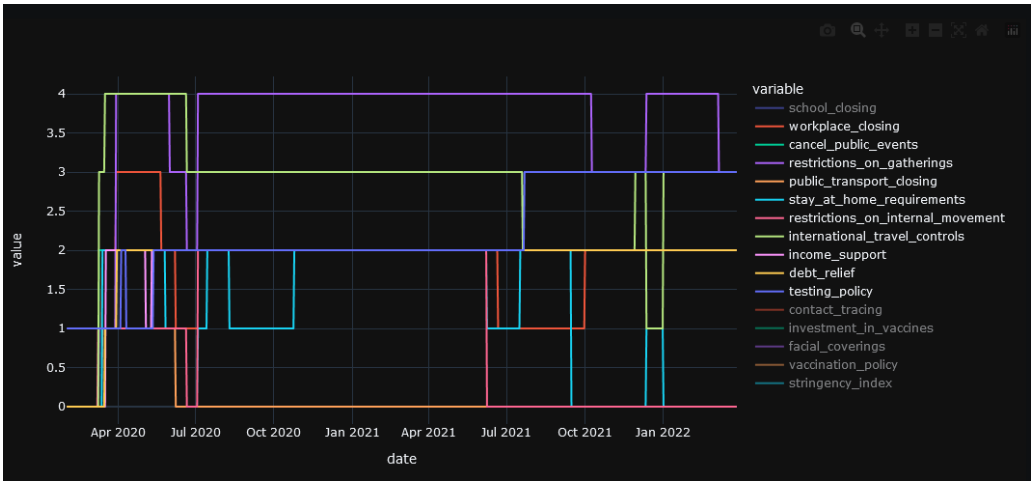

Figura 29: Activar o desactivar series temporales

Podemos interactuar con los ejes de la gráfica arrastrando para ajustar los rangos temporales. Si arrastramos de forma paralela en horizontal, ajustamos el rango temporal, También podemos ajustarlo a través del calendario selector. Si lo hacemos verticalmente se ajusta la escala del eje correspondiente al valor de las variables. En todo momento, podemos resetear la gráfica pulsando el botón de la casa en la parte superior derecha de la gráfica.

Además, podemos descargar los datos en formato CSV para analizarlos en profundidad con aplicaciones externas.

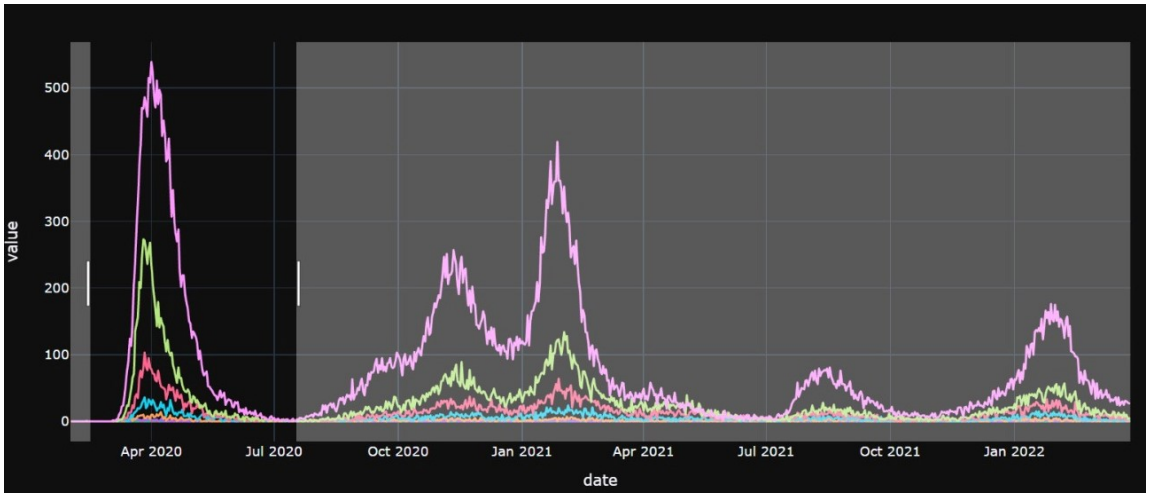

Figura 30: Selección de rango temporal arrastrando horizontalmente

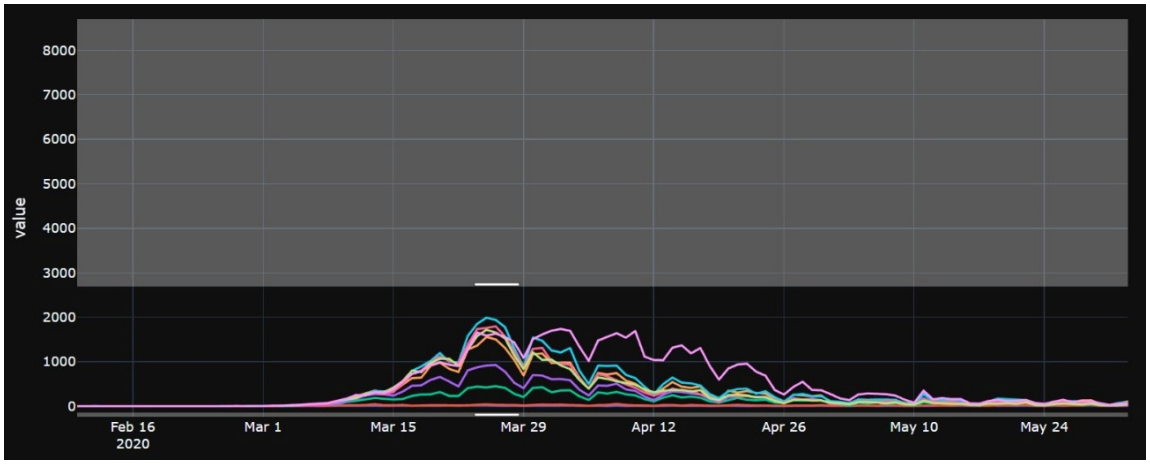

Figura 31: Selección de rango de las variables arrastrando verticalmente

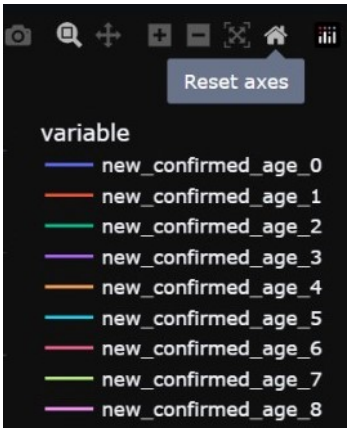

Figura 32: Botón de Reset de la gráfica

9.2 Aplicación del método

Para aplicar el método debemos seleccionar las opciones que deseemos de las descritas en el apartado anterior. El método se aplicará a todas las variables, en el rango temporal que marquen la gráfica y el calendario y de forma diaria o semanal según hayamos escogido. Para lanzar el método pulsamos el botón y se generan los resultados.

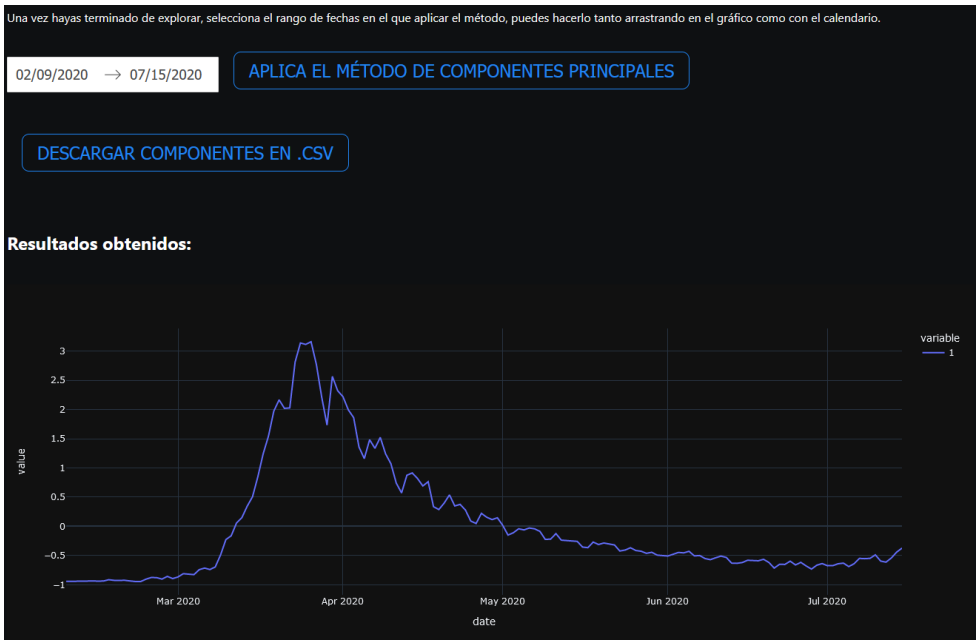

Figura 33: Aplicación del método

De nuevo, podemos explorar la componente en la gráfica o descargarlos mediante el botón correspondiente. También se muestra un breve informe con el valor propio principal y la varianza explicada que se acumularía si continuásemos aplicando el método en los siguientes espacios ortogonales.

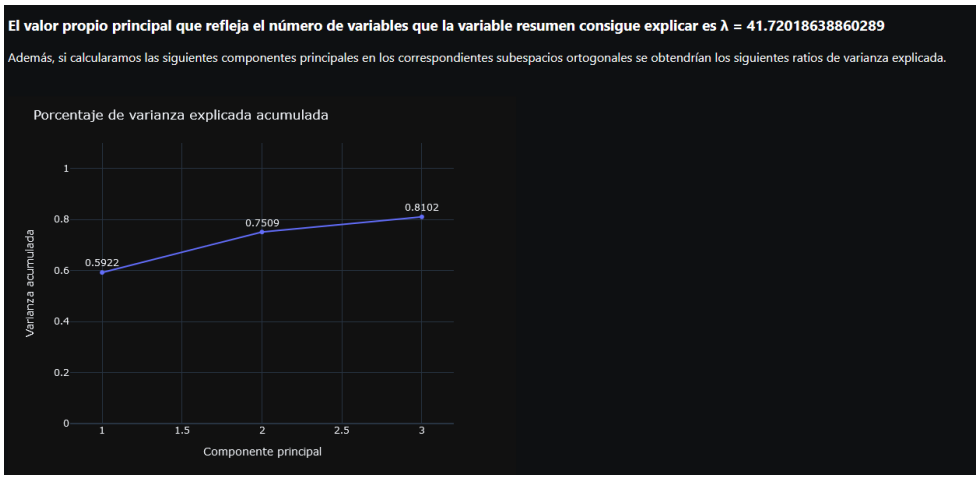

Figura 34: Informe de resultados

9.3 Comparación de variables

Una vez se ha aplicado el método, podemos comparar la componente principal con las distintas variables mediante el proceso descrito al final de la sección 5. Para ello utilizamos el selector y marcamos una variable. De esta forma se actualiza la gráfica con la componente principal dotada de las nuevas media y varianza y la otra variable en su estado original.

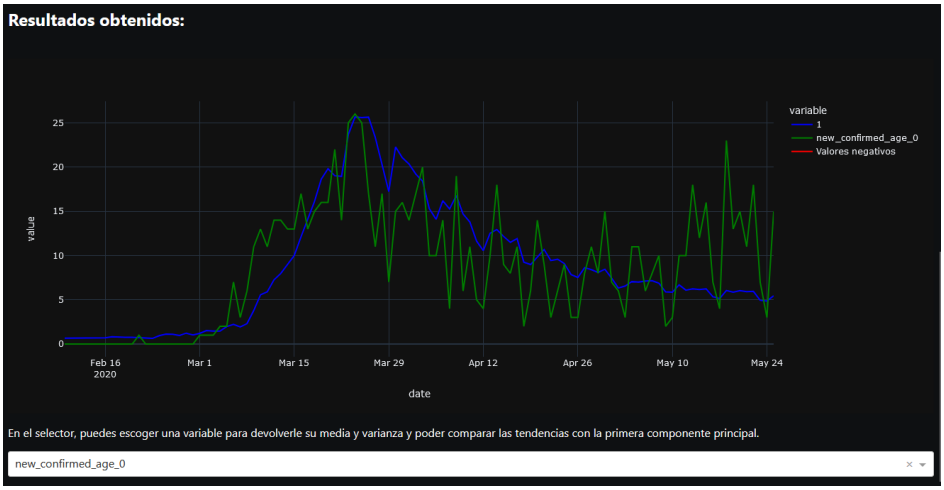

Figura 35: Comparación con una variable

Puede ocurrir en algunos casos, por las características de está técnica, que aún devolviendo la media y varianza a la componente, algunos valores continuen siendo negativos haciendo complicada la interpretación de, por ejemplo un supuesto número de muertos negativo. Esta característica no es subsanable sin variar la naturaleza del método, sin embargo no es problemática porque las tendencias continuan siendo comparables. Para denotarlo se han marcado en otro color en la gráfica estos valores negativos.

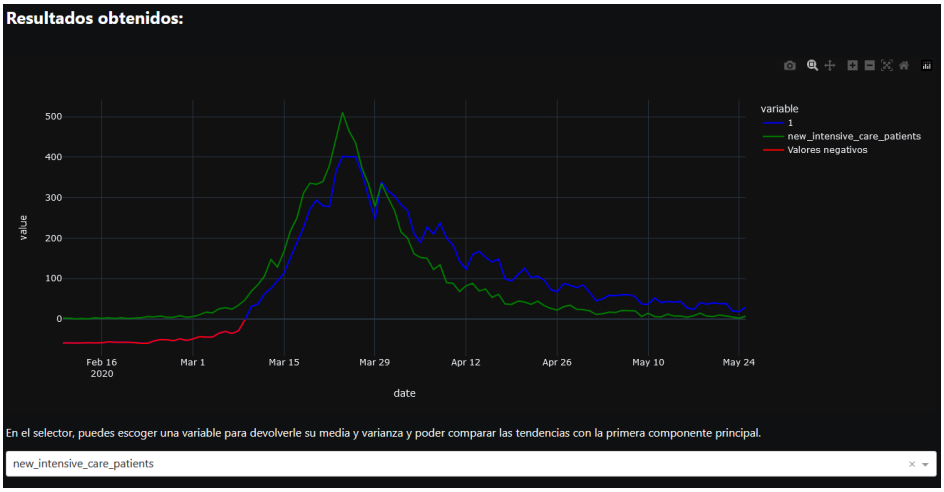

Figura 36: Comparación con variable con valores negativos en la componente

10 Despliegue en Google Cloud Platform

Para poder disponer de la aplicación en cualquier lugar y poder utilizarla para fines divulgativos hemos optado por desplegarla en un servidor web. Como no disponemos de una infraestructura local para hacerlo, utilizaremos un modelo de computación en la nube.

En la actualidad, la computación en la nube (cloud computing) se ha convertido en una parte fundamental de la infraestructura de muchas organizaciones. Google Cloud Platform (GCP) es la plataforma de servicios en la nube que ofrece Google, proporciona una amplia gama de herramientas y servicios para poder construir, implementar y escalar aplicaciones de manera eficiente. Uno de los servicios más destacados de GCP es App Engine, una plataforma de alojamiento y escalado de aplicaciones web.

Google Cloud App Engine [5] es un entorno de ejecución que permite a los desarrolladores crear y ejecutar aplicaciones web fácilmente en la infraestructura de Google sin necesidad de crear o configurar máquinas virtuales o configuraciones avanzadas de red. Con App Engine, nos podemos centrar en el desarrollo sin preocupaciones por la infraestructura subyacente, como servidores, redes o almacenamiento.

Las ventajas y características más significativas de App Engine son las siguientes:

- Escalabilidad: App Engine se encarga de escalar automáticamente la infraestructura de la aplicación en función del tráfico. Esto permite que se mantenga una alta disponibilidad incluso en momentos de alta carga. En todo momento la plataforma permite configurar los máximos de utilización para no elevar los costes de forma innecesaria.

- Facilidad de uso: App Engine proporciona un conjunto de API y servicios integrados que simplifican el desarrollo de la infraestructura para las aplicaciones. Permite aprovechar servicios de GCP como almacenamiento, instancias de bases de datos o autenticación de usuarios sin tener que configurarlos desde el principio.

- Balanceo de carga: En caso de que la aplicación este distribuida en varias instancias, App Engine distribuye automáticamente el tráfico entrante a las instancias de la aplicación, lo que garantiza una carga equilibrada y un rendimiento óptimo.

- Seguridad: Google cuenta con rigurosas medidas de seguridad en sus centros de datos, lo que proporciona a las aplicaciones que alojemos en App Engine una capa extra de protección.

- Monitorización y registro: GCP ofrece herramientas para ayudar a los desarrolladores a comprender y optimizar el rendimiento. Estas herramientas permiten rastrear métricas, registrar eventos y tratar de identificar cuellos de botella para mejorar la eficiencia.

El funcionamiento interno es sencillo, cuando una solicitud llega a una versión de la aplicación, App Engine activa instancias para servir esa solicitud. Las instancias son entornos de ejecución independientes que procesan las solicitudes de los usuarios. App Engine ajusta dinámicamente el número de instancias en función del tráfico y los parámetros de escalabilidad que hayamos configurado.

10.1 Procedimiento seguido

Una vez ha finalizado el desarrollo de la aplicación y tenemos el código preparado, necesitaremos preparar los ficheros necesarios para indicar de forma breve las dependencias necesarias y la infraestructura base a utilizar en GCP. Para guiarnos, hemos seguido el manual disponible en [6].

La estructura de ficheros es la siguiente:

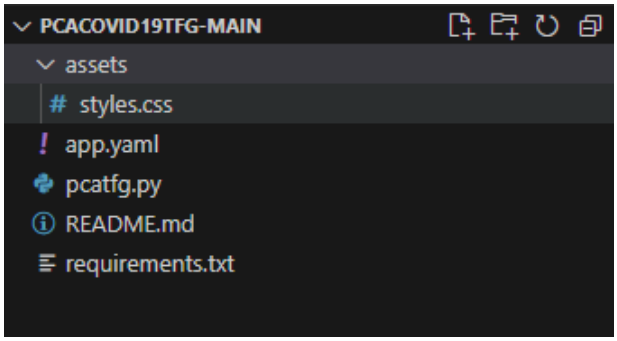

Figura 37: Estructura del directorio de la aplicación

En el fichero pcatfg.py se encuentra el código de la aplicación, en styles.css una hoja de estilos en cascada en la que se modifican algunos aspectos de estilo más avanzados de los que proporciona Dash. En el fichero requirements.txt indicamos las dependencias necesarias para ejecutar el código de la aplicación, incluyendo las librerías de Machine Learning y cálculo numérico que se han utilizado.

Por otro lado, necesitamos indicar a Google Cloud App Engine algunas características básicas. Usaremos su tipo de servicio por defecto y configuramos la versión de Python que queramos utilizar. Además necesitamos configurar las opciones máximas de escalabilidad, en nuestro caso lo limitaremos a 2 instancias que en caso de encontrarse ociosas (sin recibir peticiones) durante 10 minutos se desactivaran para ahorrar costes. Finalmente, especificamos los recursos hardware que ha de tener cada instancia. En nuestro caso es más que suficiente utilizar 1 CPU con 2 GB de RAM y 2 GB de tamaño de disco ya que la aplicación es muy ligera. Todo esto se codifica en el fichero app.yaml:

```
dash==2.8.1
pandas==1.3.4
numpy==1.20.3
scikit-learn==0.24.2
plotly==5.9.0
gunicorn
```

Figura 38: Fichero de dependencias requirements.txt

```
service: default
runtime: python37

basic_scaling:
    max_instances: 2
    idle_timeout: 10m

resources:
    cpu: 1
    memory_gb: 1
    disk_size_gb: 2

entrypoint: gunicorn -b 0.0.0.0:8080 pcatfg:server
```

Figura 39: Fichero de infraestructura app.yaml

Con todo esto listo, iniciamos sesión en la consola de los servicios de Google Cloud y creamos un proyecto destinado a la aplicación:

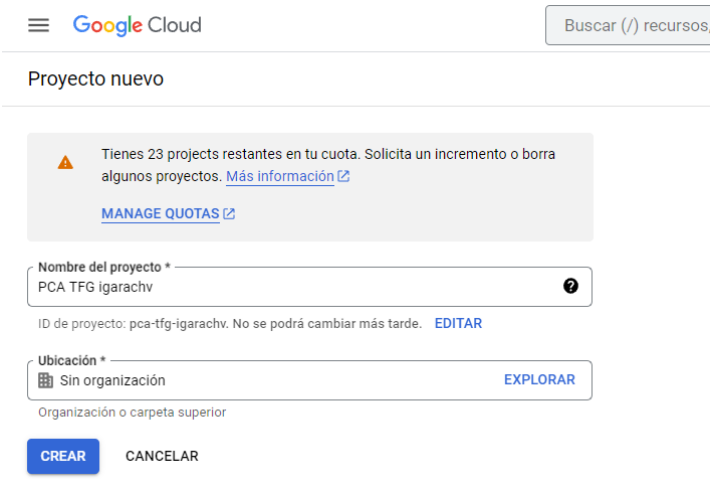

Figura 40: Creación de proyecto en GCP

En la barra de funcionalidades que ofrece la plataforma, bajamos hasta las opciones sin servidor (realmente lo gestiona de forma oculta) y seleccionamos App Engine:

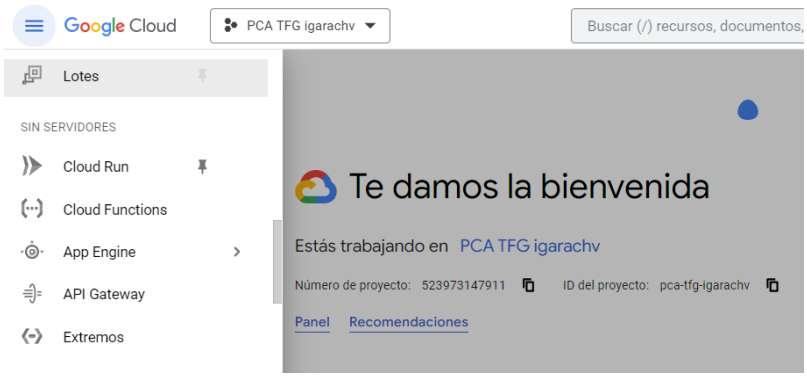

Figura 41: Opciones serverless

Dentro de App Engine, seleccionamos la opción de comenzar el despliegue y nos lleva a la siguiente pantalla en la que elegimos la región en la que se encontrará el centro de datos que gestionará la aplicación. Además, debe escogerse una cuenta de servicio de Google, por simplicidad usaremos la por defecto.

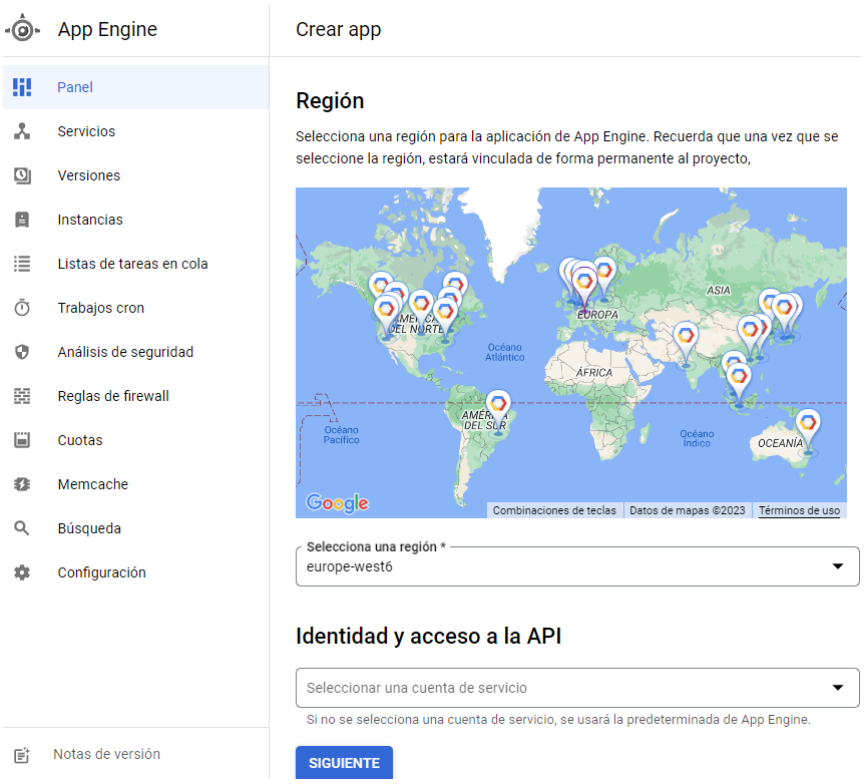

Figura 42: Configuración de región y cuenta de servicio

En la siguiente pantalla nos permite escoger el lenguaje de programación en el que está escrita la aplicación y la versión de App Engine que queramos usar.

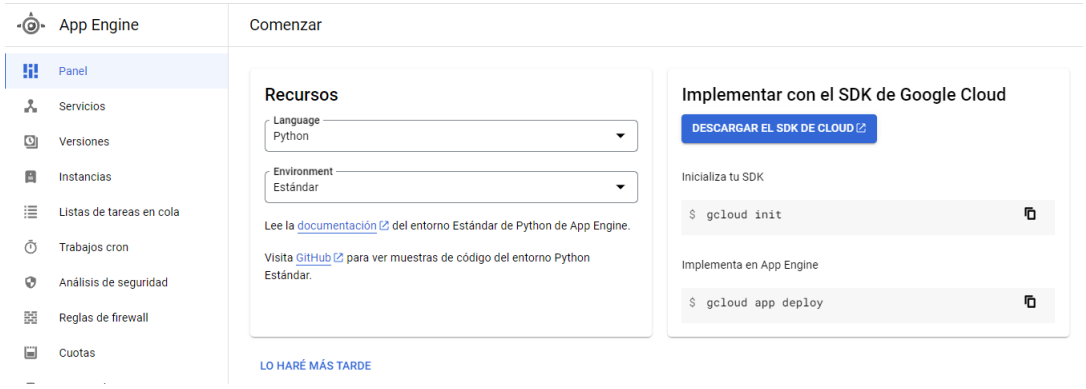

Figura 43: Selección de versión y lenguaje

Para continuar debemos activar la Cloud Shell del proyecto y subir los ficheros con nuestro código.

Figura 44: Activación de la Cloud Shell del proyecto

Podríamos hacerlo desde nuestra máquina si instalaramos la SDK de Google Cloud pero dada la sencillez de nuestro proyecto hemos optado por hacerlo en línea.

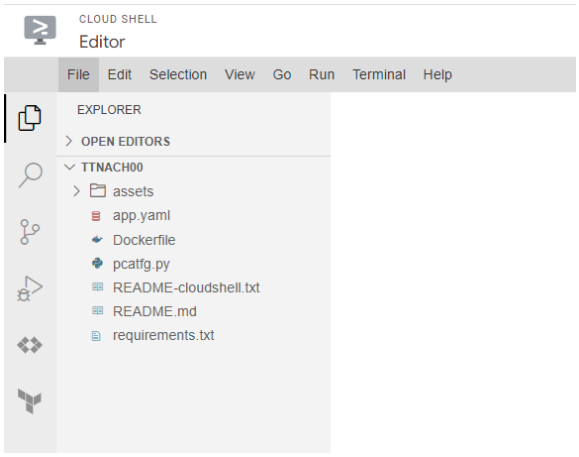

Figura 45: Subida de ficheros con el código implementado

Para terminar ejecutamos gcloud app deploy y aceptamos las preguntas que nos haga la terminal:

```
ttnach00@cloudshell:~ (pca-tfg-igarachv)$ gcloud app deploy
Services to deploy:

descriptor:                  [/home/ttnach00/app.yaml]
source:                      [/home/ttnach00]
target project:              [pca-tfg-igarachv]
target service:              [default]
target version:              [20230614t181401]
target url:                  [https://pca-tfg-igarachv.oa.r.appspot.com]
target service account:      [pca-tfg-igarachv@appspot.gserviceaccount.com]


Do you want to continue (Y/n)?
```

Figura 46: Despliegue

Al finalizar, nos muestra que se nos ha asignado la URL siguiente

https://pca-tfg-igarachv.oa.r.appspot.com

y que la aplicación está ejecutándose de acuerdo con las instrucciones indicadas en el fichero app.yaml.

Podemos acceder desde cualquier navegador web actual:

Figura 47: Acceso a la app desde Mozilla Firefox

11 Conclusiones y vías futuras

Los objetivos principales de el trabajo, que no eran otros que el desarrollo y la aplicación del método descrito a los datos de la pandemia en España han sido cumplidos. Con este método hemos conseguido extraer una alta variabilidad de los datos en una sóla variable aleatoria. Como posible vía de desarrollo futura, podríamos tomar esta variable y tratar de ajustarla con la curva de infectados del modelo SIR. Igualmente, se podría tomar como parte de un conjunto de entrenamiento mayor a la hora de realizar predicciones futuras mediante técnicas de aprendizaje automático, ya sea mediante un modelo autoregresivo ARIMA o alguno más complejo basado en redes neuronales o sistemas de decisión basados en árboles.

Además de esta limpieza de los datos, se ha realizado una breve discusión de los resultados obtenidos en las distintas olas. Se han comparado los resultados con algunas de las variables de partida y obtenido conclusiones sobre posibles desfases relacionados con el estado de gravedad de la epidemia y su posible reflejo en la variación de los datos de fallecidos. También se ha realizado un análisis comparativo con variables de movilidad basadas en densidad de dispositivos, que de una forma u otra reafirman que la variable extraída contiene también factores sociales, sobre como han variado los hábitos de la población durante las distintas olas.

Paralelamente, se ha llevado a cabo el desarrollo de una herramienta software para ejecutar el método con facilidad con diversas opciones. Se ha seguido una metodología de desarrollo incremental que ha permitido la introducción de nuevas características en todo momento según ha sido necesario. Finalmente, se han aprovechado las facilidades de los modelos de computación en la nube para desplegar la aplicación en la red.

Otra opción para desarrollos futuros que no ha podido completarse habría sido dotar de libertad al usuario final para utilizar sus propios datos. Así esta aplicación, podría utilizarse para otros ámbitos como la investigación de mercados, ámbito del cual surgió la idea para este trabajo en el libro de Teodoro Luque [12]. Incluso se podrían añadir opciones para trasladar algunas variables al futuro o al pasado y buscar correlaciones más fuertes entre los datos desfasados.

12 Anexo I: Implementación de la Aplicación Web

Por la simplicidad de la aplicación, se ha utilizado un sólo fichero de código que se pasa a mostrar, tiene 2 grandes bloques: las funciones propias de extracción y limpieza de datos y las funciones decoradas que controlan el sistema de callbacks.

12.1 Código Python y Scikit-Learn

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from statsmodels.tsa.seasonal import seasonal_decompose

def getCovidData():
"""
Recupera los datos de COVID-19 para España desde una fuente en línea
y los preprocesa.

Devuelve:
df (pd.DataFrame): Datos preprocesados de COVID-19 para España.
"""
df = pd.read_csv("https://storage.googleapis.com/covid19-open-data
/v3/location/ES.csv")
df = df[df['date'] > '2020-02-01']
df = df[df['date'] < '2022-03-28']
df.index = df['date']
mob = pd.read_csv("mobandtest.csv")
mob.index = df.index
df = pd.concat([df, mob], axis=1)
df.drop(["date"], axis=1, inplace=True)
df.reset_index(inplace=True)
return df

def cleanFull(df):
"""
Limpia los datos de COVID-19 eliminando columnas innecesarias
 y gestionando los valores faltantes.

Argumentos:
df (pd.DataFrame): Datos de COVID-19 de entrada.

Devuelve:
df (pd.DataFrame): Datos de COVID-19 limpios.
"""
df["new_persons_vaccinated"] = df["new_persons_vaccinated"].fillna(0.0)
df["new_persons_fully_vaccinated"] = df["new_persons_fully_vaccinated"].fillna(0.0)
df = df.dropna(axis=1)
df = df.drop(columns=['location_key', 'aggregation_level', 'place_id',
 'wikidata_id', 'datacommons_id','country_code', 'country_name',
 'iso_3166_1_alpha_2', 'iso_3166_1_alpha_3', 'age_bin_0',
 'age_bin_1', 'age_bin_2','age_bin_3', 'age_bin_4', 'age_bin_5',
 'age_bin_6', 'age_bin_7', 'age_bin_8',
 'population_density','human_development_index','population_age_00_09',
 'population_age_10_19','population_age_20_29','population_age_30_39',
 'population_age_40_49','population_age_50_59','population_age_60_69',
 'population_age_70_79','population_age_80_and_older','gdp_usd',
 'gdp_per_capita_usd','human_capital_index','openstreetmap_id',
 'latitude','longitude','area_sq_km','area_rural_sq_km','area_urban_sq_km',
 'life_expectancy','smoking_prevalence','diabetes_prevalence',
 'infant_mortality_rate','adult_male_mortality_rate',
```

```
 'adult_female_mortality_rate','pollution_mortality_rate',
 'comorbidity_mortality_rate','nurses_per_1000','physicians_per_1000',
 'population','population_male','population_female','population_rural',
 'population_urban','population_largest_city','population_clustered',
 'population_density','human_development_index','health_expenditure_usd',
 'out_of_pocket_health_expenditure_usd', 'public_information_campaigns'])
 df= df.drop(columns=['cumulative_confirmed', 'cumulative_deceased',
 'cumulative_hospitalized_patients', 'cumulative_intensive_care_patients',
 'cumulative_confirmed_age_0', 'cumulative_confirmed_age_1',
 'cumulative_confirmed_age_2', 'cumulative_confirmed_age_3',
 'cumulative_confirmed_age_4', 'cumulative_confirmed_age_5',
 'cumulative_confirmed_age_6', 'cumulative_confirmed_age_7',
 'cumulative_confirmed_age_8', 'cumulative_deceased_age_0',
 'cumulative_deceased_age_1', 'cumulative_deceased_age_2',
 'cumulative_deceased_age_3', 'cumulative_deceased_age_4',
 'cumulative_deceased_age_5', 'cumulative_deceased_age_6',
 'cumulative_deceased_age_7', 'cumulative_deceased_age_8',
 'cumulative_hospitalized_patients_age_0', 'cumulative_hospitalized_patients_age_1',
 'cumulative_hospitalized_patients_age_2', 'cumulative_hospitalized_patients_age_3',
 'cumulative_hospitalized_patients_age_4', 'cumulative_hospitalized_patients_age_5',
 'cumulative_hospitalized_patients_age_6', 'cumulative_hospitalized_patients_age_7',
 'cumulative_hospitalized_patients_age_8', 'cumulative_intensive_care_patients_age_0',
 'cumulative_intensive_care_patients_age_1', 'cumulative_intensive_care_patients_age_2',
 'cumulative_intensive_care_patients_age_3', 'cumulative_intensive_care_patients_age_4',
 'cumulative_intensive_care_patients_age_5', 'cumulative_intensive_care_patients_age_6',
 'cumulative_intensive_care_patients_age_7', 'cumulative_intensive_care_patients_age_8',
 'cumulative_confirmed_male', 'cumulative_confirmed_female',
 'cumulative_deceased_male','cumulative_deceased_female',
 'cumulative_hospitalized_patients_male', 'cumulative_hospitalized_patients_female',
 'cumulative_intensive_care_patients_male', 'cumulative_intensive_care_patients_female'])
return df


def normalize01(df):
"""
Normaliza los datos en el rango [0, 1] utilizando el escalador estándar.
Args:
df (pd.DataFrame): Datos de entrada.

Returns:
normalized (pd.DataFrame): Datos normalizados.
norm (StandardScaler): Objeto de escalado utilizado para la normalización.
"""
norm = StandardScaler()
normalized = pd.DataFrame(norm.fit_transform(df), columns=df.columns)
return normalized, norm


def normalizePC(df):
"""
Realiza la normalización de las componentes principales resultado,
paso previo a poder dotarlas de varianza y comparar.
Args:
df (pd.DataFrame): Datos de entrada.

Returns:
normalized (pd.DataFrame): Datos normalizados.
"""
preprocesado = df.drop(columns=["date"])
normalized, otra = normalize01(preprocesado)
normalized['date'] = df['date'].to_numpy()
return normalized


def correlationMatrix(array):
"""
Calcula y muestra la matriz de correlación.
```

```
    Args:
    array (ndarray): Array de datos.

    Returns:
    None
    """
    df = pd.DataFrame(array)
    correlations = df.corr(method='pearson')
    figure = plt.figure()
    axes = figure.add_subplot(111)
    caxes = axes.matshow(correlations)
    figure.colorbar(caxes)


def applyPCA(df):
    """
    Aplica el análisis de componentes principales (PCA) a los datos.
    Args:
    df (pd.DataFrame): Datos de entrada.

    Returns:
    comps (pd.DataFrame): Componentes principales resultantes.
    pca (PCA): Objeto PCA utilizado para la transformación.
    otra: Variable auxiliar con los datos de la normalización previa.
    """
    preprocesado = df.drop(columns=["date"])
    normalizado, otra = normalize01(preprocesado)
    pca = PCA(3)
    pc = pca.fit_transform(normalizado)

    comps = pd.DataFrame(pc, columns=np.arange(1, 4))
    comps['date'] = df['date'].to_numpy()
    return comps, pca, otra


def showCumulativeVariance(pca):
    """
    Muestra la varianza explicada acumulada por cada componente principal.
    Args:
    pca (PCA): Objeto PCA de Scikit Learn.
    Returns:
    fig: Gráfico de línea que muestra la varianza explicada acumulada.
    """
    prop_varianza_acum = pca.explained_variance_ratio_.cumsum()
    ejex = np.arange(pca.n_components_) + 1
    ejey = prop_varianza_acum
    ejey = [round(y, 4) for y in ejey]

    fig = px.line(x=ejex, y=ejey, text=ejey)

    fig.update_layout(title="Porcentaje_de_varianza_explicada_acumulada",
    xaxis_title="Componente_principal",
    yaxis_title="Varianza_acumulada")
    fig.update_xaxes(range=[0.8, pca.n_components_ + 0.2])
    fig.update_yaxes(range=[0, 1.1])
    fig.update_traces(textposition="top_center")
    return fig

def getSmoothTrendBasic(df, days):
    """
    Obtiene la tendencia suavizada básica de los datos.
    Args:
    df (pd.DataFrame): Datos de entrada.
    days (int): Número de días para agrupar los datos.

    Returns:
    newdf (pd.DataFrame): Datos con la tendencia suavizada.
```

```
    """
    newdf = df.copy()
    newdf['date'] = pd.to_datetime(newdf['date'])
    newdf = newdf.groupby(pd.Grouper(key='date', freq=str(days)+'D')).sum().reset_index()
    return newdf


def makeVisualization(data):
    """
    Crea visualizaciones utilizando la biblioteca Plotly Express.

    Args:
    data (pd.DataFrame): Datos de entrada.

    Returns:
    base, confirmed, deceased, hospitalized, uci, sp (plotly.graph_objs._figure.Figure):
     Visualizaciones de los distintos grupos de variables.
    """
    base = px.line(data, x="date", y=["new_confirmed", "new_deceased", "new_hospitalized_patients",
    "new_intensive_care_patients"])
    confirmed = px.line(data, x="date", y=["new_confirmed_age_0", "new_confirmed_age_1",
    "new_confirmed_age_2", "new_confirmed_age_3", "new_confirmed_age_4",
    "new_confirmed_age_5", "new_confirmed_age_6", "new_confirmed_age_7",
    "new_confirmed_age_8"])
    deceased = px.line(data, x="date", y=["new_deceased_age_0", "new_deceased_age_1",
    "new_deceased_age_2", "new_deceased_age_3", "new_deceased_age_4",
    "new_deceased_age_5", "new_deceased_age_6", "new_deceased_age_7",
    "new_deceased_age_8"])
    hospitalized = px.line(data, x="date", y=["new_hospitalized_patients_age_0",
    "new_hospitalized_patients_age_1", "new_hospitalized_patients_age_2",
    "new_hospitalized_patients_age_3", "new_hospitalized_patients_age_4",
    "new_hospitalized_patients_age_5", "new_hospitalized_patients_age_6",
    "new_hospitalized_patients_age_7", "new_hospitalized_patients_age_8"])
    uci = px.line(data, x="date", y=["new_intensive_care_patients_age_0",
    "new_intensive_care_patients_age_1", "new_intensive_care_patients_age_2",
    "new_intensive_care_patients_age_3", "new_intensive_care_patients_age_4",
    "new_intensive_care_patients_age_5", "new_intensive_care_patients_age_6",
    "new_intensive_care_patients_age_7", "new_intensive_care_patients_age_8"])
    sp = px.line(data, x="date", y=["school_closing", "workplace_closing",
    "cancel_public_events", "restrictions_on_gatherings", "public_transport_closing",
    "stay_at_home_requirements", "restrictions_on_internal_movement",
    "international_travel_controls", "income_support", "debt_relief", "testing_policy",
    "contact_tracing", "investment_in_vaccines", "facial_coverings", "vaccination_policy",
    "stringency_index"])
    return base, confirmed, deceased, hospitalized, uci, sp
```

12.2 Funciones decoradas y callbacks

```
import matplotlib.pyplot as plt
import plotly.express as px
import dash
from dash import dcc
from dash import html
from datetime import date
import plotly.express as px
import plotly.io as pio
import pandas as pd
import time

df = getCovidData()
df = cleanFull(df)
newdf = getSmoothTrendBasic(df, 7)
pio.templates.default = "plotly_dark"
df['date'] = pd.to_datetime(df['date'])
variables = [a for a in df.columns]
variables.pop(0)


# Definir la función de callback para actualizar el rango
@app.callback(
dash.dependencies.Output('picker-range', 'start_date'),
dash.dependencies.Output('picker-range', 'end_date'),
dash.dependencies.Input('graph', 'relayoutData'),
prevent_initial_call=True
)
def update_range(relayout_data):
        if relayout_data is None:
                start_date = df['date'].min()
                end_date = df['date'].max()
        else:
                if 'xaxis.range[0]' in relayout_data and 'xaxis.range[1]' in relayout_data:
                        start_date = pd.to_datetime(relayout_data['xaxis.range[0]'])
                        end_date = pd.to_datetime(relayout_data['xaxis.range[1]'])
        else:
                if 'xaxis.autorange' in relayout_data:
                        start_date = df['date'].min()
                        end_date = df['date'].max()
                else:
                        raise dash.exceptions.PreventUpdate

        return start_date.date(), end_date.date()
```

```
# Definir la función de callback para actualizar el gráfico en función de las opciones
@app.callback(
dash.dependencies.Output('graph', 'figure'),
dash.dependencies.Input('picker-range', 'start_date'),
dash.dependencies.Input('picker-range', 'end_date'),
dash.dependencies.Input('radius', 'value'),
dash.dependencies.Input('radsem', 'value'),
dash.dependencies.Input('radnorm', 'value'),
prevent_initial_call=True
)
def update_initial_graph(d1, d2, value, sem, norm):
        if norm == "Escala real":
                if sem == "Datos Diarios":
                        if value == "Básicas":
                                fig = base
                        if value == "Infectados":
                                fig = confirmed
                        if value == "Fallecidos":
                                fig = deceased
                        if value == "Hospitalizados":
                                fig = hospitalized
                        if value == "UCI":
                                fig = uci
                        if value == "Sociopolíticas":
                                fig = sp
                else:
                        if value == "Básicas":
                                fig = base7
                        if value == "Infectados":
                                fig = confirmed7
                        if value == "Fallecidos":
                                fig = deceased7
                        if value == "Hospitalizados":
                                fig = hospitalized7
                        if value == "UCI":
                                fig = uci7
                        if value == "Sociopolíticas":
                                fig = sp7
        else:
                if sem == "Datos Diarios":
                        if value == "Básicas":
                                fig = basen
                        if value == "Infectados":
                                fig = confirmedn
                        if value == "Fallecidos":
                                fig = deceasedn
                        if value == "Hospitalizados":
                                fig = hospitalizedn
                        if value == "UCI":
                                fig = ucin
                        if value == "Sociopolíticas":
                                fig = spn
                else:
                        if value == "Básicas":
                                fig = base7n
                        if value == "Infectados":
                                fig = confirmed7n
                        if value == "Fallecidos":
                                fig = deceased7n
                        if value == "Hospitalizados":
                                fig = hospitalized7n
                        if value == "UCI":
                                fig = uci7n
                        if value == "Sociopolíticas":
                                fig = sp7n
        fig.update_layout(xaxis_range=[d1, d2])
        return fig
```

```
#Función que aplica el método de componentes principales y genera un informe
@app.callback(
dash.dependencies.Output("components-graph", "children"),
dash.dependencies.Output("intermediate-value", "data"),
dash.dependencies.Output("intermediate-value_stats", "data"),
dash.dependencies.Output("btn-download-2", "style"),
dash.dependencies.Input("applymethod", "n_clicks"),
dash.dependencies.State("components-graph", "children"),
dash.dependencies.State("picker-range", "start_date"),
dash.dependencies.State("picker-range", "end_date"),
dash.dependencies.State('radsem', 'value'),
prevent_initial_call=True
)
def apply_method(n_clicks, children, ini, fin, period):
        if n_clicks is None:
                raise dash.exceptions.PreventUpdate
        else:
                if period == "Datos Semanales":
                        dfauxiliar = newdf.copy()
                else:
                        dfauxiliar = df.copy()

                dfauxiliar = dfauxiliar[dfauxiliar['date'] >= ini]
                dfauxiliar = dfauxiliar[dfauxiliar['date'] <= fin]

                pc, stats, estadisticas = applyPCA(dfauxiliar)

                a = pd.DataFrame(estadisticas.feature_names_in_, columns=["feature"])
                b = pd.DataFrame(estadisticas.mean_, columns=["mean"])
                c = pd.DataFrame(estadisticas.var_, columns=["variance"])
                dfstats = pd.concat((a, b, c), axis=1).set_index("feature")

                normalizado, aux = normalize01(dfauxiliar.drop(columns=['date']))
                pcnormal = normalizePC(pc)
                pcwithnormvariables = pd.concat([pcnormal, normalizado], axis=1)

                full = px.line(pcwithnormvariables, x="date", y=[1])
                cv = showCumulativeVariance(stats)

                if children:
                        children[1]["props"]["figure"] = full
                        children[4] = html.H3(children='El valor propio principal que refleja
                        el número de variables que la variable resumen consigue explicar es
                         lambda = '     +str(stats.explained_variance_[0]))
                        children[6]["props"]["figure"] = cv
                else:
                        children.append(html.H2(children='Resultados obtenidos:'))
                        children.append(dcc.Graph(figure=full, id='full-graph'))
                        children.append(html.P(children='En el selector, puedes escoger
                        una variable para devolverle su media y varianza
                        y poder comparar las tendencias con la primera componente principal.'))
                        children.append(dcc.Dropdown(variables, multi=False,
                        placeholder="Seleccione una variable para comparar", id='dropdown'))
                        children.append(html.H3(children='El valor propio principal que refleja
                        el número de variables que la variable resumen consigue explicar
                        es $\lambda$
                        = ' +str(stats.explained_variance_[0])))
                        children.append(html.P(children='Además, si calcularamos las siguientes
                        componentes principales en los correspondientes subespacios ortogonales
                        se obtendrían los siguientes ratios de varianza explicada.'))
                        children.append(dcc.Graph(figure=cv, id='cumulative',
                        style={'width': '50%'}))
    return children, pcwithnormvariables.to_json(date_format='iso', orient='split'),
    dfstats.to_json(date_format='iso', orient='split'), {'visibility':'visible'}
```

```
#Función que actualiza la gráfica de comparación de variables dotando a la componente
principal de la varianza de la variable con la que se la compara.
@app.callback(
dash.dependencies.Output('full-graph', 'figure'),
dash.dependencies.Input('dropdown', 'value'),
dash.dependencies.State("components-graph", "children"),
dash.dependencies.State("intermediate-value", "data"),
dash.dependencies.State("intermediate-value_stats", "data"),
prevent_initial_call=True
)
def update_initial_graph(valores, children, jsonified_cleaned_data, jsonified_stats):
    readdf = pd.read_json(jsonified_cleaned_data, orient='split')
    stats = pd.read_json(jsonified_stats, orient='split')
    readdf[1] = readdf[1] * np.sqrt(stats.loc[valores]["variance"]) +
    stats.loc[valores]["mean"]
    readdf[valores] = readdf[valores] * np.sqrt(stats.loc[valores]["variance"]) +
    stats.loc[valores]["mean"]
    columns = [1, valores]
    full = px.line(readdf, x="date", y=columns, color_discrete_sequence=["blue", "green"])
    children[0]["props"]["figure"] = full
    full.add_scattergl(x=readdf["date"], y=readdf[1].where(readdf[1] < 0), line={'color':
    'red'}, name="Valores negativos")
    return children[0]["props"]["figure"]


#Funciones que activan la descarga de los Dataframes en formato CSV
@app.callback(
dash.dependencies.Output('download-dataframe-csv', 'data'),
dash.dependencies.Input('btn-download-1', 'n_clicks'),
prevent_initial_call=True
)
def download_original_data(n_clicks):
    return dcc.send_data_frame(df.set_index('date').to_csv, "cleaned_data.csv")


@app.callback(
dash.dependencies.Output('download-components-csv', 'data'),
dash.dependencies.Input('btn-download-2', 'n_clicks'),
dash.dependencies.State("intermediate-value", "data"),
prevent_initial_call=True
)
def download_comp_data(n_clicks, jsonified_cleaned_data):
    readdf = pd.read_json(jsonified_cleaned_data, orient='split')
    return dcc.send_data_frame(readdf.set_index('date')[1].to_csv, "components.csv")


# Ejecutar la aplicación
if __name__ == '__main__':
app.run(debug=True)
```

Referencias

[1] Alfaro-Martínez, J. J., García del Pozo, J. S., et al. (2023). Estudio de la incidencia de COVID-19 en España y su relación geográfica provincial. Journal of Healthcare Quality Research.

[2] Cleveland, R. B., Cleveland, W. S., et al. (1990). STL: A Seasonal-Trend Decomposition Procedure Based on LOESS. Journal of Official Statistics, 6, 3-73.

[3] El Ghaoui, Spectral Theorem: Eigenvalue Decomposition for Symmetric Matrices en L. Optimization Models and Applications. Disponible en: https://ecampusontario.pressbooks.pub/optimizationmodelsandapplctns. Accedido el 2023-04-12.

[4] Fernández-Granda, C. (2019). Principal Component Analysis. En Mathematical Tools for Data Science. NYU's Center for Data Science. Spring 2019.

[5] Google. (2023). Documentación de Google Cloud App Engine. Disponible en: https://cloud.google.com/appengine/docs?hl=es-419. Accedido el 2023-06-11.

[6] Hodge, M., UK ONS Data Science Campues. (2022). Guía de despliegue de Dash en GCP. Disponible en:https://datasciencecampus.github.io/deploy-dash-with-gcp/. Accedido el 2023-06-11.

[7] Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology, 24(6), 417–441.

[8] Jolliffe, I. T. (2002). Principal Component Analysis. Springer Series in Statistics. Springer, New York, NY, 1, 1-6.

[9] Kermack, W., McKendrick, A. (1991). Contributions to the mathematical theory of epidemics – I. Bulletin of Mathematical Biology, 53(1–2), 33–55.

[10] Kolmogorov, A. N. (1933). Foundations of the theory of probability. New York: Chelsea Pub. Co, 1, 2-3.

[11] López Camino, R. (2014). Topología. Editorial Universidad de Granada, 266-267.

[12] Luque Martínez, T. (2000). Técnicas de análisis de datos en investigación de Mercados. Editorial Pirámide, 2, 40-58.

[13] Marschner, I. C. (2021). Estimating age-specific COVID-19 fatality risk and time to death by comparing population diagnosis and death patterns: Australian data. BMC Med Res Methodol, 21, 126.

[14] Mathieu, E., Ritchie, H., et al. (2020). Coronavirus Pandemic Data (COVID-19). Repositorio Covid-19 OWID. Disponible en: https://ourworldindata.org/coronavirus. Accedido el 2022-12-16.

[15] Merino, L., Santos, E. (2006). Álgebra Lineal con métodos elementales. Ediciones Paraninfo, 4-5, 203-293.

[16] Ortega, J. M. (1987). Matrix Theory: A Second Course. Springer-Verlag US, 1, 29-35.

[17] Ortega Ríos, R. (s.f.). Teorema de Perron Frobenius. Apuntes para Modelos Matemáticos I. Disponible en: https://www.ugr.es/~rortega/PDFs/TPF.pdf. Accedido el 2023-05-29.

[18] Ortiz, M., Modelos de Desarrollo de Software. . Disponible en: http://isw-udistrital.blogspot.com/2012/09/ingenieria-de-software-i.html. Accedido el 2023-04-06.

[19] Pandas. (2023). Guía de usuario. Disponible en: https://pandas.pydata.org/docs/. Accedido el 2023-03-19.

[20] Pastor, O. (2001). An Object-Oriented Approach for Web-Solutions Modeling. Proceedings Media In Information Society, 127–130.

[21] Pearson, K. (1901). LIII. On lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 2(11), 559-572.

[22] Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. JMLR, 12, 2825-2830.

[23] Petrov, V. V., Mordecki, E. (2002). Teoría de la Probabilidad. Editorial URSS, 1 y 3-4, 13-107.

[24] Plotly. (2023). Guía de usuario de la librería Dash para Python. Disponible en: https://dash.plotly.com . Accedido el 2023-03-19.

[25] Ponce Campuzano, J. C. (s.f.). Calculadora de curvas del modelo SIR en GEOGEBRA. Disponible en: https://www.geogebra.org/m/ymwxkyna. Accedido el 2023-06-18.

[26] Pressman, R. (2010). Software Engineering: A Practitioner's Approach. Boston: McGraw Hill. 48–49.

[27] Santillán-García, A., et al. (2020). Es hora de aceptar que el SARS-CoV-2 se transmite por aerosoles y actuar en consecuencia. Index Enferm, 29(4), 262. Disponible en: http://scielo.isciii.es/scielo.php?script=sci_arttext&pid=S1132-12962020000300016&lng=es. Accedido el 2023-06-15.

[28] Varea-Jiménez, E., et al. (2022). Comparative severity of COVID-19 cases caused by Alpha, Delta or Omicron SARS-CoV-2 variants and its association with vaccination. Enfermedades Infecciosas y Microbiología Clínica, 1-3.

[29] Wahltinez, O., et al. (2020). COVID-19 Google Open-Data: curating a fine-grained, global-scale data repository for SARS-CoV-2. Repositorio Covid-19 Open Data. Disponible en: https://github.com/GoogleCloudPlatform/covid-19-open-data/tree/main . Accedido el 2022-12-15.

[30] Weiss, H. H. (2013). The SIR model and the foundations of public health. Materials Mathematics UAB, Georgia Institute of Technology, 2-4.