



Facultad de ciencias económicas y empresariales ICADE

Alimentación sostenible: Análisis de la eficiencia de mercado mediante el uso del MF-DFA

Autor: Gabriel Domínguez Lázaro

Clave: 202001808

Director: Alejandro Rodríguez Gallego

(E2+Business Analytics)

Madrid | Junio 2025

Índice

1.-Introducción.....	6
2.-Objetivo	7
3.-Metodología	8
4.- Alimentación sostenible	9
4.1.- <i>Criterios ESG</i>	11
4.2.- <i>ODS</i>	13
5.-Marco teórico.....	14
5.1.- <i>Efficient Market Hypothesis</i>	14
5.2.- <i>DFA</i>	15
5.3.- <i>MF-DFA</i>	17
6.-Base de datos.....	21
7.-Análisis	26
7.1.- <i>Impacto del COVID19 en los sectores de la alimentación</i>	28
7.2.- <i>Análisis de la caída de los índices de alimentación sostenible a partir de 2021</i>	29
7.3.- <i>DFA</i>	30
7.4.- <i>MF-DFA</i>	34
8.-Conclusión.....	44
9.-Bibliografía	46
10.- Anexos.....	52
10.1.- <i>Métricas MF-DFA</i>	52
11.-Declaración de Uso de Herramientas de Inteligencia Artificial Generativa en Trabajos Fin de Grado	55
12.- Código de Python	57

Ilustración 1: Grafico de líneas que representa la evolución desde el 2005 hasta el 2023 del porcentaje y el número de personas que padecen desnutrición. Fuente: elaboración propia a partir de datos del informe “The state of food security and nutrition in the world” FAO,2024	6
Ilustración 2: Distribución sectorial de las compañías que componen el índice MSCI. Fuente: elaboración propia a partir de información obtenida de la "Index factheet", (MSCI Inc., 2025).....	22
Ilustración 3: Distribución geográfica de las compañías que componen el índice MSCI. Fuente: elaboración propia a partir de información obtenida de la "Index factheet", (MSCI Inc., 2025).....	22
Ilustración 4: Distribución geográfica de las compañías que componen el índice MVIS. Fuente: elaboración propia a partir de información obtenida de la "Index factheet", (MarketVector Indexes GmnH, 2025).....	24
Ilustración 5: Distribución sectorial de las compañías que componen el índice MVIS. Fuente: elaboración propia a partir de información obtenida de la "Index factheet", (MarketVector Indexes GmnH, 2025).....	24
Ilustración 6: Retornos acumulados de los índices MSCI, FXBYFOOD, MVIS y el SP500. Fuente: elaboración propia.....	27
Ilustración 7: Evolución del coeficiente de Hurst, obtenido mediante DFA, del índice MVIS. Fuente: elaboración propia	31
Ilustración 8: Evolución del coeficiente de Hurst, obtenido mediante DFA, del índice MSCI. Fuente: elaboración propia.....	31
Ilustración 9: Evolución del coeficiente de Hurst, obtenido mediante DFA, del índice FXBYFOOD. Fuente: elaboración propia.....	32
Ilustración 10: Evolución del coeficiente de Hurst, obtenido mediante DFA, del índice SP500. Fuente: elaboración propia.....	32
Ilustración 11: Resultados MF-DFA sobre los índices MSCI, FXBYFOOD (alimentación sostenible), SP500 y fGn. Periodo: pre-COVID19.	38
Ilustración 12: Resultados MF-DFA sobre los índices MSCI, FXBYFOOD (alimentación sostenible), SP500 y fGn. Periodo: pre-COVID19 shuffled.	38
Ilustración 13: Resultados MF-DFA sobre los índices MSCI, FXBYFOOD, MVIS (alimentación sostenible), SP500 y fGn. Periodo: post-COVID19 shuffled.....	39
Ilustración 14: Resultados MF-DFA sobre los índices MSCI, FXBYFOOD, MVIS (alimentación sostenible), SP500 y fGn. Periodo: post-COVID19.....	39

Ilustración 15: Comparativa del índice MSCI entre el primer y segundo periodo.....	41
Ilustración 16: Comparativa del índice FXBYFOOD entre el primer y segundo periodo.	41
Ilustración 17: Comparativa del índice SP500 entre el primer y segundo periodo	41
Ilustración 20: Comparativa del índice FXBYFOOD normal y shuffled en el primer periodo.	42
Ilustración 18: Comparativa del índice SP500 normal y shuffled en el primer periodo.	42
Ilustración 19: Comparativa del índice MSCI normal y shuffled en el primer periodo.	42
Ilustración 21: Comparativa del índice SP500 normal y shuffled en el segundo periodo.	43
Ilustración 22: Comparativa del índice MVIS normal y shuffled en el segundo periodo.	43
Ilustración 23: Comparativa del índice MSCI normal y shuffled en el segundo periodo.	43
Ilustración 24: Comparativa del índice FXBYFOOD normal y shuffled en el segundo periodo.	43
 Tabla 1: Análisis financiero de rentabilidad anualizada, volatilidad anualizada y Sharpe Ratio por índice, MSCI, FXYFOOD, MVIS, SP500. Fuente: elaboración propia a partir de los retornos de los índices.....	26
Tabla 2: Resultados del MF-DFA sobre los índices MSCI, MVIS, FXBYFOOD, SP500 y fGn por periodos pre-COVID19 y post-COVID19. Fuente: elaboración propia a partir de los resultados obtenidos del análisis MF-DFA.	37
Tabla 3: Métricas análisis fGn. Fuente: elaboración propia.	52
Tabla 4: Métricas análisis MSCI, pre-COVID19. Fuente: elaboración propia.	52
Tabla 5: Métricas análisis FXBYFOOD, pre-COVID19. Fuente: elaboración propia. .	52
Tabla 6: Métricas análisis SP500, pre-COVID19. Fuente: elaboración propia.....	52
Tabla 7: Métricas análisis MSCI, post-COVID19. Fuente: elaboración propia.....	53
Tabla 8: Métricas análisis FXBYFOOD, post-COVID19. Fuente: elaboración propia.	53
Tabla 9: Métricas análisis MVIS, post-COVID19. Fuente: elaboración propia.	53
Tabla 10: Métricas análisis SP500, post-COVID19. Fuente: elaboración propia.	53

Resumen

Este estudio analiza cómo la pandemia de COVID-19 afectó la eficiencia de mercado de cuatro índices (tres ESG de alimentación sostenible y el *SP500*) usando análisis monofractal y multifractal (MF-DFA) sobre series originales y *shuffled*.

Antes de la crisis, todos los índices mostraron eficiencia intermedia con multifractalidad moderada ligada a la memoria temporal. Tras el COVID-19, los índices sostenibles perdieron parte de esa eficiencia y presentaron mayor complejidad multifractal (reflejo de su sensibilidad a shocks y a una menor liquidez), mientras que el *SP500* se mantuvo cercano al comportamiento aleatorio ideal.

El contraste entre series originales y barajadas revela que gran parte de la multifractalidad procede de las correlaciones temporales, apuntando a un ajuste gradual hacia dinámicas más estables. Estos resultados sugieren que los índices sostenibles pueden igualar la eficiencia de los tradicionales e incorporar rápidamente la información ESG, demostrando que la sostenibilidad y la eficiencia financiera pueden convivir.

Palabras clave: Alimentación Sostenible, eficiencia, índices sostenibles, *MF-DFA*, *ESG*, Seguridad alimentaria, Sostenibilidad.

Abstract

This study examines how the COVID-19 pandemic disrupted four indexes' efficiency (three sustainable food ESG indices and the *SP500*) by applying monofractal and multifractal analysis (MF-DFA) to original and shuffled series. Pre-crisis, indexes were moderately efficient with temporal-memory multifractality. After COVID-19, sustainable indexes lost some of that efficiency and exhibited greater multifractal complexity (due to their sensitivity to shocks and lower liquidity), while the *SP500* stayed close to ideal random behavior. The comparison between original and shuffled series shows that much of the observed multifractality stems from temporal correlations, pointing to a gradual adjustment toward more stable dynamics. These findings suggest that sustainable indices can match the efficiency of traditional ones and quickly incorporate ESG information, demonstrating that sustainability and financial efficiency can coexist.

Keywords: sustainable food, market efficiency, sustainable indices, *MF-DFA*, *ESG*, food security, sustainability.

1.-Introducción

El hambre y la inseguridad alimentaria siguen siendo retos estructurales que afectan a cientos de millones de personas en el mundo.

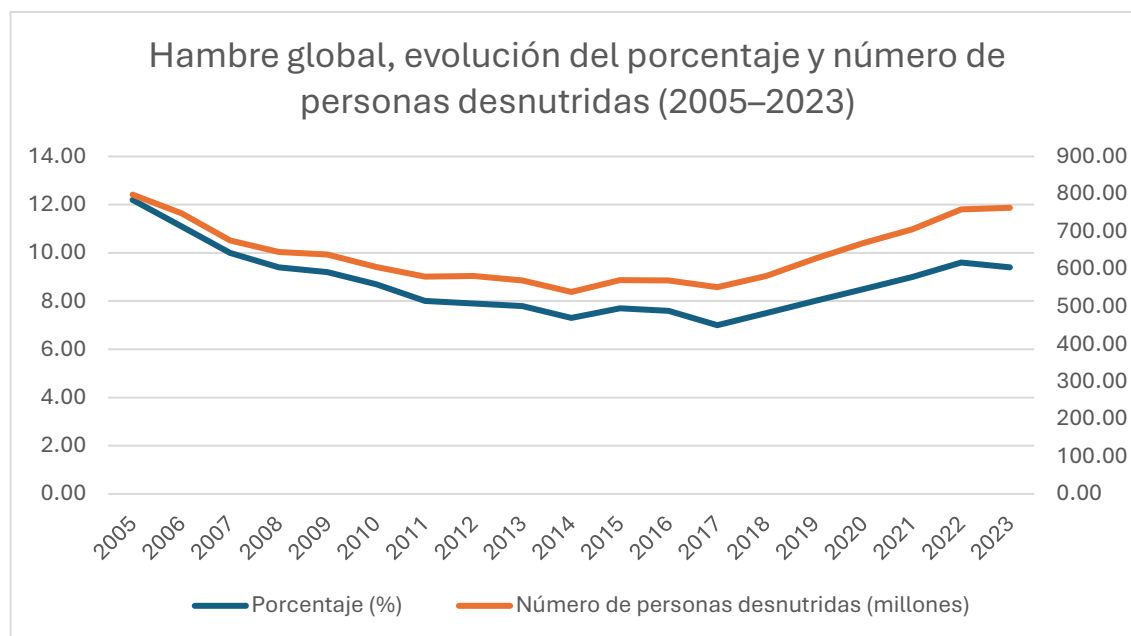


Ilustración 1: Grafico de líneas que representa la evolución desde el 2005 hasta el 2023 del porcentaje y el número de personas que padecen desnutrición. Fuente: elaboración propia a partir de datos del informe “The state of food security and nutrition in the world” [FAO,2024](#)

Según la [FAO \(2024\)](#) y como se puede observar en la Ilustración 1, una de cada diez personas está subalimentada, una cifra que ha empeorado en regiones particularmente vulnerables, debido a conflictos armados, inestabilidad política y los efectos del cambio climático. En respuesta a esta situación, la Agenda 2030 de las Naciones Unidas propone, a través del Objetivo de Desarrollo Sostenible número 2, “poner fin al hambre, lograr la seguridad alimentaria y la mejora de la nutrición, y promover la agricultura sostenible”. Alcanzar este objetivo exige no solo acción pública, sino también un papel activo del sector privado, y en particular, del capital financiero como catalizador de cambio.

Invertir en empresas que producen alimentos accesibles y nutritivos, desarrollan soluciones tecnológicas para el sector agroalimentario o contribuyen a la sostenibilidad del sistema alimentario global, se ha convertido en una vía relevante para conectar el mundo financiero con los retos sociales y ecológicos del siglo XXI. Las finanzas sostenibles, en este contexto, no solo buscan rentabilidad, también se presentan como

instrumentos clave para transformar las capacidades productivas actuales y facilitar el acceso a una alimentación adecuada. La incorporación de criterios ESG (ambientales, sociales y de gobernanza) en las decisiones de inversión permite alinear los flujos de capital con los valores de sostenibilidad, generando impactos positivos medibles en la seguridad alimentaria global ([Pelster et al., 2024](#); [IFC, 2008](#)).

Sin embargo, para que estas inversiones sean realmente efectivas (desde una lógica financiera y desde una perspectiva de impacto social) es necesario analizar si los mercados en los que se realizan presentan características de eficiencia. En términos clásicos, [Fama \(1970\)](#) define la eficiencia de mercado como la capacidad de los precios para reflejar toda la información disponible. En un mercado eficiente, los activos se valoran de forma justa, lo que permite a los inversores tomar decisiones informadas, gestionar el riesgo con mayor precisión y construir carteras más sólidas. Pero cuando hablamos de sectores emergentes, como el de la alimentación sostenible, surgen interrogantes importantes: ¿son estos mercados igualmente eficientes? ¿Cómo responden a eventos críticos, como una pandemia global? ¿Y qué implicaciones tiene esto para los Objetivos de Desarrollo Sostenible?

Este trabajo propone analizar la eficiencia de las inversiones en el sector de la alimentación sostenible a través de herramientas econométricas avanzadas, como técnicas monofractales y multifractales, aplicadas a índices financieros especializados. Se estudiará la evolución de la eficiencia antes y después de la pandemia de COVID19, considerando el impacto que este evento disruptivo tuvo sobre los mercados, y se evaluará hasta qué punto los mercados financieros dedicados a la alimentación sostenible reflejan una estructura madura o, por el contrario, presentan ineficiencias estructurales que puedan ser comprendidas desde la visión de la inversión sostenible. A través de este análisis, se pretende contribuir a una comprensión más profunda de cómo las finanzas pueden ser una herramienta no solo para generar retorno, sino también para impulsar transformaciones alineadas con los grandes desafíos del sistema alimentario global.

2.-Objetivo

El objetivo principal de este trabajo consiste en analizar la eficiencia del mercado. En un contexto global donde los sistemas alimentarios se enfrentan a desafíos como el cambio

climático, la presión demográfica y la inestabilidad geopolítica, se considera imprescindible evaluar si los mercados financieros se adaptan de manera eficiente a esta transformación estructural. Para ello, el estudio se centrará en el análisis de carteras compuestas por compañías que desarrollan tecnologías, productos o servicios clave para el futuro sostenible de la alimentación.

El propósito específico es estudiar el comportamiento de estas carteras utilizando herramientas avanzadas como el análisis multifractal (*MF-DFA*), que permite captar estructuras complejas y dependencias temporales no lineales en las series de precios. A través del cálculo del exponente de Hurst y del espectro de singularidad, se determinará si las carteras analizadas presentan signos de eficiencia o ineficiencia de mercado. En particular, es interesante observar cómo estas propiedades cambian a lo largo del tiempo y frente a eventos disruptivos, como la pandemia de COVID19, que alteró profundamente las cadenas de suministro alimentario y los flujos de inversión sostenible.

Además, este trabajo busca aportar evidencia empírica sobre el potencial y la posibilidad de inversión en sostenibilidad alimentaria como estrategia financiera viable. Más allá de la rentabilidad, se quiere entender si estas carteras son capaces de generar impacto positivo y, al mismo tiempo, funcionar dentro de dinámicas de mercado eficientes.

3.-Metodología

En este trabajo se utilizará tanto el análisis monofractal como el multifractal para evaluar la eficiencia del mercado en el sector de la alimentación sostenible. Ambas técnicas permiten identificar y modelar las características de los datos financieros que no pueden ser capturadas por métodos tradicionales, lo que es crucial para entender mejor las dinámicas de los mercados agrícolas y cómo estas pueden ser aprovechadas para promover inversiones sostenibles.

El punto de partida será la selección de varios índices bursátiles que agrupan empresas del sector alimentario con enfoque innovador y responsable. Se estudiará cómo evolucionan a lo largo del tiempo en distintos contextos económicos. Esta aproximación permitirá observar si dichas inversiones funcionan de manera estable y como responden a los cambios del entorno.

El enfoque adoptado busca entender si los movimientos de precios en este tipo de activos muestran señales de eficiencia o, por el contrario, dejan entrever irregularidades y tendencias. Con esta base, el análisis permitirá extraer conclusiones sobre la solidez, coherencia y grado de madurez de los mercados relacionados con la sostenibilidad alimentaria, contribuyendo así a una visión más completa de su potencial desde el punto de vista financiero y social.

Para realizar el procesamiento, análisis de datos, análisis monofractal y multifractal de las series temporales se va a utilizar como herramienta principal el lenguaje de programación Python. Gracias a las bibliotecas integradas, como pandas, numpy y módulos específicos para MF-DFA, desarrollado por [Gorjão \(2022\)](#) se podrá automatizar la extracción de fluctuaciones de los índices, calcular los exponentes de escala y generar los espectros de singularidad de forma rápida y reproducible para distintos periodos. Python permite integrar visualizaciones dinámicas y asegurar la trazabilidad completa de los pasos del análisis, facilitando tanto la validación de los resultados como la réplica del estudio por parte de terceros. El código se podrá localizar en el Anexo del trabajo.

4.- Alimentación sostenible

Primero de todo, se ha de definir que es la alimentación sostenible para comprender realmente el alcance de este trabajo, y entender el papel que tiene la alimentación sostenible en el futuro. Observando la definición de la alimentación sostenible, propuesta por la Organización de las Naciones Unidas para la Alimentación y la Agricultura (FAO) en un informe hecho por un comité de expertos, es la alimentación que: “brinda seguridad alimentaria y nutrición para todos de tal manera que las bases económicas, sociales y ambientales para generar seguridad alimentaria y nutrición para las generaciones futuras no se vean comprometidas” ([HLPE, 2014, p.31](#)).

Según el [High Level Panel of Experts \(2014\)](#), la sostenibilidad de los sistemas alimentarios está determinada por factores ambientales, económicos y sociales. Muchos de estos factores son endógenos al sistema alimentario, y otros provienen del entorno, como las políticas de protección social. La forma en que se utilizan los recursos dentro del sistema es clave para evaluar su sostenibilidad. Por ejemplo, el uso excesivo de recursos compromete no solo el medio ambiente, sino también las bases económicas y

sociales sobre las que se sustenta la seguridad alimentaria, afectando dimensiones como la disponibilidad, el acceso, la utilización y la estabilidad del suministro de alimentos

Los autores enfatizan la necesidad de considerar las inversiones como una herramienta crucial para lograr sistemas alimentarios sostenibles. Esto incluye tanto inversiones públicas como privadas, especialmente en infraestructuras logísticas, como cadenas de frío y transporte para productos perecederos. La transformación de productos alimentarios, su procesamiento y el desarrollo de capacidades mediante la educación y la formación son también fundamentales para reducir las pérdidas y desperdicios alimentarios y, con ello, mejorar la seguridad alimentaria desde una perspectiva sostenible.

Por otro lado, “los sistemas alimentarios deben ser entendidos como sistemas sociales-ecológicos complejos, caracterizados por múltiples interacciones entre componentes humanos y naturales” ([Allen & Prosperi, 2016, p. 957](#)). Esta concepción sistémica implica que cualquier transformación significativa en el sistema (transición hacia una alimentación más sostenible) requiere comprender también cómo funcionan los mercados que lo sustentan. Es decir, si los sistemas alimentarios dependen de su capacidad para mantenerse funcionales en el tiempo y adaptarse a crisis o perturbaciones, entonces también deben ser analizados los mecanismos de inversión, financiación y asignación de recursos que permiten, o dificultan esa sostenibilidad.

Los mercados juegan un papel central en este proceso, porque canalizan las decisiones de millones de actores económicos y, con ello, influyen directamente qué tipo de agricultura se financia, qué modelos de distribución se priorizan y qué tecnologías alimentarias se desarrollan. Analizar cómo se comportan estos mercados no es solo una cuestión de interés financiero, sino también una herramienta clave para evaluar si los recursos están fluyendo hacia modelos productivos que realmente fortalecen la resiliencia, la equidad y la sostenibilidad del sistema alimentario global.

Tal como señala [Goold et al. \(2020\)](#), los mercados de los sistemas alimentarios están expuestos a una serie de factores endógenos y exógenos. Entre ellos, encontramos la volatilidad de los precios, el crecimiento de la población, y los efectos del cambio climático que condicionan su funcionamiento y sostenibilidad. Su estudio demuestra que modelos sencillos pueden señalar cuándo un sistema alimentario se vuelve insostenible: depende más de importaciones, sus precios suben, la demanda cae y las reservas

disminuyen. En consecuencia, analizar las reglas del mercado y su evolución no es un complemento, sino un componente central para anticipar escenarios de vulnerabilidad o colapso. De poco sirve hablar de sistemas resilientes si no se examinan las lógicas de intercambio, competencia o especulación que pueden amplificar los riesgos y debilitar la seguridad alimentaria a largo plazo.

La alimentación y la agricultura sostenibles están profundamente interrelacionadas, aunque no son términos equivalentes. Se puede decir que la agricultura sostenible es una condición necesaria, pero no suficiente, para alcanzar una alimentación sostenible.

La agricultura sostenible se refiere a un modelo de producción agrícola que busca mantener la productividad a largo plazo respetando los límites ecológicos, garantizando la viabilidad económica de los agricultores y promoviendo condiciones sociales justas. Incluye prácticas como el uso eficiente de recursos, la protección del suelo y el agua, la reducción del uso de agroquímicos y la promoción de la biodiversidad. Este enfoque agrícola asegura el suministro de alimentos y el sustento tanto presente como futuro, al tiempo que protege el entorno natural ([FAO, 2018](#)).

Según el [U.S. Farm Bill de 1990](#), la agricultura sostenible no solo protege el medio ambiente, sino que también garantiza la viabilidad económica de las explotaciones y el uso eficiente de los recursos, todo ello ligado a objetivos sociales y ambientales. Por eso, al evaluar inversiones sostenibles en el ámbito alimentario es fundamental comprobar si los mercados respaldan estos tres pilares estructurales ([Velten et al, 2015](#)).

4.1.- Criterios ESG

Entre 2004 y 2008, una iniciativa impulsada por el Pacto Mundial de la ONU y la IFC sentó las bases conceptuales y estratégicas de los criterios ESG, proponiendo integrar aspectos ambientales, sociales y de gobernanza en las decisiones financieras ([IFC, 2008](#)). El objetivo central es alinear las decisiones de inversión con una visión más sostenible del desarrollo económico, reconociendo que estos factores no solo representan riesgos no financieros, sino también oportunidades significativas de creación de valor ([IFC, 2008](#)).

Los criterios ESG son dimensiones clave para evaluar empresas y proyectos según su impacto ambiental, su compromiso social y la calidad de su gobernanza corporativa, y su análisis mejora la toma de decisiones en los mercados financieros. Desde 2004 se ha

comprobado que los asuntos ESG pueden influir de forma significativa en el desempeño de compañías y sectores, y que su integración en el análisis de inversiones está al alcance de la profesión financiera. No son meros principios éticos, sino factores materiales que condicionan la rentabilidad, la estabilidad y el perfil de riesgo de los activos ([IFC, 2008](#)).

El informe concluye que, si bien ha habido avances, la integración plena de los criterios ESG en los mercados financieros requiere el liderazgo de altos ejecutivos y una implementación sistemática en todos los niveles organizativos. Asimismo, remarca que el uso de estos criterios puede representar “una fuente importante de diferenciación competitiva y creación de valor para las instituciones financieras que los integran en su estrategia” ([IFC, 2008, p.3](#)).

Los criterios ESG se dividen en tres dimensiones clave: el aspecto ambiental se enfoca en cómo las empresas gestionan su impacto ecológico y el cambio climático; el componente social analiza su compromiso con los derechos laborales, la inclusión y la comunidad; y la gobernanza examina la transparencia, la ética y la estructura de toma de decisiones ([ISO, 2018](#)).

De este modo, los criterios ESG no solo permiten evaluar el impacto social y ambiental de las empresas, sino que también funcionan como un indicador de su capacidad para gestionar riesgos y oportunidades en mercados cada vez más exigentes ([Pelster et al., 2024](#)).

El concepto ESG, aunque consolidado en el siglo XXI, tiene sus raíces en prácticas de exclusión ética (hoy en día llamada “*screening*” a la hora de seleccionar que activos incluir en las carteras) del siglo XIX, como las promovidas por los cuáqueros, que evitaban invertir en sectores considerados inmorales ([Eccles et al., 2020](#)). Con el tiempo, esta ética evolucionó hacia la Inversión Socialmente Responsable, y en los años 80 y 90 se profesionalizó con la creación de métricas no financieras gracias a entidades como *KLD* o *Innovest*. El fortalecimiento institucional llegó con los Principios para la Inversión Responsable en 2006 y normas ISO que estandarizaron la medición de impactos ambientales ([ISO, 2018](#)). En la última década, el auge de los bonos verdes y la incorporación de criterios ESG en las carteras han demostrado su utilidad para gestionar riesgos climáticos, aunque persisten críticas sobre su falta de estandarización y el riesgo de *greenwashing* ([Pelster et al., 2024](#); [Pollman, 2022](#)).

El *greenwashing* consiste en el uso estratégico de los ODS y los criterios ESG por parte de las empresas para proyectar una imagen de sostenibilidad sin implementar cambios verdaderos en sus operaciones. Según [Freitas Netto et al. \(2020\)](#), muchas organizaciones aprovechan la ambigüedad de estos marcos para hacer afirmaciones genéricas o promocionales, desviando la atención de prácticas que continúan siendo dañinas y erosionando la confianza de inversores y consumidores.

4.2.-ODS

En 2015, la Organización de las Naciones Unidas (ONU) lanzó los Objetivos de Desarrollo Sostenible (ODS) como parte central de su Agenda 2030, con el propósito de ofrecer un marco integral para enfrentar los principales retos sociales, económicos y medioambientales a nivel global ([United Nations, 2020](#)). Este conjunto de 17 objetivos promueve un desarrollo más justo e inclusivo, abarcando cuestiones fundamentales como la eliminación de la pobreza, la mejora de la seguridad alimentaria, el acceso a una educación de calidad, la protección del clima y el impulso al crecimiento económico. No se tratan solo de marcar objetivos en papel; los ODS impulsan que gobiernos, empresas y ciudadanos unan esfuerzos, compartan ideas y desarrollen proyectos de manera conjunta. De esta forma, se puede proteger el medio ambiente, impulsando el progreso económico y mejorando la calidad de vida, asegurando un futuro sostenible para las próximas generaciones ([FAO, 2019](#)).

La producción y el consumo de alimentos influyen de manera directa en múltiples desafíos globales, por lo que la alimentación sostenible es un pilar indispensable para avanzar en los Objetivos de Desarrollo Sostenible (ODS). Este trabajo se vincula de manera directa con dos de ellos: el ODS 2, que persigue poner fin al hambre y garantizar el acceso a una alimentación segura, nutritiva y suficiente durante todo el año; y el ODS 12, que promueve modalidades de consumo y producción sostenibles. Ambos objetivos reconocen que la transformación de los sistemas alimentarios es clave para avanzar hacia un desarrollo inclusivo, resiliente y ambientalmente responsable ([FAO, 2019](#)).

El ODS 2 no se limita a la producción de alimentos, sino que también pone el foco en la sostenibilidad de los sistemas agrícolas y en la capacidad de resistir crisis como el cambio climático o los conflictos geopolíticos. El ODS 12 subraya la urgencia de reducir el desperdicio alimentario, mejorar la eficiencia en el uso de recursos naturales y fomentar

cadena de valor sostenibles. Estas metas sólo pueden alcanzarse con el impulso conjunto de políticas públicas e inversiones privadas orientadas al largo plazo. Desde esta perspectiva, el análisis de la eficiencia de los mercados financieros vinculados a la alimentación sostenible se convierte en una herramienta valiosa: permite por un lado comprobar si los riesgos de seguridad alimentaria (como el crecimiento poblacional, el cambio climático y la presión sobre los recursos) se han incorporado al precio de los índices temáticos de manera eficiente ([Godfray et al., 2010](#)). Por otro lado, permite comprobar si el mercado reconoce y valora adecuadamente el impacto financiero de los factores ESG en las empresas del sector ([Clark et al., 2015](#)), por tanto, si los flujos de capital están alineados con los principios que sustentan los ODS.

Con todo lo previamente descrito, este trabajo no se limita a una exploración financiera. Más bien, plantea que el comportamiento multifractal de los precios en índices agroalimentarios puede ofrecer pistas sobre la madurez de estos mercados y sobre su capacidad para canalizar inversión hacia soluciones coherentes con los compromisos internacionales en materia de seguridad alimentaria, eficiencia productiva y sostenibilidad. En definitiva, si los mercados muestran señales de eficiencia creciente y estabilidad, entonces pueden considerarse como aliados fundamentales para avanzar hacia los ODS 2 y 12 de forma tangible y medible.

5.-Marco teórico

A continuación, se explicarán las tres bases teóricas en las que se fundamenta este trabajo. La *Efficient Market Hypothesis*, el *DFA* (derivado del *MF-DFA*) y el *MF-DFA*.

5.1.- Efficient Market Hypothesis

La Hipótesis del Mercado Eficiente (*EMH*, por sus siglas en inglés) sostiene que los precios de los activos financieros reflejan en todo momento toda la información disponible en el mercado. [Fama \(1970\)](#) establece que en un mercado eficiente ningún inversor puede obtener rendimientos superiores al promedio de manera sistemática, ya que los precios incorporan instantáneamente cualquier nueva información relevante. La *EMH* se articula en tres formas: débil, semi-fuerte y fuerte, dependiendo del grado de información considerada (histórica, pública y privada respectivamente). La eficiencia

implica que los precios siguen un comportamiento de paseo aleatorio, en el que las variaciones futuras no pueden predecirse a partir de los datos pasados.

[Malkiel, \(2003\)](#) argumentó que, aunque se han documentado algunas anomalías y patrones temporales en los mercados, estos tienden a desaparecer rápidamente cuando se hacen públicos, lo que refuerza la idea de que los mercados son eficientes en promedio. Sin embargo, otros investigadores han señalado que existen episodios de ineficiencia ligados a comportamientos irracionales de los inversores, barreras informativas o shocks externos. Estas críticas han dado lugar al desarrollo de enfoques complementarios, como las finanzas conductuales, que reconocen la influencia de factores psicológicos en la toma de decisiones financieras.

La *EMH* es un punto de referencia fundamental en el análisis de mercados financieros. Su relevancia para este trabajo radica en que sirve como marco de comparación para evaluar la eficiencia de los mercados de alimentación sostenible. Mediante técnicas como el análisis multifractal, es posible detectar desviaciones del comportamiento aleatorio esperado en un mercado eficiente, lo cual puede señalar oportunidades de inversión, pero también advertir sobre vulnerabilidades estructurales en sectores emergentes.

5.2.-DFA

El *Detrended Fluctuation Analysis (DFA)*, un tipo de estudio monofractal, es método muy extendido para analizar la consistencia de los datos y las correlaciones temporales dentro de una serie. ([Berthouze & Farmer, 2012](#)). Un análisis *DFA* permite obtener un único valor del exponente de Hurst, que describe la persistencia o anti-persistencia de la serie temporal ([Hurst, 1951](#)).

[Hurst \(1951\)](#), al definir el exponente que lleva su nombre mientras estudiaba la capacidad de almacenamiento de agua en el río Nilo, descubrió que las fluctuaciones en su caudal mostraban memoria de largo plazo inexplicable mediante procesos estocásticos tradicionales. El exponente de Hurst (H) es una medida que describe la dependencia temporal en una serie de datos y su grado de aleatoriedad ([Hurst, 1951](#)). Se utiliza para analizar la memoria de un sistema, es decir, si los valores futuros dependen de los valores pasados. Si $H = 0.5$, la serie es completamente aleatoria, como en el caso de un paseo aleatorio, donde los cambios no tienen una relación con el pasado. Cuando $H > 0.5$, el

sistema presenta persistencia, lo que significa que una tendencia observada en el pasado tiende a continuar en el futuro. En cambio, si $H < 0.5$, la serie es anti-persistente, lo que indica que los cambios suelen revertirse, es decir, un aumento en el pasado hace más probable una disminución en el futuro y viceversa.

DFA fue desarrollado para abordar el problema de la no estacionariedad, diferenciándose del análisis clásico de rango sobre desviación estándar (*R/S Analysis*) mediante la aplicación de una operación de “*detrend*”, en la cual un polinomio (típicamente de primer orden) se ajusta localmente en cada ventana de la serie, permitiendo eliminar tendencias espurias y preservar la verdadera estructura de correlación de los datos ([Bryce & Sprague, 2012](#); [Granger & Newbold, 1974](#)). Al eliminar los efectos de no estacionariedad, el *DFA* permite ver cómo las fluctuaciones de una serie aumentan de forma proporcional al tamaño de la ventana temporal siguiendo una ley de potencia y, a partir de ahí, calcular con exactitud el exponente de Hurst, que indica si una serie financiera mantiene memoria de su propia trayectoria o, por el contrario, se comporta de manera completamente aleatoria.

Desde un punto de vista empírico, se ha demostrado que *DFA* proporciona estimaciones sólidas en comparación con otros métodos heurísticos, como el *análisis R/S*, y compite con otros enfoques, también teóricamente justificados, como el método de *Local Whittle* ([Whittle, 1953](#)), en la estimación de la dependencia de largo alcance en señales con ruido fraccionario browniano ([Bryce & Sprague, 2012](#)).

Según [Peng et al. \(1994\)](#), *DFA* es una herramienta robusta para identificar estructuras de dependencia en datos ruidosos, lo que lo hace especialmente útil en mercados financieros. El exponente de Hurst, indica el grado de eficiencia del mercado. En mercados eficientes, los precios tienden a mostrar un comportamiento de paseo aleatorio, mientras que en mercados ineficientes pueden existir patrones predecibles ([Mandelbrot, 1997](#)). Esto convierte al *DFA* en una técnica valiosa para evaluar si los precios de los activos reflejan la información disponible o si presentan ineficiencias que pueden ser explotadas por inversores.

En *DFA*, al no existir una distribución asintótica conocida para el exponente de escala α , se recurre al *bootstrap* para simular su distribución en muestras de tamaño finito y extraer valores críticos que sirvan de referencia en el contraste estadístico. Concretamente, se generan numerosas réplicas de la serie original mediante el remuestreo con reemplazo, se

calcula el exponente de Hurst en cada réplica y, a partir de la distribución empírica resultante, se obtienen percentiles (por ejemplo, el 5% y el 95 %) que delimitan la región de aceptación de la hipótesis nula. Estos valores críticos permiten decidir si el α observado en la serie real refleja una dependencia a largo plazo estadísticamente significativa o si, por el contrario, podría atribuirse a la variabilidad inherente a la muestra ([Grau-Carles, 2006](#)).

5.3.-MF-DFA

El análisis multifractal es una extensión del análisis monofractal que permite capturar una gama más amplia de comportamientos en una serie temporal. Esto permite ir modelando no solo la regularidad de las fluctuaciones, sino también la variabilidad en diferentes escalas de tiempo ([Mandelbrot, 1999](#)). A diferencia de los sistemas monofractales, los sistemas multifractales se caracterizan por una distribución continua de exponentes de singularidad que describen el comportamiento en diversas escalas.

El análisis multifractal se ha aplicado con éxito en finanzas para identificar propiedades complejas en los mercados financieros que no pueden ser explicadas por modelos tradicionales de paseo aleatorio o por el exponente de Hurst monofractal ([Calvet & Fisher, 2002](#)). Para este trabajo, se utilizará la técnica *Multi-Fractal Detrended Fluctuation Analysis* (MF-DFA), desarrollada por [Kantelhardt et al. \(2002\)](#), que permite cuantificar la multifractalidad en series temporales financieras. Esta técnica será útil para poder identificar patrones de comportamiento de largo plazo y variaciones a diferentes escalas en los índices que se analizarán.

El uso de MF-DFA es muy interesante para analizar la complejidad inherente a los mercados de alimentos y las inversiones relacionadas con la agricultura sostenible. Las series temporales de precios en este sector pueden estar afectadas por fenómenos exógenos, como el cambio climático o la inestabilidad geopolítica, lo que introduce una variabilidad que es capturada por la multifractalidad ([Barunik et al., 2012](#)).

La explicación del proceso de cálculo del MF-DFA se basa en el trabajo de [Kantelhardt et al. \(2002\)](#).

El primer paso para aplicar el MF-DFA sobre una serie de datos, consiste en definir la serie temporal sobre la cual se realizará el análisis. Dado un conjunto de precios históricos

de un índice financiero, $P(t)$, con $t = 1, 2, \dots, N$, se calcula el retorno logarítmico de la serie para eliminar tendencias exponenciales y garantizar la estacionariedad de los datos. El retorno logarítmico se obtiene a través de la ecuación:

$$R(t) = \ln P(t) - \ln P(t - 1) \quad [1]$$

Donde $R(t)$ representa la variación logarítmica de los precios entre dos instantes de tiempo consecutivos. La transformación logarítmica es una práctica muy común en finanzas para analizar los retornos de los mercados, ya que permite normalizar la serie y reducir efectos de heterocedasticidad en los datos ([Fama, 1970](#)).

Una vez obtenidos los retornos, se construye el perfil acumulado de la serie, que es un paso clave para identificar tendencias ocultas y facilitar el análisis de fluctuaciones. Este perfil acumulado se define como:

$$Y_t = \sum_{k=1}^t (r_k - \mu_X), \quad \text{siendo } t = 1, 2, \dots, N \quad [2]$$

donde μ_X es la media de los retornos logarítmicos. La sustracción de la media garantiza que el perfil acumulado tenga un comportamiento centrado y evita la influencia de tendencias globales en la serie.

El siguiente paso es dividir la serie en segmentos de tamaño s y eliminar la tendencia local dentro de cada segmento. La serie se divide en ventanas de longitud s , tanto desde el inicio como desde el final, de modo que se obtienen $2N/s$ segmentos en total. Dentro de cada segmento, se ajusta un polinomio de orden m , típicamente un polinomio cuadrático ($m = 2$), para capturar tendencias locales de la serie. Posteriormente, se calcula la función de fluctuación sin tendencia dentro de cada segmento como:

$$F^2(v, s) = \frac{1}{s} \sum_{j=1}^s [Y_{v,s}(j) - y_{v,s}(j)]^2, \quad v = 1, \dots, 2N/s \quad [4]$$

Para cada segmento v . Se obtiene así la fluctuación promedio sobre todas las ventanas:

$$F_q(s) = \left\{ \frac{1}{2N_s} \sum_{v=1}^{2N_s} [F^2(v, s)]^{q/2} \right\}^{1/q} \quad [5]$$

para diferentes valores de q . Esta ecuación es clave en el análisis multifractal, ya que permite evaluar cómo varía la dependencia en la serie en función de distintas escalas de tiempo y momentos estadísticos ([Mandelbrot, 1999](#)).

El siguiente paso es analizar cómo escala la función de fluctuación $F_q(s)$ con el tamaño de la ventana s . Si la serie es verdaderamente multifractal, la relación entre estas variables debe seguir una ley de potencias de la forma:

$$F_q(s) \sim s^{H(q)}, \quad H(q) \sim \frac{\log F_q(s)}{\log s}, \quad [6]$$

Donde $H(q)$ es el exponente de Hurst generalizado, que varía en función del parámetro q . Este parámetro distingue entre pequeñas fluctuaciones ($q < 0$) y grandes fluctuaciones ($q > 0$), lo que permite evaluar si la dinámica de los precios es homogénea o si presenta heterogeneidades en distintas escalas. En un mercado eficiente, se espera que $H(q)$ sea aproximadamente constante y cercano a 0.5 para todos los valores de q , lo que indicaría que los precios siguen un paseo aleatorio puro ([Fama, 1970](#)). En contraste, una dependencia significativa de $H(q)$ con q sugiere multifractalidad fuerte, lo que implica que el mercado tiene memoria y posibles patrones predecibles.

A partir de los valores de $H(q)$, se calcula el exponente de masa multifractal $\tau(q)$, definido como:

$$\tau(q) = qH(q) - 1 \quad [7]$$

Este exponente indica la dependencia de la serie con respecto a diferentes momentos estadísticos. En un sistema monofractal, $\tau(q)$ es una función lineal de τ , mientras que en un sistema multifractal presenta una curvatura significativa ([Calvet & Fisher, 2002](#)).

Finalmente, se construye el espectro de singularidad $D(\alpha)$, herramienta fundamental para visualizar la distribución de irregularidades en la serie de precios. Se define mediante:

$$\alpha(q) = h(q) + q \frac{dh(q)}{dq} \quad [8]$$

$$D(\alpha) = q(\alpha(q) - H(q)) + 1 \quad [9]$$

El ancho del espectro $\alpha_{max} - \alpha_{min}$ es un indicador clave de la complejidad del mercado. Un espectro estrecho sugiere que la serie es cercana a un paseo aleatorio y, por ende, refleja un comportamiento eficiente. En cambio, un espectro ancho indica ineficiencia y heterogeneidad en la estructura del mercado ([Kantelhardt et al., 2002](#)).

Aplicación del MF-DFA en finanzas y agricultura

Aunque el uso del análisis monofractal y multifractal ha sido previamente aplicado en el estudio de mercados financieros tradicionales, su implementación específica en el análisis de inversiones ligadas a la agricultura sostenible representa una aproximación metodológica novedosa. En sectores tan sensibles como el agroalimentario, donde los precios están altamente expuestos a factores externos como la variabilidad climática, los cambios abruptos en la demanda global y las alteraciones regulatorias, las herramientas fractales permiten capturar comportamientos no lineales que quedarían invisibles bajo métodos estadísticos convencionales. Esta complejidad estructural requiere instrumentos capaces de identificar dependencias a largo plazo y patrones de comportamiento persistente o anti-persistente, lo que convierte al análisis de fluctuación sin tendencia (*DFA*) y su extensión multifractal (*MF-DFA*) en metodologías idóneas para el caso.

Un estudio previo realizado por [Wang et al. \(2023\)](#) trató de comprender la complejidad de los mercados agrícolas globales, aplicando el análisis de *MF-DFA* a los índices de precios de los mercados de granos, incluyendo maíz, trigo, arroz y cebada. Su investigación se centró en evaluar la existencia de multifractalidad intrínseca, es decir, estructuras complejas en la dinámica de precios que no pueden ser explicadas únicamente por distribuciones no normales o por efectos de largo plazo.

El estudio reveló que los precios de los mercados de maíz y cebada presentan un comportamiento multifractal notable, incluso después de aplicar técnicas de aleatorización como el *shuffling*, que eliminan posibles efectos estadísticos artificiales.

Esto significa que la complejidad observada en estos mercados es real y está asociada a dinámicas internas propias, no a distorsiones del análisis. En otras palabras, los precios en estos mercados siguen patrones no lineales y con memoria de largo plazo, que no pueden ser captados por métodos estadísticos tradicionales. Por el contrario, en los mercados de trigo y arroz, esta complejidad fue menos evidente o estuvo más influida por factores externos, lo que sugiere que su estructura es menos compleja o más cercana a un comportamiento aleatorio.

6.-Base de datos

Para poder analizar correctamente la eficiencia del mercado de la alimentación sostenible se han seleccionado, en base a documentación e histórico de precios, 3 índices que representan el mercado de la alimentación sostenible. Los índices *MSCI ACWI IMI Food Revolution*, en adelante *MSCI*, el *MVIS Global Future of Food ESG*, en adelante *MVIS*, y el *Foxberry Tematica Research Sustainable Future of Food*, en adelante *Foxberry*, ofrecen perspectivas complementarias sobre la innovación, sostenibilidad y eficiencia en la producción de alimentos, ofreciendo cada uno de ellos un enfoque diferenciador.

La siguiente información ha sido obtenida de las “Index Factsheets” de los 3 índices ([MSCI Inc., 2025](#); [MarketVector Indexes GmbH, 2025](#); [Foxberry Ltd., 2022](#)).

Mientras que el índice *MSCI* abarca de manera global la disrupción en la industria alimentaria, el de *MVIS* se centra en empresas con criterios ESG estrictos y el de *Foxberry* prioriza tendencias emergentes y tecnologías disruptivas en la sostenibilidad alimentaria. El estudio de eficiencia de estos índices va a ser fundamental para evaluar el impacto de la inversión sostenible en la seguridad alimentaria y la transición hacia sistemas de producción más eficientes y responsables.

Los índices *MSCI*, *MVIS* y *Foxberry* son carteras financieras diseñadas para reflejar el desempeño de empresas involucradas en la innovación y sostenibilidad del sector alimentario.

El índice *MSCI* se centra en compañías que impulsan la transformación de la industria alimentaria a nivel global, abarcando desde la producción sostenible hasta la tecnología alimentaria avanzada. Este índice forma parte de la serie de índices de MSCI que buscan

capturar oportunidades en tendencias de largo plazo. (MSCI, 2013).

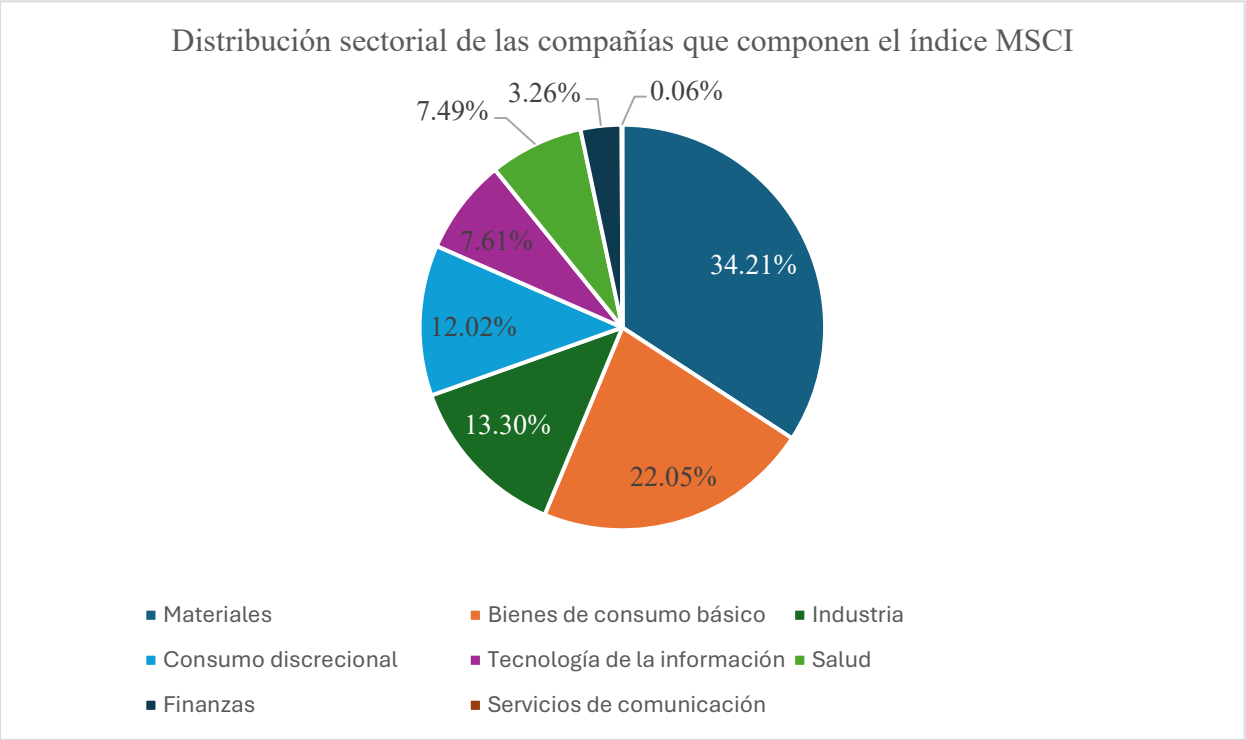


Ilustración 2: Distribución sectorial de las compañías que componen el índice MSCI. Fuente: elaboración propia a partir de información obtenida de la "Index factheet", (MSCI Inc., 2025)

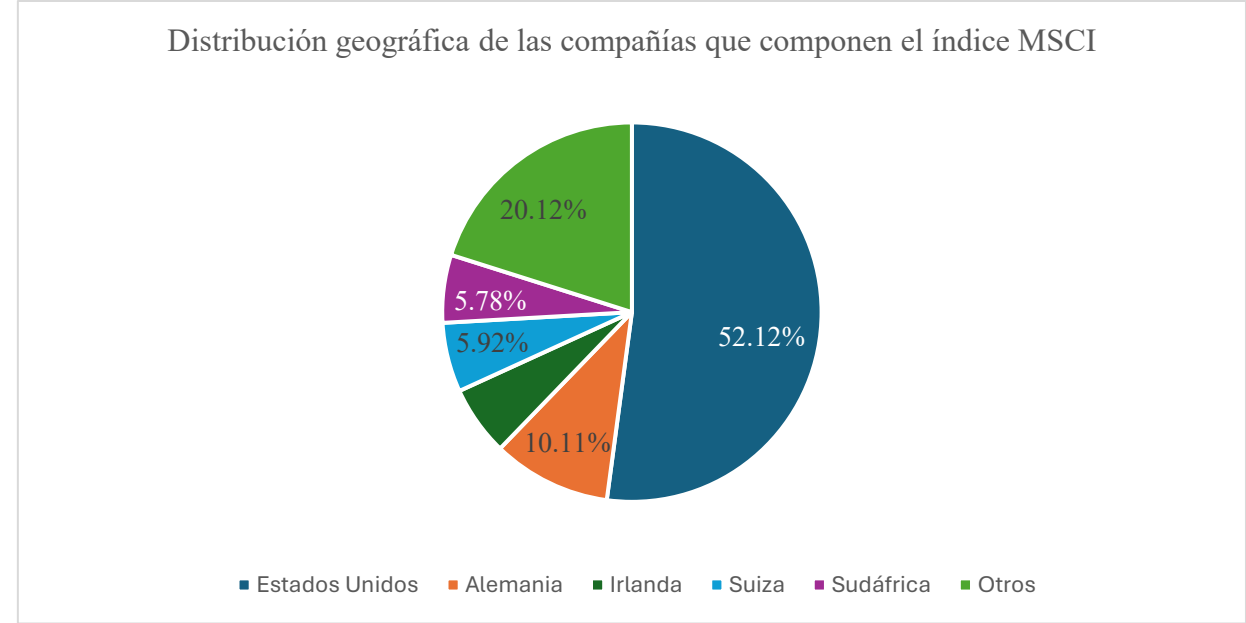


Ilustración 3: Distribución geográfica de las compañías que componen el índice MSCI. Fuente: elaboración propia a partir de información obtenida de la "Index factheet", (MSCI Inc., 2025)

En las Ilustraciones 2 y 3 se aprecia que el índice MSCI está fuertemente sesgado hacia Estados Unidos, que por sí solo concentra el 52,12 % del peso total, mientras que Alemania (10,11 %), Irlanda (5,95 %) y Suiza (5,92 %) se sitúan como los siguientes mercados más relevantes, quedando un 20,12 % agrupado en “Otros”. A nivel sectorial, Materiales acapara la mayor parte con un 34,21 %, seguido de Bienes de consumo básico (22,05 %) e Industriales (13,30 %), mientras que Consumo discrecional, Tecnología de la información y Salud oscilan entre el 7 % y el 12 %, y Finanzas y Servicios de comunicación tienen una presencia marginal (3,26 % y 0,06 %, respectivamente). Estas gráficas reflejan la doble concentración geográfica y sectorial del índice, tanto en el mercado estadounidense como en las industrias vinculadas a materias primas y productos esenciales.

Por su parte, el índice *MVIS*, fue lanzado el 9 de septiembre de 2021 y está compuesto por 35 empresas que ofrecen productos y servicios relacionados con alternativas a la carne y los lácteos, alimentos orgánicos, sabores alimentarios o tecnologías agrícolas innovadoras. Para ser incluidas, las empresas deben generar al menos el 50% de sus ingresos de estas áreas y cumplir con criterios ambientales, sociales y de gobernanza (*ESG*). Además, deben tener una capitalización de mercado mínima de 150 millones de dólares y cumplir con requisitos de liquidez específicos. El índice se revisa trimestralmente y las ponderaciones de las empresas están limitadas al 8% para garantizar la diversificación ([MarketVector, 2020](#)). En la Ilustración 4 se observa la marcada concentración geográfica, pues 17 empresas de Estados Unidos acaparan el 52,85 % del peso total de la cartera, lo que implica un riesgo país elevado y una fuerte dependencia del mercado estadounidense. Asimismo, existen notables desajustes entre el número de componentes y su aportación al índice: por ejemplo, aunque Japón aporta cuatro compañías, su peso combinado apenas alcanza el 4,28 %, mientras que las dos entidades de las Islas Caimán suman un 4,88 %, evidenciando que el impacto individual de cada acción varía sustancialmente según su capitalización y rol estratégico en la temática

sostenible.

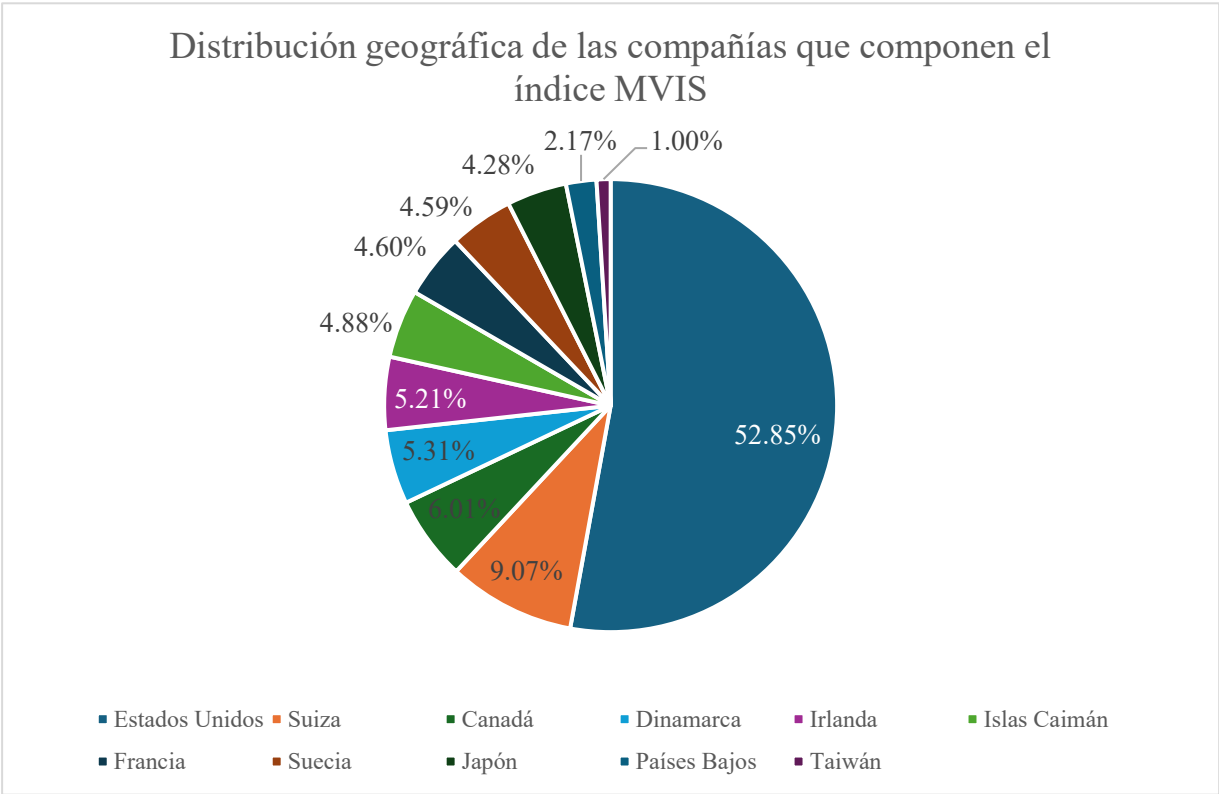


Ilustración 4: Distribución geográfica de las compañías que componen el índice MVIS. Fuente: elaboración propia a partir de información obtenida de la "Index factheet", ([MarketVector Indexes GmnH, 2025](#))

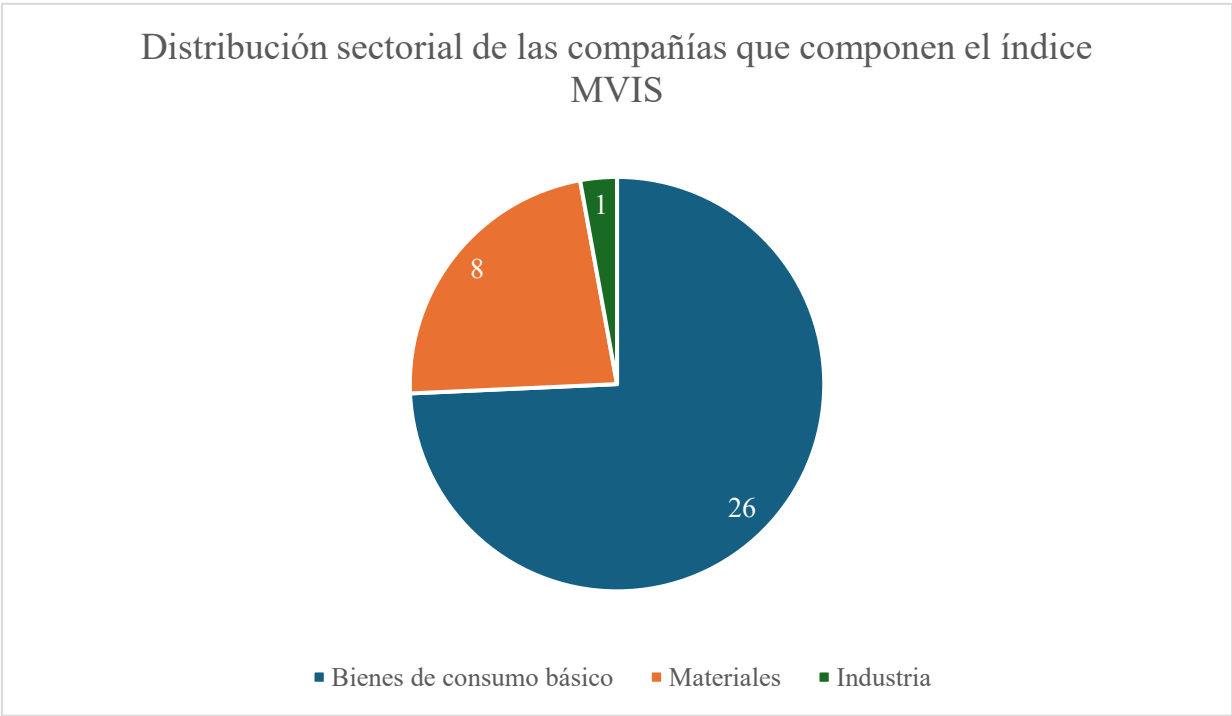


Ilustración 5: Distribución sectorial de las compañías que componen el índice MVIS. Fuente: elaboración propia a partir de información obtenida de la "Index factheet", ([MarketVector Indexes GmnH, 2025](#))

Como muestra la Ilustración 5, el índice MVIS se encuentra muy concentrado en el sector de bienes de consumo básico, con 26 empresas que suman el 70,11 % del peso total, lo que subraya el protagonismo de compañías centradas en productos esenciales dentro de la alimentación sostenible. El rubro de materiales ocupa un 27,73 % (8 firmas), reflejando la relevancia de proveedores de insumos y tecnologías para el sector, mientras que el sector industrial apenas contribuye con un 2,17 % (1 empresa).

Finalmente, el índice *Foxberry* se centra en 158 empresas que contribuyen a un futuro alimentario sostenible, incluyendo aquellas que desarrollan proteínas alternativas, agricultura vertical y otras innovaciones sostenibles en la cadena de suministro alimentaria. Este índice es el resultado de la colaboración entre *Foxberry* y *Tematica Research*, combinando la experiencia en índices financieros con el conocimiento temático en sostenibilidad alimentaria ([Foxberry, 2015](#)).

Se ha realizado la elección de estos tres índices debido a la composición de los portafolios, debido a su enfoque en la sostenibilidad y la innovación dentro de la industria alimentaria. Cada uno aborda diferentes aspectos de la transformación del sector: el índice *MSCI* captura una visión global de la revolución alimentaria; el *MVIS* incorpora criterios *ESG* y se centra en tecnologías y productos específicos que representan el futuro de la alimentación; y el *FXBYFOOD*, destaca empresas que están a la vanguardia de prácticas sostenibles y soluciones innovadoras en la cadena de suministro alimentaria. Esta combinación proporciona una perspectiva integral y diversificada para analizar las tendencias actuales y futuras en la industria alimentaria sostenible.

Además de los 3 índices relacionados con la alimentación sostenible, se ha tenido en cuenta para hacer una comparativa el índice de referencia de los Estados Unidos, el *SP500*. El *SP500* se ha convertido en el “termómetro” del mercado porque agrupa las 500 empresas más grandes de EE. UU. por capitalización, cubriendo cerca del 85 % del valor total de la bolsa y ofreciendo una buena mezcla sectorial. Se utiliza como punto de comparación sólido y reconocible en análisis financieros ([De, 2013](#)). El *SP500* servirá como referencia para comparar la eficiencia de mercado de los índices de alimentación sostenible.

La muestra consiste en datos desde el 03 de marzo de 2015 hasta el 13 de febrero de 2025 para los índices *MSCI*, *FXBYFOOD* y *SP500* y desde el 01 de enero de 2020 hasta el 13 de febrero de 2025 para el índice *MVIS*.

7.-Análisis

Antes de comenzar el análisis es importante entender como han funcionado los índices en el periodo analizado. Como se puede observar en la Tabla 1, el *SP500* muestra un comportamiento, a nivel financiero, muy superior: con una rentabilidad anualizada del 12.6 %, prácticamente duplica al *MSCI* y deja muy atrás a los índices de alimentación sostenible (2.4 % para *FXBYFOOD* y -2 % para *MVIS*).

Aun cuando todos presentan volatilidades parecidas (16–19 %), solo el *SP500* (con un sharpe de 0.50) y el *MSCI* (0.19) ofrecen retornos ajustados al riesgo positivos. Los índices sostenibles arrojan Sharpe negativos, lo que significa que su “montaña rusa” de precios no compensa el viaje: con la misma turbulencia, se obtienen peores resultados.

Tabla 1: Análisis financiero de rentabilidad anualizada, volatilidad anualizada y Sharpe Ratio por índice, MSCI, FXBYFOOD, MVIS, SP500. Fuente: elaboración propia a partir de los retornos de los índices

Índice	Rentabilidad Anualizada (%)	Volatilidad Anualizada (%)	Sharpe Ratio
<i>MSCI</i>	5.71%	16.35%	0.186
<i>FXBYFOOD</i>	2.41%	16.03%	-0.007
<i>MVIS</i>	-1.97%	16.98%	-0.264
<i>SP500</i>	12.632	18.72%	0.502

Además, se presenta, en la Ilustración 6, la evolución de los retornos acumulados de los 3 índices analizados, así como la evolución del índice de referencia, el *SP500*.

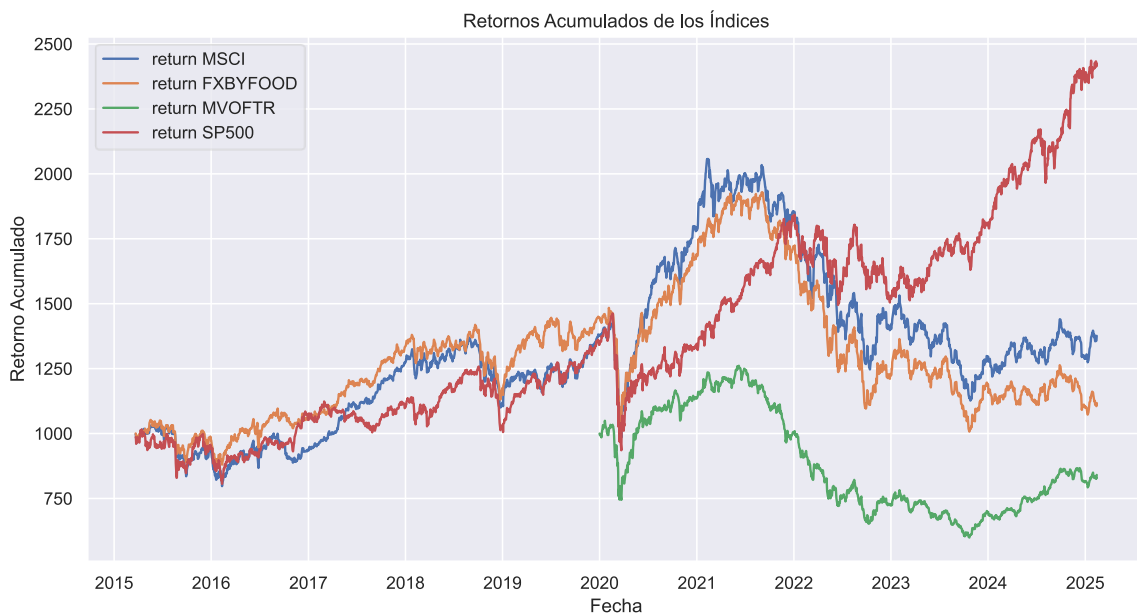


Ilustración 6: Retornos acumulados de los índices MSCI, FXBYFOOD, MVOFTR y el SP500. Fuente: elaboración propia.

Como primer impacto visual es importante resaltar la caída en los 4 índices a raíz de la pandemia del COVID19. Según el estudio de [Baker et al. \(2020\)](#), la caída de los mercados financieros durante el inicio de la pandemia de COVID19 se debió principalmente al aumento súbito y generalizado de la incertidumbre económica. A través del análisis del índice de volatilidad *VIX*, los autores muestran que el nivel de incertidumbre generado por la pandemia superó al de crisis anteriores, incluida la crisis financiera de 2008. Esta incertidumbre afectó simultáneamente tanto a la oferta como a la demanda: por un lado, las cadenas globales de suministro se vieron interrumpidas por el cierre de fábricas y restricciones logísticas, y por otro, el consumo se desplomó ante el confinamiento de millones de personas. Como resultado, los inversores respondieron con una rápida retirada de capitales y ventas masivas de activos, provocando caídas abruptas en los principales índices bursátiles a nivel global ([Baker et al., 2020](#)).

Otro impacto, observable en la Ilustración 5, es la caída de precios de los índices de alimentación sostenible a partir de 2021, que se examinará más adelante.

7.1.- Impacto del COVID19 en los sectores de la alimentación

La pandemia de COVID19 marcó un punto de inflexión en el sistema alimentario mundial al exponer con crudeza las vulnerabilidades estructurales de las cadenas de suministro agrícolas. Las restricciones sanitarias y los confinamientos interrumpieron el transporte de mercancías, generaron escasez de mano de obra y provocaron un aumento de los costes logísticos, afectando desde la producción hasta la distribución de alimentos ([Hobbs, 2020](#)). Estas disrupciones alteraron los flujos comerciales, y además generaron una alta volatilidad en los mercados agrícolas, lo cual dificultó la valoración precisa de los activos y redujo la eficiencia informativa de los precios. Como explican [Baldwin y di Mauro \(2020\)](#), esta situación incrementó la percepción de riesgo entre los inversores, que optaron por reajustar sus carteras hacia sectores considerados más estables, reflejando una clara pérdida de confianza en las inversiones vinculadas a la agricultura convencional durante los momentos más críticos de la pandemia.

Sin embargo, este escenario también impulsó un cambio significativo en la percepción del riesgo financiero, favoreciendo a las inversiones sostenibles. La pandemia actuó como catalizador para que los inversores reconsideraran la solidez a largo plazo de sus carteras, priorizando activos que demostraran resiliencia ante crisis globales. En este sentido, los estudios de [Ding et al. \(2021\)](#) y [Garel & Petit-Romec \(2021\)](#) muestran que las empresas con prácticas sostenibles no solo fueron menos castigadas por el mercado durante los momentos de incertidumbre, además ganaron atractivo por su capacidad de adaptación. Esta tendencia fortaleció la demanda por activos vinculados a la sostenibilidad, incluida la agricultura sostenible, posicionándolos como una alternativa sólida frente a la volatilidad estructural. La pandemia no solo generó un entorno de ineficiencia temporal, sino que también reconfiguró las estrategias de inversión, consolidando el papel de la sostenibilidad como criterio clave en la gestión de riesgos financieros.

Dos meses después del desplome inicial de los precios, los índices comenzaron a recuperarse progresivamente hasta alcanzar su punto máximo en febrero de 2021. A partir de ese momento, los índices de alimentación sostenible (*MVIS*, *FXBYFOOD* y *MSCI*) iniciaron una tendencia bajista, mientras que, en contraste, el *SP500* mantuvo su trayectoria ascendente.

7.2.-Análisis de la caída de los índices de alimentación sostenible a partir de 2021

Este trabajo no tiene como objetivo un análisis financiero de retornos de los índices de alimentación sostenible, sino de la eficiencia de este mercado. No obstante, resulta pertinente examinar la caída de los precios desde 2021, debido a que, en comparación con el índice de referencia, el *SP500*, tienen una actuación muy distinta, recogiendo unos retornos acumulados negativos muy distintos a la tendencia alcista del índice de referencia americano.

La caída en los retornos acumulados de los índices de alimentación sostenible a partir de 2021 puede explicarse por una combinación de factores estructurales, macroeconómicos y financieros. que afectaron específicamente a este sector tras el auge de recuperación inicial experimentado después la pandemia.

Durante el año 2020, en pleno auge de la pandemia de COVID19, los fondos temáticos orientados a la sostenibilidad (alimentación, la agricultura tecnológica y la distribución digital) atrajeron un volumen significativo de capital por parte de inversores que buscaban sectores resilientes y alineados con los Objetivos de Desarrollo Sostenible. Este fenómeno fue impulsado por la percepción de que las estrategias ESG ofrecían mayor capacidad de respuesta frente a crisis globales, tanto en términos financieros como sociales ([Katugampola, 2020](#); [UNCTAD, 2021](#)). Sin embargo, una vez mitigada la emergencia sanitaria global, los mercados experimentaron una fase de reequilibrio. Durante 2021 se registraron salidas netas de capital de los fondos sostenibles (sobre todo de los menos valorados en *ESG*) y esto, junto con la presión de venta, contribuyó a una caída en los precios de los índices sostenibles. Como bien muestran [Pástor y Vorsatz \(2020\)](#) y [Fang y Parida \(2022\)](#) tras la recuperación inicial de la pandemia, los inversores retiraron dinero de los fondos con peor desempeño ESG, mientras que los más sólidos siguieron recibiendo entradas, agravando la caída de precios en el conjunto del sector. Esto refleja cómo la presión vendedora se prolongó y afectó al valor de los índices sostenibles a lo largo de 2021.

A partir de 2022, el entorno macroeconómico internacional se volvió especialmente adverso para las inversiones de largo plazo asociadas a sostenibilidad. El endurecimiento de la política monetaria, acompañado de altos niveles de inflación y de incertidumbre derivada del conflicto en Ucrania, deterioró el apetito de los inversores por sectores

intensivos en innovación o aún en etapas de consolidación. Como muestran [Iannone et al. \(2024\)](#), las inversiones *ESG* se vieron especialmente expuestas a la volatilidad en países como Alemania e Italia, lo que refleja una sensibilidad superior frente a disrupciones exógenas. Esto penalizó a muchas de las empresas presentes en índices de alimentación sostenible, que suelen operar con márgenes reducidos y modelos de crecimiento a largo plazo, viéndose afectadas por el aumento del coste del capital y la mayor exigencia de retornos inmediatos por parte del mercado.

Además, la diferencia en el desempeño frente a índices como el *SP500* puede explicarse por una cuestión de composición sectorial. Mientras que el *SP500* se ha visto impulsado por el extraordinario crecimiento de empresas tecnológicas (particularmente en inteligencia artificial, software y comercio electrónico), los índices centrados en sostenibilidad alimentaria no gozan de esa misma exposición. En cambio, dependen en mayor medida de sectores vinculados a la agricultura, la cadena logística o la manufactura alimentaria, que son más vulnerables a la volatilidad de precios de materias primas, a costes regulatorios y a tensiones geopolíticas ([UNCTAD, 2021](#); [Iannone et al., 2024](#)).

7.3.- DFA

Tanto para el análisis *DFA*, *MF-DFA*, y la generación de gráficos se ha utilizado como herramienta Python. El código se encuentra en el Anexo del trabajo.

A continuación, se realizará un análisis monofractal de eficiencia de los índices (como referencia del mercado de la alimentación sostenible) utilizando el *DFA*. Como se ha definido en el marco teórico, el *Detrended Fluctuation Analysis (DFA)* es una técnica ampliamente utilizada para evaluar la regularidad y la dependencia temporal en series temporales. Esta técnica permite obtener un único valor del exponente de Hurst, que describe la persistencia o anti-persistencia de la serie temporal ([Hurst, 1951](#)).

De acuerdo con el marco teórico, en el análisis *DFA* se establecen como límites críticos los valores $\alpha = 0.4$ y $\alpha = 0.6$ para testar la hipótesis nula de eficiencia de mercado al 95 % de confianza. Así, siempre que el exponente α se mantenga dentro del rango $0.4 < \alpha < 0.6$ no se rechaza la hipótesis nula, indicando ausencia de evidencia de ineficiencia informacional. Por el contrario, si $\alpha > 0.6$ el sistema exhibe comportamiento persistente, mientras que valores $\alpha < 0.4$ señalan dinámica anti-persistente y, por tanto, rechazo de la

hipótesis de eficiencia.

A continuación, se presentan los resultados de *DFA* sobre los 4 índices:

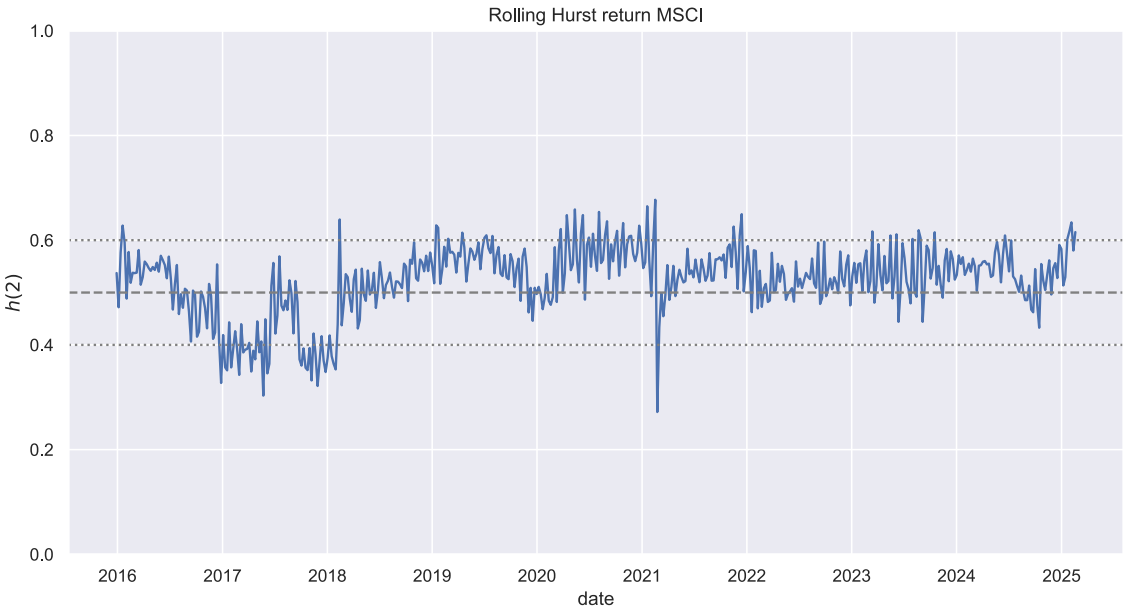


Ilustración 8: Evolución del coeficiente de Hurst, obtenido mediante DFA, del índice MSCI. Fuente: elaboración propia

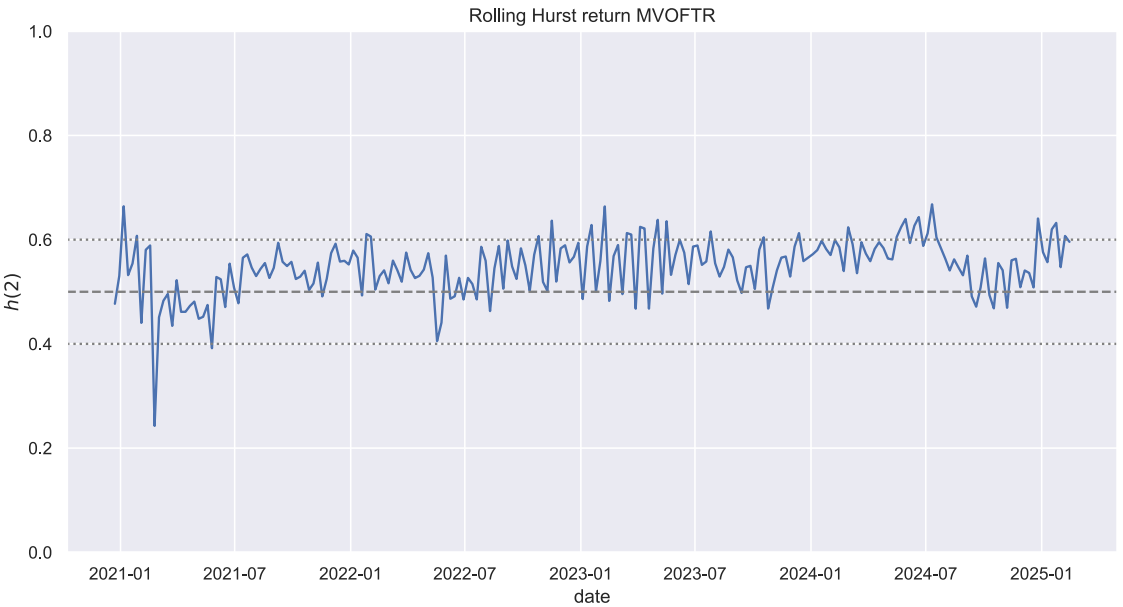


Ilustración 7: Evolución del coeficiente de Hurst, obtenido mediante DFA, del índice MVIS. Fuente: elaboración propia

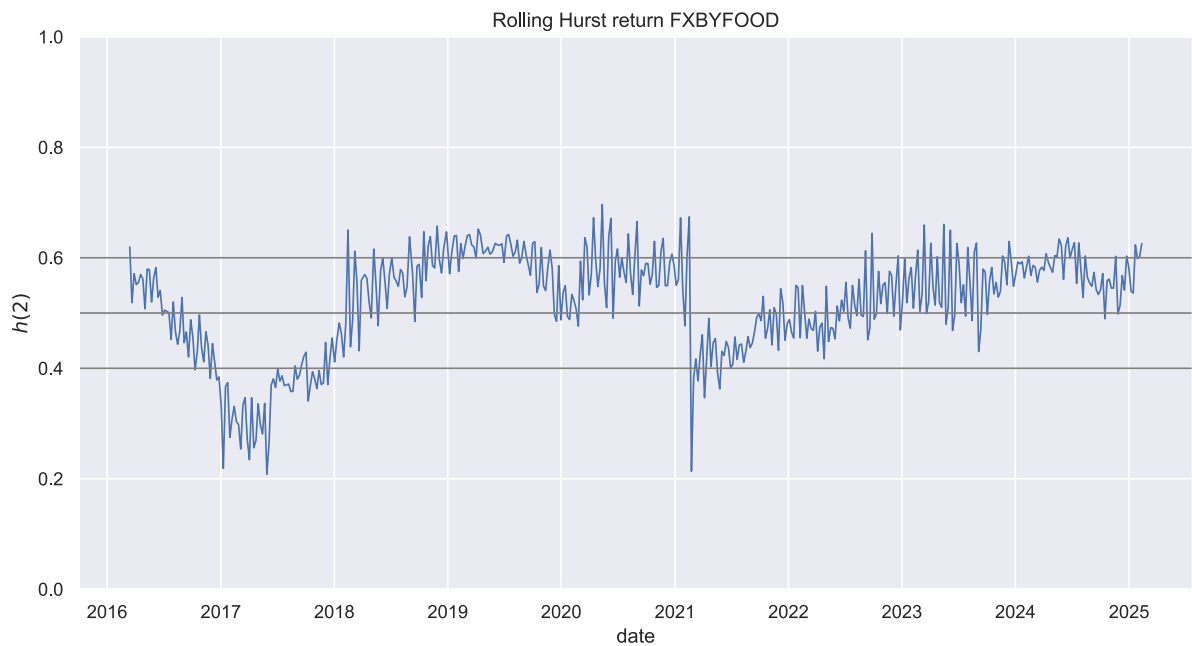


Ilustración 9: Evolución del coeficiente de Hurst, obtenido mediante DFA, del índice FXBYFOOD. Fuente: elaboración propia.

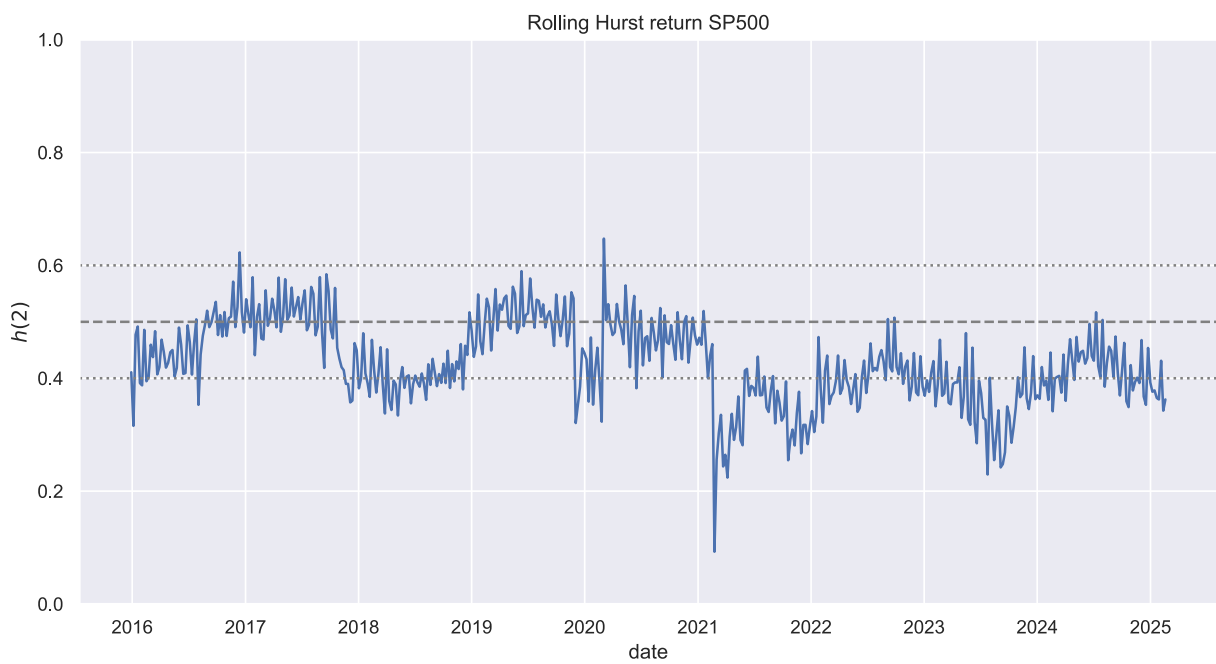


Ilustración 10: Evolución del coeficiente de Hurst, obtenido mediante DFA, del índice SP500. Fuente: elaboración propia

La evolución del coeficiente de Hurst en los cuatro índices analizados, el *SP500* como referencia de mercado consolidado, y los tres índices especializados en sostenibilidad alimentaria (*MSCI*, *MVIS* y *Foxberry*), permite identificar patrones comunes y divergencias significativas, útiles para evaluar su grado de eficiencia relativa. Todos los índices presentan una estructura de comportamiento monofractal que varía a lo largo del tiempo, con valores de Hurst que oscilan entre la persistencia ($H > 0.6$) y la anti-persistencia ($H < 0.4$), lo cual es esperable en mercados que enfrentan fases alternas de estabilidad e incertidumbre. Lo importante es que, en términos generales, los cuatro tienden a estabilizarse en torno a valores cercanos a 0.5 durante distintos tramos del periodo observado. Esta convergencia sugiere que, aunque no todos alcanzan niveles óptimos de eficiencia, los mercados relacionados con la alimentación sostenible no están estructuralmente lejos de un comportamiento eficiente.

El *SP500*, como índice global y diversificado, muestra la mayor estabilidad en su coeficiente de Hurst. Incluso durante la pandemia, su comportamiento apenas se aleja de la zona en torno al 0.5, lo que indica que sus precios se aproximan al ideal de un paseo aleatorio, una característica típica de los mercados eficientes según [Fama \(1970\)](#). Esta estabilidad se asocia a factores como la alta liquidez, la profundidad del mercado y la cobertura sectorial diversificada. Aunque es cierto que en el periodo 2021, 2022 y 2023 muestra un comportamiento cíclico anti-persistente. En contraste, los tres índices de alimentación sostenible presentan una mayor variabilidad en sus valores de Hurst, reflejando una sensibilidad superior a perturbaciones externas, pero sin llegar a exhibir desviaciones estructuralmente alarmantes.

Por ejemplo, el *MVIS* experimenta ciclos marcados con picos de persistencia ($H > 0.6$) seguidos de caídas, lo que podría interpretarse como una reacción a dinámicas propias del sector, como cambios en regulaciones ESG, alteraciones en los precios de insumos agrícolas o variaciones en la demanda sostenible, que tienden a amplificarse en mercados más concentrados. Por su parte, el *MSCI* muestra una recuperación más lenta tras la pandemia, con valores de Hurst que tardan en normalizarse, aunque con una tendencia clara a estabilizarse conforme avanza el periodo. Finalmente, el índice de *FXBYFOOD* se comporta de forma más volátil, con transiciones más abruptas entre fases de persistencia y anti-persistencia, lo que podría estar vinculado a una mayor exposición a empresas jóvenes o emergentes, cuya cotización es más sensible a la percepción de riesgo y a los ciclos de capitalización.

A pesar de estas diferencias, todos los índices muestran una estructura común: periodos de relativa eficiencia interrumpidos por episodios de inestabilidad asociados a eventos exógenos, siendo el más relevante la pandemia de COVID19. Entre 2020 y 2021, se observan oscilaciones bruscas del coeficiente de Hurst en todos los gráficos, lo que indica una ruptura en la estructura habitual de los precios. Este comportamiento responde al impacto generalizado de la pandemia sobre las cadenas de suministro, los flujos logísticos y la demanda global. En los índices de alimentación sostenible, el efecto fue más duradero y profundo, en parte debido a su mayor dependencia de la logística agroalimentaria, a la concentración temática de sus carteras y a la presencia de empresas de menor capitalización y madurez. Sin embargo, lo relevante es que, una vez superado el shock más severo, todos los índices muestran señales de recuperación y estabilización en sus valores de Hurst, acercándose nuevamente a niveles que reflejan un comportamiento compatible con cierta eficiencia de mercado.

Este proceso de normalización refuerza la idea de que, aunque los mercados especializados, como los de alimentación sostenible, aún están en fase de consolidación, su distancia con respecto a los mercados más desarrollados no es estructuralmente amplia. De hecho, la evolución observada apunta a una creciente madurez, en la que los mecanismos de ajuste y la integración de información van ganando eficacia, especialmente en contextos menos turbulentos.

En definitiva, los resultados sugieren que los índices de alimentación sostenible están avanzando hacia comportamientos cada vez más cercanos a la eficiencia, aunque aún persistan diferencias en estabilidad y resiliencia frente a los *benchmarks* tradicionales. Este diagnóstico no es negativo, sino reflejo de un mercado joven que responde a una transformación estructural en los patrones de consumo y producción. Seguir midiendo la evolución de su eficiencia resulta clave para comprender mejor su rol en una transición económica orientada a la sostenibilidad.

7.4.-MF-DFA

A continuación, se va a realizar el análisis *MF-DFA* de los 3 índices de alimentación sostenible y del *SP500*. Primero se analizarán de manera genérica los resultados de los 2 periodos y luego se procederá a un análisis más detallado y comparativo de ambos.

Como previamente comentado, el COVID19 ha supuesto un antes y un después en el mercado de la alimentación sostenible, es por ello que se considera relevante separar el horizonte temporal en 2 periodos.

Como nota aclaratoria, debido a que el periodo durante el COVID19 es un periodo breve (1 año en 2020) los resultados hubieran salido erróneos debido a que la muestra es muy pequeña. La técnica *MF-DFA* está pensada para realizarse sobre una gran muestra con lo que no es efectiva para un periodo corto. Por eso, el análisis se va a basar en una comparativa de la eficiencia del mercado pre y post COVID19. La división de los datos se ha realizado de la siguiente manera:

La división pre-COVID19 consiste en los datos desde el 2015 hasta el 2020, y la post-COVID19 consiste desde el 2020 hasta el 2025. El índice *MVIS*, debido a que se creó en el 2020, únicamente se analizará en el periodo post-COVID.

Para poder hacer una comparativa con un mercado plenamente eficiente, se ha creado una serie sintética de *fractional Gaussian noise* (*fGn*). El *fGn* es un proceso estocástico que permite modelar diferentes grados de dependencia temporal en una serie, controlados por el exponente de Hurst. Cuando el valor de Hurst $H \approx 0.5$, el *fGn* representa un comportamiento completamente aleatorio y sin memoria, equivalente a un mercado eficiente según la hipótesis de [Fama \(1970\)](#), donde los precios reflejan toda la información disponible y los retornos son impredecibles ([Mandelbrot, 1968](#)). De este modo, cualquier desviación del patrón multifractal del *fGn* puede interpretarse como evidencia de ineficiencias de mercado, como correlaciones de largo alcance, impactos exógenos o asimetrías en la distribución de los retornos. Al comparar el índice con un modelo teórico claro, se comprueba de forma práctica qué tan eficiente es y permite medir cuánto se aleja de esa referencia.

Por último, se ha realizado sobre los 4 índices y 2 periodos (menos el *MVIS* pre-COVID19) un *shuffling* de los datos. La técnica de *shuffling*, o reordenamiento aleatorio, consiste en desorganizar la secuencia temporal de una serie de datos sin modificar su distribución marginal. En otras palabras, se conservan los mismos valores, pero se eliminan las correlaciones temporales que puedan existir entre ellos ([Kantelhardt et al., 2002](#)).

Según [Kantelhardt et al. \(2002\)](#), el uso del *shuffling* permite distinguir el origen de la

multifractalidad observada en una serie temporal. Si tras aplicar *MF-DFA* a la versión reordenada aleatoriamente de la serie original (es decir, sin correlaciones de largo alcance) se reduce o elimina la multifractalidad, se puede concluir que esta multifractalidad estaba asociada a la estructura de correlaciones temporales de los datos. Por el contrario, si la multifractalidad persiste incluso después del *shuffling*, entonces su origen está en la forma no gaussiana de la distribución de probabilidad de los datos, como colas pesadas o asimetrías.

Esta técnica permite identificar si la complejidad de una serie temporal está determinada por su dinámica interna (correlaciones) o por sus propiedades estadísticas (distribución).

Como métricas analizadas se han considerado el *alpha*, *spectral width* y *MDM*. La *Market Deficiency Measure (MDM)* es un indicador sencillo que agrupa en un solo valor cuánta “memoria” de largo plazo exhibe la serie de precios de un activo frente a un movimiento completamente aleatorio. Cuanto más se aleja ese valor de cero, mayor es la dependencia temporal (o multifractalidad) y un *MDM* cercano a cero señala en cambio un mercado casi perfectamente eficiente ([Mensi et al, 2019](#)).

$$MDM = \frac{1}{2} (|h_{min}(q) - 0.5| + |h_{max}(q) - 0.5|) \quad [10]$$

Resultados del MF-DFA sobre los 4 índices (*shuffling*) y fGn

Tabla 2: Resultados del MF-DFA sobre los índices MSCI, MVIS, FXBYFOOD, SP500 y fGn por periodos pre-COVID19 y post-COVID19. Fuente: elaboración propia a partir de los resultados obtenidos del análisis MF-DFA.

	Enero.2015 – enero.2020 (pre-COVID19)			Enero.2020 - enero.2025 (post-COVID19)		
	Spectral width	alpha	MDM	Spectral width	alpha	MDM
MSCI	0.789	0.541	0.218	0.948	0.528	0.264
<i>(shuffling)</i>	<i>(0.725)</i>	<i>(0.526)</i>	<i>(0.193)</i>	<i>(0.774)</i>	<i>(0.536)</i>	<i>(0.245)</i>
MVIS				1.037	0.635	0.352
<i>(shuffling)</i>				<i>(0.840)</i>	<i>(0.547)</i>	<i>(0.251)</i>
FXBYFOOD	0.611	0.489	0.225	1.022	0.557	0.258
<i>(shuffling)</i>	<i>(0.713)</i>	<i>(0.528)</i>	<i>(0.192)</i>	<i>(0.769)</i>	<i>(0.540)</i>	<i>(0.226)</i>
SP500	0.962	0.513	0.359	0.909	0.487	0.317
<i>(shuffling)</i>	<i>(0.746)</i>	<i>(0.541)</i>	<i>(0.219)</i>	<i>(0.841)</i>	<i>(0.563)</i>	<i>(0.270)</i>
fGn	0.632	0.504	0.138	0.632	0.504	0.138

Periodo 1: Enero, 2015 – Enero, 2020 (Pre-COVID)

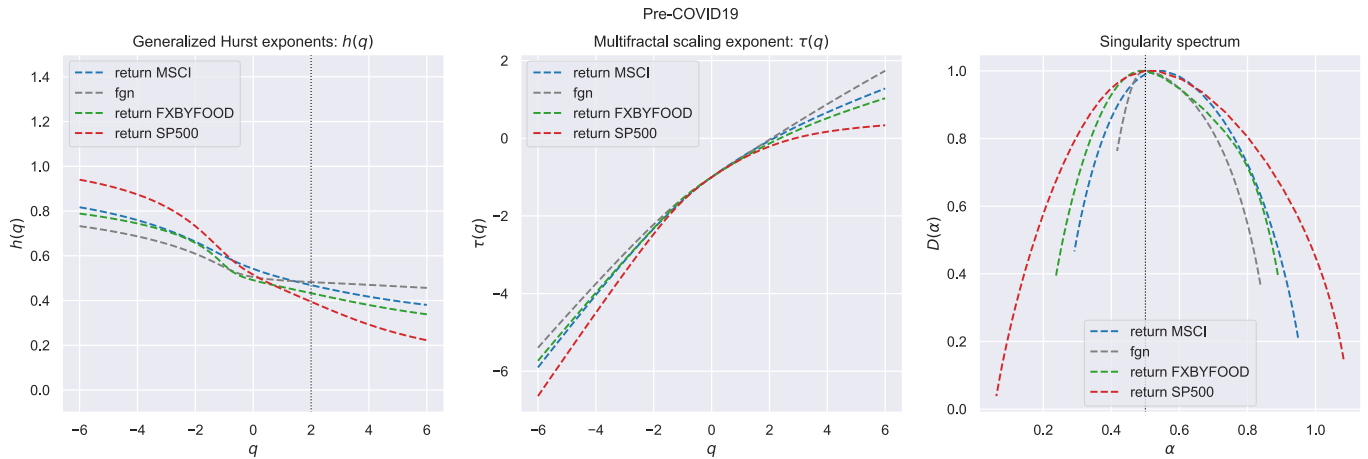


Ilustración 11: Resultados MF-DFA sobre los índices MSCI, FXBYFOOD (alimentación sostenible), SP500 y fGn. Periodo: pre-COVID19.

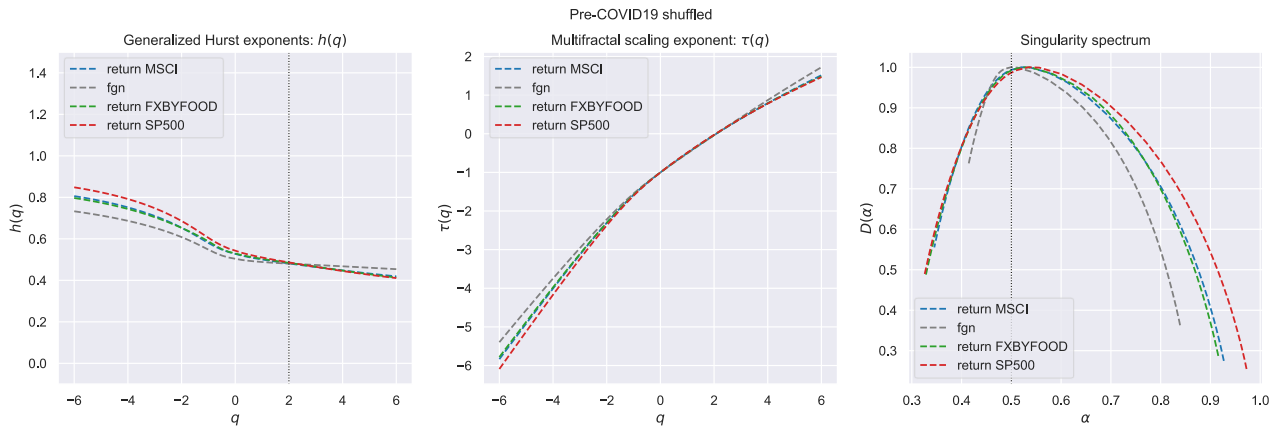


Ilustración 12: Resultados MF-DFA sobre los índices MSCI, FXBYFOOD (alimentación sostenible), SP500 y fGn. Periodo: pre-COVID19 shuffled.

Durante este periodo pre-COVID19, los índices muestran un comportamiento moderadamente eficiente. El SP500 presenta un *spectral width* elevado (0.962) pero un *alpha* cercano a 0.5 (0.513) y un MDM más alto que el de los índices sostenibles, lo que sugiere más multifractalidad. En cambio, MSCI y FXBYFOOD tienen *spectral width* menores (0.789 y 0.611), pero también *alpha* por debajo de 0.5 (indicando anti-persistencia parcial). Sin embargo, al observar el *shuffling*, vemos que los valores bajan levemente, lo que indica que hay correlación temporal pero no excesiva que afecta a la anti-persistencia.

Los resultados de fGn sirven como referencia: *spectral width* 0.632, *alpha* 0.504 y MDM 0.138. En comparación, los índices sostenibles no se alejan dramáticamente de esta base,

lo que sugiere un comportamiento de los índices en el periodo pre-COVID19 no plenamente eficiente, pero tampoco ineficiente.

En conclusión, en el primer periodo pre-COVID19 no se observa ni persistencia ni anti-persistencia marcada en los índices de alimentación sostenible. También al aplicar el shuffling se puede observar en la ilustración 7 en la gráfica del espectro de singularidad, como los *peak alpha* se acercan considerablemente a 0.5 asociando esa multifractalidad que se observa en la Ilustración 6, a correlaciones temporales de largo alcance.

Periodo 2: Enero, 2020 – Enero, 2025 (Post-COVID)

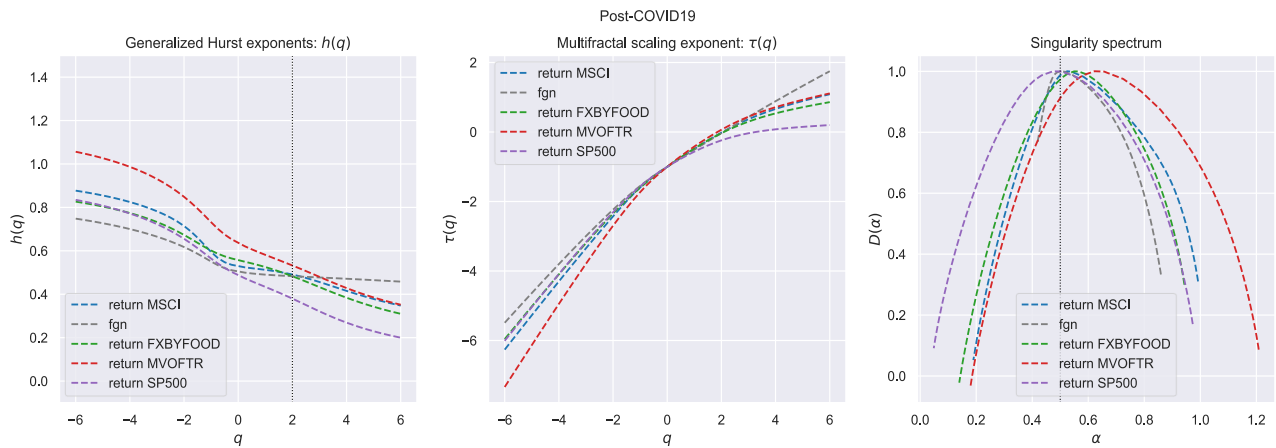


Ilustración 14: Resultados MF-DFA sobre los índices MSCI, FXBYFOOD, MVIS (alimentación sostenible), SP500 y fGn. Periodo: post-COVID19.

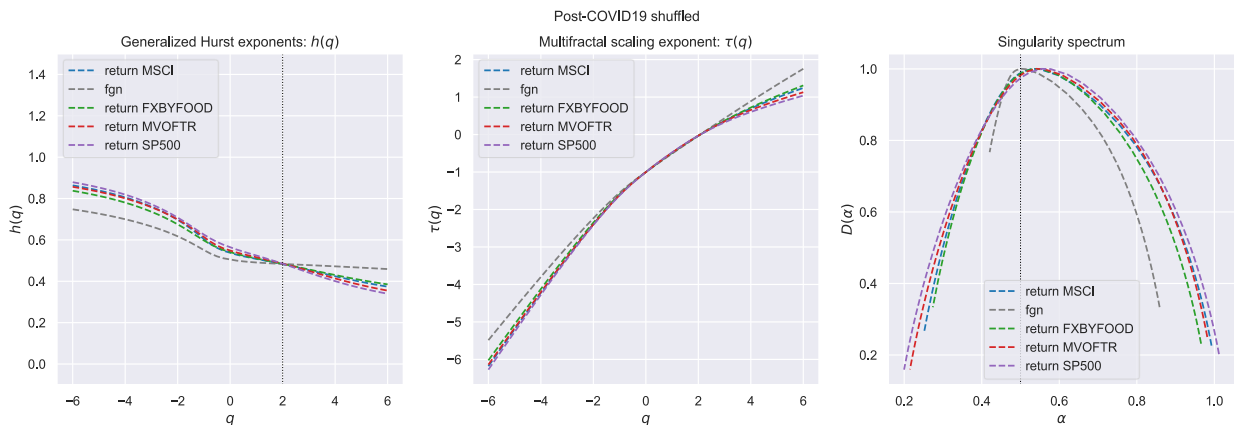


Ilustración 13: Resultados MF-DFA sobre los índices MSCI, FXBYFOOD, MVIS (alimentación sostenible), SP500 y fGn. Periodo: post-COVID19 shuffled.

En el segundo periodo, (post-COVID19) todos los índices muestran un cierto

empeoramiento menos el índice *MSCI*. El spectral width del *MSCI* baja de 1.149 a 0.949, el α baja también, y la *MDM* disminuye a 0.262. En *FXBYFOOD* ocurre algo similar. Sin embargo, en el *MVIS*, el spectral width se mantiene alto (0.885), aunque no se disponen de datos pre-COVID para comparar.

En todos los casos, al ver la comparativa entre los valores originales y los *shuffling* se observa cómo se reducen los valores, especialmente en los índices sostenibles. Esto sugiere que la estructura de correlaciones de largo plazo se ha reducido, y que los precios tienden a comportarse de forma más aleatoria y eficiente, aunque aún con ligeras huellas multifractales. El *SP500* también retoma un nivel de spectral width intermedio (0.909) y mantiene valores de Hurst estables ($\alpha = 0.487$), cercanos al ideal de 0.5.

Los resultados del periodo 2, reflejan que los índices de alimentación sostenible no presentan una ineficiencia estructural profunda, pero sí han cambiado (son más ineficientes), esto es debido a una mayor sensibilidad a contextos exógenos, en este caso la pandemia. Su multifractalidad se intensifica durante los momentos de crisis, lo que indica una mayor complejidad en la dinámica de precios (posiblemente ligada a su menor capitalización, menor liquidez y concentración temática).

El uso de la técnica *shuffling* permite identificar que una parte significativa de la multifractalidad observada en estos índices proviene de la correlación temporal, como se observa en el índice *MVIS*, que previo a aplicar la técnica de “*barajado*” presentaba gran multifractalidad y después su α se reducía considerablemente.

Evolución de la eficiencia del mercado sostenible y del SP500 pre-COVID19, y post-COVID19

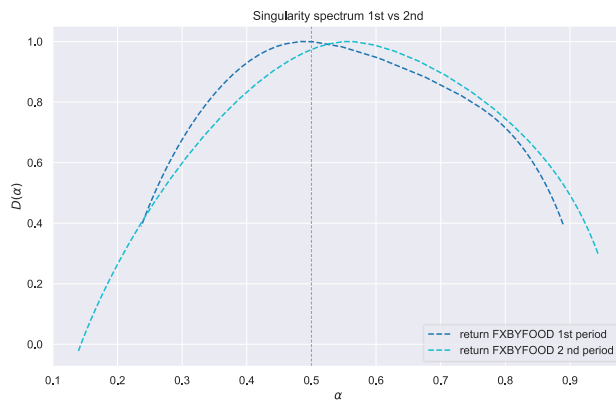


Ilustración 16: Comparativa del índice FXBYFOOD entre el primer y segundo periodo.

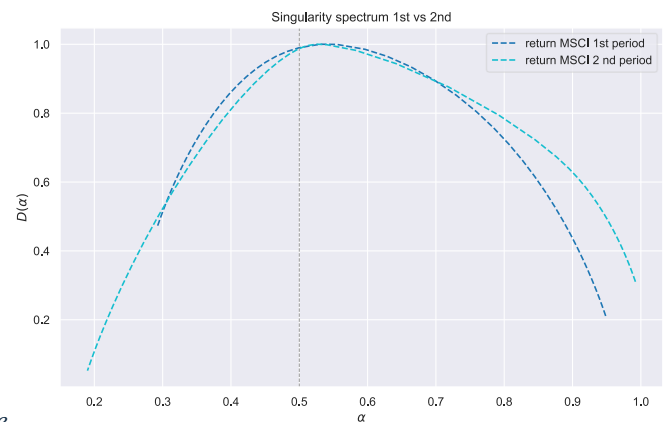


Ilustración 15: Comparativa del índice MSCI entre el primer y segundo periodo

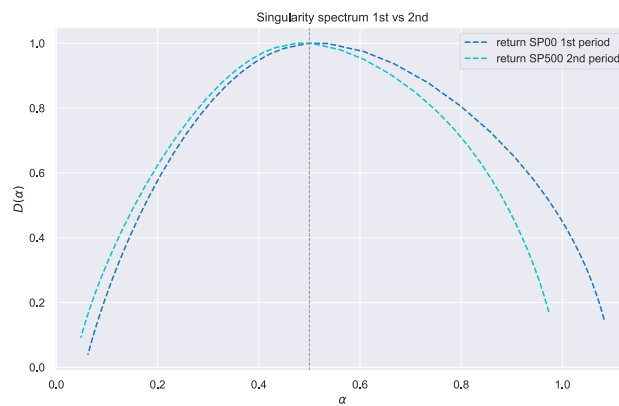


Ilustración 17: Comparativa del índice SP500 entre el primer y segundo periodo

A partir del análisis de las tres figuras que dan el análisis entre periodos (Ilustración 15, Ilustración 16 e Ilustración 17) se puede observar de forma clara la evolución dinámica de la eficiencia de mercado en los índices de alimentación sostenible (*MSCI*, *FXBYFOOD*) en comparación con el índice de referencia *SP500*. Del periodo pre al post-COVID19, los índices sostenibles perdieron algo de eficiencia mientras que el *SP500* se mantuvo igual o incluso más eficiente.

Pre-COVID19 normal vs pre-COVID19 *shuffled*

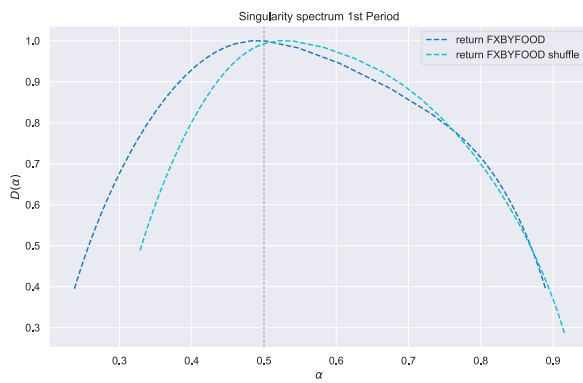


Ilustración 18: Comparativa del índice FXBYFOOD normal y shuffled en el primer periodo.

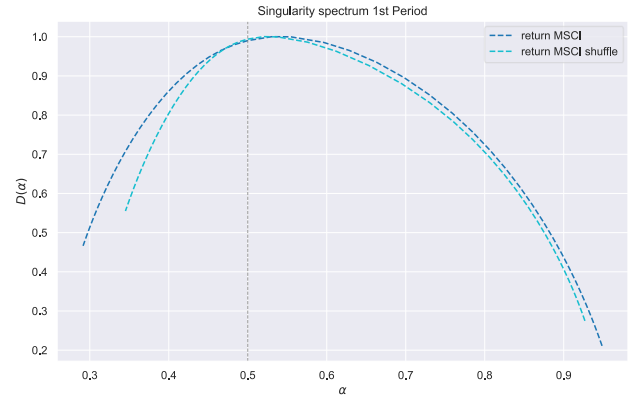


Ilustración 20: Comparativa del índice MSCI normal y shuffled en el primer periodo.

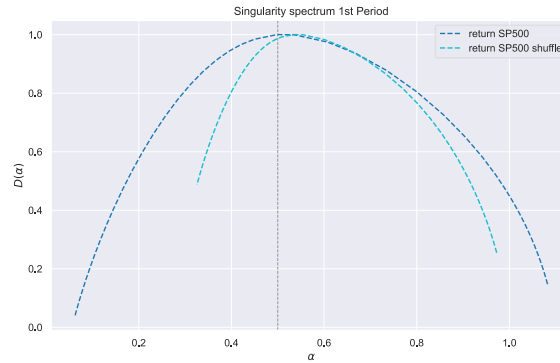


Ilustración 19: Comparativa del índice SP500 normal y shuffled en el primer periodo.

Durante el periodo pre-COVID19, los espectros de singularidad $D(\alpha)$ muestran una estructura más amplia y asimétrica en los índices *MSCI* y *FXBYFOOD* respecto al *SP500* (Ilustración 18, Ilustración 19 e Ilustración 20). lo que indica una mayor multifractalidad y, por tanto, una menor eficiencia relativa. El hecho de que estas curvas se alejen más del espectro de referencia del *fGn* (observado en la Ilustración 11) sugiere la presencia de correlaciones de largo alcance y de una dinámica más irregular, típica de mercados menos desarrollados o más susceptibles a factores exógenos. Aun así, los espectros no muestran distorsiones extremas, por lo que se puede hablar de mercados con un grado intermedio de eficiencia, caracterizado por fases puntuales de ineficiencia, pero no estructurales.

La técnica del *shuffling* en los tres índices reduce la multifractalidad, pero no

significativamente (excepto en el *SP500*) con lo que se puede concluir que en el periodo pre-COVID19 los 3 índices presentaban una estructura altamente eficiente.

Post-COVID19 normal vs post-COVID19 *shuffled*

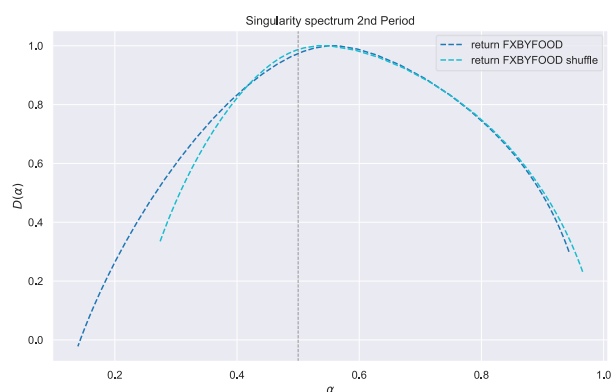


Ilustración 24: Comparativa del índice *FXBYFOOD* normal y *shuffled* en el segundo periodo.

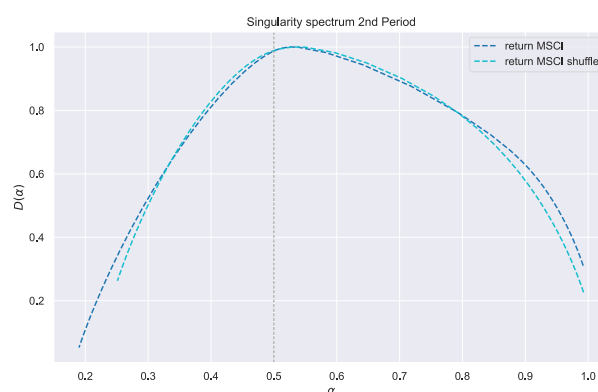


Ilustración 23: Comparativa del índice *MSCI* normal y *shuffled* en el segundo periodo.

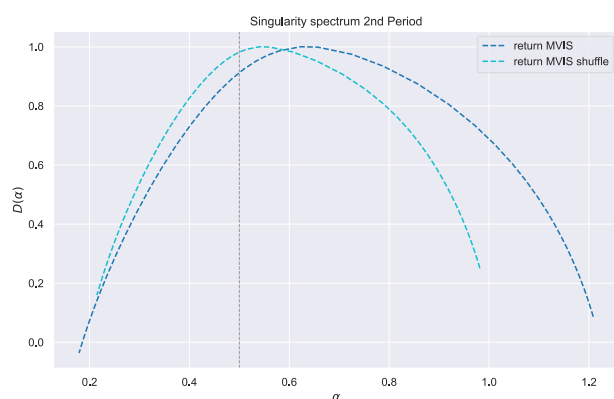


Ilustración 22: Comparativa del índice *MVIS* normal y *shuffled* en el segundo periodo.

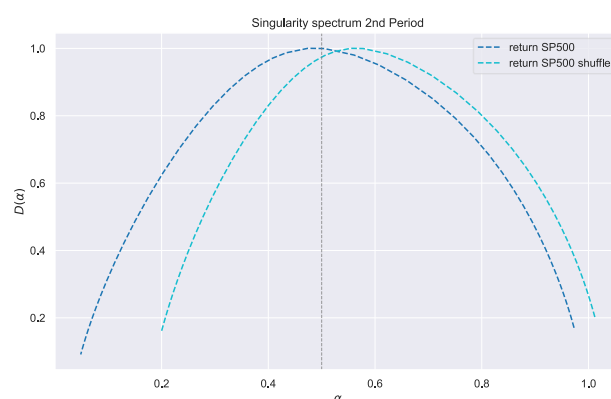


Ilustración 21: Comparativa del índice *SP500* normal y *shuffled* en el segundo periodo.

En las gráficas superiores, para cada índice, está dibujado el espectro de singularidad $D(\alpha)$ de la serie original (línea continua) frente al de la serie barajada o “*shuffled*” (línea punteada). La anchura de ese espectro mide la multifractalidad: cuanto más amplio, mayor diversidad de escalas y niveles fractales coexistentes en la dinámica de precios.

Para los índices *FXBYFOOD*, *MSCI* y *MVIS*, al barajar la serie el espectro se estrecha de manera notable. Esto sucede porque buena parte de su multifractalidad proviene de las correlaciones de largo plazo que existen entre precios sucesivos. Al destruir esa memoria

temporal, queda únicamente la contribución residual de la distribución de retornos, que por sí sola genera una multifractalidad mucho más reducida.

En el caso del *SP500* se observa (Ilustración 21) el efecto inverso: el espectro se ensancha al aplicar *shuffling*. Puede explicarse por el peso preponderante de las colas gordas y asimetrías de la distribución de retornos en su multifractalidad. Al eliminarse cualquier estructura temporal sutil que en la serie original moderaba la variabilidad, afloran con fuerza los saltos extremos y la heterogeneidad pura de la distribución, ampliando así la multifractalidad aparente.

El *shuffling* actúa como un filtro de memoria: tiende a reducir la multifractalidad basada en correlaciones temporales, pero al mismo tiempo deja ver con mayor nitidez la multifractalidad intrínseca a las colas de la distribución. En los tres primeros índices el balance queda claramente a favor de la reducción, mientras que en el *SP500* el carácter extremo de sus retornos prevalece y amplifica el espectro.

En conjunto, se puede concluir que la eficiencia de los índices sostenibles se vio afectada durante la pandemia por un aumento de la complejidad multifractal, atribuible a shocks exógenos como la disrupción de cadenas de suministro y la incertidumbre financiera global. Sin embargo, observando los resultados después del *shuffling* se puede decir que, pese a que no alcanzan una eficiencia perfecta, hay una evolución que sugiere un ajuste progresivo hacia dinámicas más estables y predecibles.

8.-Conclusión

Este trabajo partía de una pregunta sencilla pero esencial: ¿son los índices de alimentación sostenible eficientes? Con un análisis riguroso de sus dinámicas de mercado y rendimiento, se ha querido mostrar que preocuparse por el medio ambiente y la sociedad no está reñido únicamente con obtener buenos resultados financieros. Un mercado eficiente incorpora de inmediato toda la información ESG en los precios, alinea incentivos hacia prácticas responsables y ajusta las primas de riesgo, impulsando así retornos sólidos y sostenibles a largo plazo.

La eficiencia del mercado de la alimentación sostenible, tal y como han reflejado los análisis y conclusiones de las técnicas fractales y multifractales utilizadas, revela un

sistema en evolución, con signos de maduración progresiva pero aún con ciertas ineficiencias estructurales. Los resultados obtenidos muestran que los índices especializados en sostenibilidad alimenticia (*MSCI*, *MVIS* y *Foxberry*) presentan una mayor sensibilidad a shocks externos y una volatilidad más marcada en comparación con referentes consolidados como el *SP500*. No obstante, esa misma complejidad y variabilidad no debe interpretarse como debilidad, sino como una señal de la juventud del sector y de su potencial transformador dentro de un paradigma económico que prioriza la sostenibilidad y la resiliencia.

Desde la perspectiva de los Objetivos de Desarrollo Sostenible, el ODS 2 ("Hambre cero") y el ODS 12 ("Producción y consumo responsables"), los mercados financieros juegan un papel esencial en canalizar inversiones hacia modelos alimentarios más justos, accesibles y sostenibles. La evolución de los índices analizados muestra que, pese a sus limitaciones actuales, estos instrumentos financieros ya permiten medir el impacto y el comportamiento de un ecosistema de empresas comprometidas con la innovación agroalimentaria, la seguridad nutricional y la agricultura sostenible. Su inclusión en carteras sostenibles es coherente con los principios *ESG*, dado que las compañías que componen estos índices operan bajo estándares ambientales, sociales y de gobernanza que reflejan una visión de largo plazo y una apuesta por la sostenibilidad como ventaja competitiva.

Por tanto, este trabajo concluye que, aunque el mercado de la alimentación sostenible no ha alcanzado aún niveles plenos de eficiencia, muestra un desarrollo positivo en esa dirección y ofrece oportunidades reales de inversión responsable. Lejos de representar únicamente un enfoque ético o filantrópico, invertir en alimentación sostenible puede tener fundamentos financieros sólidos, con potencial de generación de valor tanto económico como social y ambiental. Que estos mercados funcionen con mayor eficiencia, no resuelve únicamente un reto técnico: también permite construir el cimiento de un sistema alimentario más justo, capaz de resistir tempestades y de convivir en armonía con los frágiles equilibrios de nuestro planeta.

9.-Bibliografía

- Allen, T., & Prosperi, P. (2016). Modeling sustainable food systems. *Environmental Management*, 57(5), 956–975. <https://doi.org/10.1007/s00267-016-0664-8>
- Baker, S. R., Bloom, N., Davis, S. J., & Terry, S. J. (2020). *COVID-induced economic uncertainty* (No. w26983). National Bureau of Economic Research. <https://doi.org/10.3386/w26983>
- Baldwin, R., & di Mauro, B. W. (2020). *Economics in the Time of COVID19*. CEPR Press. <https://cepr.org/publications/books-and-reports/economics-time-COVID19>
- Barunik, J., Vacha, L., & Kristoufek, L. (2012). Modeling and Forecasting Persistent Financial Durations. *The European Physical Journal B*, 85(1), 3–15. <http://dx.doi.org/10.2139/ssrn.2108394>
- Berthouze, L., & Farmer, S. F. (2012). Adaptive time-varying detrended fluctuation analysis. *Journal of Neuroscience Methods*, 209(1), 178–188. <https://doi.org/10.1016/j.jneumeth.2012.05.030>
- Bryce, R. M., & Sprague, K. B. (2012). Revisiting detrended fluctuation analysis. *Scientific Reports*, 2, 315. <https://doi.org/10.1038/srep00315>
- Clark, G. L., Feiner, A., & Viehs, M. (2015). *From the stockholder to the stakeholder: How sustainability can drive financial performance*. SSRN. <https://doi.org/10.2139/ssrn.2508281>
- Calvet, L. E., & Fisher, A. J. (2002). Multifractality in asset returns: Theory and evidence. *The Review of Economics and Statistics*, 84(3), 381–406. <http://dx.doi.org/10.1162/003465302320259420>
- De, I. (2013). *Understanding the S&P 500: This Index Offers a Lot of International Exposure*. *Journal of Investment Consulting*, 14(2), 26–37. <https://ssrn.com/abstract=2371340>
- Deaton, B. J., & Deaton, B. J. (2020). Food security and Canada's agricultural system challenged by COVID19. *Canadian Journal of Agricultural Economics/Revue canadienne d'agroeconomie*, 68(2), 143–149. <https://doi.org/10.1111/cjag.12227>

- Ding, W., Levine, R., Lin, C., & Xie, W. (2021). Corporate immunity to the COVID19 pandemic. *Journal of Financial Economics*, 141(2), 802–830. <https://doi.org/10.1016/j.jfineco.2021.03.005>
- Eccles, R. G., Lee, L.-E., & Strohle, J. (2020). The social origins of ESG: An analysis of Innovest and KLD. *Organization & Environment*, 33(4), 575–596. <https://doi.org/10.1177/1086026619888994>
- Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, 25(2), 383–417. <https://doi.org/10.1111/j.1540-6261.1970.tb00518.x>
- Fang, F., & Parida, S. (2022). *Sustainable mutual fund performance and flow in the recent years through the COVID19 pandemic*. *International Review of Financial Analysis*, 84, 102387. <https://doi.org/10.1016/j.irfa.2022.102387>
- Food and Agriculture Organization (FAO). (2018). *Sistemas alimentarios sostenibles: Conceptos y marcos*. Organización de las Naciones Unidas para la Alimentación y la Agricultura. <http://www.fao.org/3/ca2079es/CA2079ES.pdf>
- Food and Agriculture Organization (FAO). (2019). *La FAO y los Objetivos de Desarrollo Sostenible: Cumplir la Agenda 2030 mediante el empoderamiento de las comunidades locales* (CC2063ES). FAO. Recuperado de <http://www.fao.org/3/CC2063ES/online/CC2063ES.html>
- Food and Agriculture Organization (FAO). (2024). *The state of food security and nutrition in the world*. FAO Publishing. <https://www.fao.org/publications/fao-flagship-publications/the-state-of-food-security-and-nutrition-in-the-world/en>
- Foxberry. (2015). *Foxberry Tematica Research Sustainable Future of Food Index*. Recuperado el 20 de febrero de 2025, de https://www.foxberry.com/indices/thematics/foxberry_tematica_sust_food_tr
- Foxberry Ltd. (2022, marzo). *Foxberry Tematica Research Sustainable Future of Food Index Rules* [Index methodology document]. https://www.solactive.com/wp-content/uploads/2022/03/Foxberry_Tematica_Research_Sustainable_Future_of_Food-2022-03-2.pdf

- Freitas Netto, S. V. de, Sobral, M. F. F., Ribeiro, A. R. B., & Soares, G. R. da L. (2020). *Concepts and forms of greenwashing: a systematic review*. Environmental Sciences Europe, 32, Article 19. <https://doi.org/10.1186/s12302-020-0300-3>
- Garel, A., & Petit-Romec, A. (2021). Investor rewards to environmental responsibility: Evidence from the COVID19 crisis. *Journal of Corporate Finance*, 72, 102673. <https://doi.org/10.1016/j.jcorpfin.2021.101948>
- Godfray, H. C. J., Beddington, J. R., Crute, I. R., Haddad, L., Lawrence, D., Muir, J. F., Pretty, J., Robinson, S., Thomas, S. M., & Toulmin, C. (2010). *Food security: The challenge of feeding 9 billion people*. Science, 327(5967), 812–818. <https://doi.org/10.1126/science.1185383>
- Gorjão, L. R., Hassan, G., Kurths, J., & Witthaut, D. (2022). *MFDFA: Efficient multifractal detrended fluctuation analysis in Python*. Computer Physics Communications, 273, 108254. <https://doi.org/10.1016/j.cpc.2021.108254>
- Goold, C., Pfuderer, S., James, W. H. M., Lomax, N., Smith, F., & Collins, L. M. (2020). A mathematical model of national-level food system sustainability. <https://doi.org/10.48550/arXiv.2012.08355>
- Granger, C. W. J., & Newbold, P. (1974). *Spurious regressions in econometrics*. Journal of Econometrics, 2(2), 111–120. [https://doi.org/10.1016/0304-4076\(74\)90034-7](https://doi.org/10.1016/0304-4076(74)90034-7)
- Grau-Carles, P. (2006). *Bootstrap testing for detrended fluctuation analysis*. Physica A: Statistical Mechanics and its Applications, 360(1), 89–98. <https://doi.org/10.1016/j.physa.2005.05.074>
- Heneghan, C., & McDarby, G. (2000). Establishing the relation between detrended fluctuation analysis and power spectral density analysis for stochastic processes. *Physical Review E*, 62(5), 6103–6110. <http://dx.doi.org/10.1103/PhysRevE.62.6103>
- High Level Panel of Experts (HLPE). (2014). *Food losses and waste in the context of sustainable food systems* (Report No. 8). Committee on World Food Security, FAO. https://www.researchgate.net/publication/263652466_Food_losses_and_waste_in_the_context_of_sustainable_food_systems

- Hobbs, J. E. (2020). Food supply chains during the COVID19 pandemic. *Canadian Journal of Agricultural Economics/Revue canadienne d'agroeconomie*, 68(2), 171–176. <https://doi.org/10.1111/cjag.12237>
- Hurst, H. E. (1951). Long-term storage capacity of reservoirs. *Transactions of the American Society of Civil Engineers*, 116(1), 770–799. <https://doi.org/10.1061/TACEAT.0006518>
- Iannone, B., Dutillo, P., & Gattone, S. A. (2024). Evaluating the resilience of ESG investments in European markets during turmoil periods. *arXiv*. <https://arxiv.org/abs/2501.03269>
- International Finance Corporation (IFC). (2008). *Who Cares Wins: Connecting Financial Markets to a Changing World (2004–2008 Issue Brief)*. Washington, DC: World Bank Group. <https://documents1.worldbank.org/curated/en/444801491483640669/pdf/113850-BRI-IFC-Breif-whocares-PUBLIC.pdf>
- ISO. (2018). *ISO Standards on ESG and Environmental Metrics*. <https://www.iso.org/home.html>
- Kantelhardt, J. W., Zschiegner, S. A., Koscielny-Bunde, E., Havlin, S., Bunde, A., & Stanley, H. E. (2002). Multifractal detrended fluctuation analysis of nonstationary time series. *Physica A: Statistical Mechanics and its Applications*, 316(1–4), 87–114. <https://doi.org/10.48550/arXiv.physics/0202070>
- Katugampola, N. (2020). El efecto mariposa y el COVID19: seis repercusiones para la inversión sostenible. *Morgan Stanley*. https://www.morganstanley.com/im/publication/insights/articles/article_thebutterflyeffect_es.pdf
- Malkiel, B. G. (2003). The Efficient Market Hypothesis and Its Critics. *Journal of Economic Perspectives*, 17(1), 59–82. <https://doi.org/10.1257/089533003321164958>
- Mandelbrot, B. B., & Van Ness, J. W. (1968). Fractional Brownian motions, fractional noises and applications. *SIAM Review*, 10(4), 422–437. <https://doi.org/10.1137/1010093>

- Mandelbrot, B. B. (1997). *Fractals and Scaling in Finance: Discontinuity, Concentration, Risk*. Springer. Recuperado de <https://citations.springernature.com/book?doi=10.1007/978-1-4757-2763-0>
- Mandelbrot, B. B. (1999). A multifractal walk down Wall Street. *Scientific American*, 280(2), 70–73. <https://doi.org/10.1038/SCIENTIFICAMERICAN0299-70>
- MarketVector. (2020). *MVIS Global Future of Food ESG Index*. Recuperado el 20 de febrero de 2025, de <https://www.marketvector.com/indexes/sector/mvis-global-future-of-food-esg>
- MarketVector Indexes GmbH. (2025). *MVIS® Global Future of Food ESG Index [Index factsheet]*. <https://www.marketvector.com/factsheets/download/MVFOF.d.pdf>
- Mensi, W., Lee, Y.-J., Al-Yahyaee, K. H., Sensoy, A., & Yoon, S.-M. (2019). Intraday downward/upward multifractality and long memory in Bitcoin and Ethereum markets: An asymmetric multifractal detrended fluctuation analysis. *Finance Research Letters*, 31, 19–25. <https://doi.org/10.1016/j.frl.2019.03.029>
- MSCI. (2013). *MSCI ACWI IMI Food Revolution Index*. Recuperado el 20 de febrero de 2025, de <https://www.msci.com/indexes/index/737018>
- MSCI Inc. (2025). *MSCI ACWI IMI Food Revolution Index (USD) [Index factsheet]*. Recuperado el 16 de junio de 2025, de <https://www.msci.com/documents/10199/255599/msci-acwi-imi-food-revolution-index-usd-net.pdf>
- Pástor, L., & Vorsatz, B. (2020). *Mutual fund performance and flows during the COVID19 crisis*. The Review of Asset Pricing Studies, 10(4), 791–833. <https://doi.org/10.1093/rapstu/raaa015>
- Pelster, M., Horn, M., & Oehler, A. (2024). Who cares about ESG? *Journal of Climate Finance*, 8(1), 100045. <https://doi.org/10.1016/j.jclimf.2024.100045>
- Peng, C. K., Buldyrev, S. V., Havlin, S., Simons, M., Stanley, H. E., & Goldberger, A. L. (1994). Mosaic organization of DNA nucleotides. *Physical Review E*, 49(2), 1685–1689. <https://doi.org/10.1103/PhysRevE.49.1685>
- Pollman, E. (2022). The making and meaning of ESG. *ECGI Law Working Paper No.*

659/2022, University of Pennsylvania. <https://ssrn.com/abstract=4219857>

Schmitt, F., Schertzer, D., & Lovejoy, S. (1999). Multifractal analysis of the foreign exchange market. *International Journal of Theoretical and Applied Finance*, 2(01), 125–150.

https://www.researchgate.net/publication/246898155_Multifractal_analysis_of_foreign_exchange_data

UNCTAD. (2021). Informe sobre las inversiones en el mundo 2021 (Panorama general). https://unctad.org/system/files/official-document/wir2021_overview_es.pdf

United Nations, Department of Economic and Social Affairs. (2020). *Transforming our world: the 2030 Agenda for Sustainable Development*. Recuperado de <https://sdgs.un.org/2030agenda>

U.S. Farm Bill. (1990). Food, Agriculture, Conservation, and Trade Act of 1990, *Public Law 101–624, Title XVI, Subtitle A, Section 1603*. <https://www.congress.gov/bill/101st-congress/senate-bill/2830>

Velten, S., Leventon, J., Jager, N., & Newig, J. (2015). What is sustainable agriculture? A systematic review. *Sustainability*, 7(6), 7833–7865. <https://doi.org/10.3390/su7067833>

Wang, L., Gao, X.-L., & Zhou, W.-X. (2023). Testing for intrinsic multifractality in the global grain spot market indices: A multifractal detrended fluctuation analysis. *Physica A: Statistical Mechanics and its Applications*, 483, 25–35. <https://doi.org/10.48550/arXiv.2306.10496>

Whittle, P. (1953). Estimation and information in stationary time series. *Arkiv för Matematik*, 2(5), 423–434. <http://dx.doi.org/10.1007/BF02590998>

10.- Anexos

10.1.- Métricas *MF-DFA*

fGn Métricas

Tabla 3: Métricas análisis fGn. Fuente: elaboración propia.

	slope	hq	tau	alpha	Dalpha
-6	1.73	0.73	-5.39	0.84	0.38
-4	1.69	0.69	-3.75	0.80	0.54
-2	1.61	0.61	-2.22	0.71	0.80
2	1.48	0.48	-0.04	0.47	0.97
4	1.47	0.47	0.88	0.44	0.90
6	1.46	0.46	1.74	0.42	0.77

MSCI Métricas pre-COVID19

Tabla 4: Métricas análisis MSCI, pre-COVID19. Fuente: elaboración propia.

	slope	hq	tau	alpha	Dalpha
-6	1.82	0.82	-5.90	0.95	0.22
-4	1.76	0.76	-4.04	0.91	0.41
-2	1.66	0.66	-2.33	0.78	0.76
2	1.47	0.47	-0.06	0.41	0.88
4	1.42	0.42	0.67	0.33	0.65
6	1.38	0.38	1.28	0.29	0.48

FXBYFOOD Métricas Pre-COVID19

Tabla 5: Métricas análisis FXBYFOOD, pre-COVID19. Fuente: elaboración propia.

	slope	hq	tau	alpha	Dalpha
-6	1.79	0.79	-5.73	0.89	0.40
-4	1.74	0.74	-3.98	0.86	0.54
-2	1.66	0.66	-2.32	0.79	0.73
2	1.43	0.43	-0.13	0.38	0.89
4	1.38	0.38	0.52	0.28	0.62
6	1.34	0.34	1.03	0.24	0.40

SP500 Métricas Pre-COVID19

Tabla 6: Métricas análisis SP500, pre-COVID19. Fuente: elaboración propia.

	slope	hq	tau	alpha	Dalpha
-6	1.94	0.94	-6.64	1.08	0.15
-4	1.87	0.87	-4.50	1.06	0.27

-2	1.73	0.73	-2.47	0.94	0.59
2	1.39	0.39	-0.21	0.28	0.78
4	1.29	0.29	0.17	0.12	0.30
6	1.22	0.22	0.33	0.06	0.05

MSCI Métricas post- COVID19

Tabla 7: Métricas análisis MSCI, post-COVID19. Fuente: elaboración propia.

	slope	hq	tau	alpha	Dalpha
-6	1.84	0.84	-6.06	1.00	0.05
-4	1.77	0.77	-4.08	0.96	0.23
-2	1.63	0.63	-2.26	0.82	0.62
2	1.41	0.41	-0.19	0.36	0.91
4	1.36	0.36	0.44	0.27	0.64
6	1.32	0.32	0.91	0.21	0.38

FXBYFOOD Métricas post-COVID19

Tabla 8: Métricas análisis FXBYFOOD, post-COVID19. Fuente: elaboración propia.

	slope	hq	tau	alpha	Dalpha
-6	1.83	0.83	-6.00	0.97	0.17
-4	1.77	0.77	-4.08	0.94	0.34
-2	1.66	0.66	-2.32	0.80	0.72
2	1.45	0.45	-0.09	0.39	0.87
4	1.39	0.39	0.58	0.29	0.57
6	1.35	0.35	1.09	0.24	0.34

MVIS Métricas post-COVID19

Tabla 9: Métricas análisis MVIS, post-COVID19. Fuente: elaboración propia.

	slope	hq	tau	alpha	Dalpha
-6	2.09	1.09	-7.53	1.23	0.12
-4	2.02	1.02	-5.07	1.21	0.24
-2	1.87	0.87	-2.74	1.09	0.56
2	1.54	0.54	0.08	0.47	0.87
4	1.48	0.48	0.93	0.39	0.63
6	1.45	0.45	1.67	0.36	0.46

SP500 Métricas post-COVID19

Tabla 10: Métricas análisis SP500, post-COVID19. Fuente: elaboración propia.

	slope	hq	tau	alpha	Dalpha
-6	1.90	0.90	-6.40	1.04	0.18

-4	1.84	0.84	-4.34	1.01	0.29
-2	1.70	0.70	-2.40	0.89	0.62
2	1.41	0.41	-0.18	0.35	0.87
4	1.36	0.36	0.44	0.27	0.65
6	1.32	0.32	0.93	0.23	0.46

11.-Declaración de Uso de Herramientas de Inteligencia Artificial Generativa en Trabajos Fin de Grado

ADVERTENCIA: Desde la Universidad consideramos que ChatGPT u otras herramientas similares son herramientas muy útiles en la vida académica, aunque su uso queda siempre bajo la responsabilidad del alumno, puesto que las respuestas que proporciona pueden no ser veraces. En este sentido, NO está permitido su uso en la elaboración del Trabajo fin de Grado para generar código porque estas herramientas no son fiables en esa tarea. Aunque el código funcione, no hay garantías de que metodológicamente sea correcto, y es altamente probable que no lo sea.

Por la presente, yo, Gabriel Domínguez, estudiante de E2 + Analytics de la Universidad Pontificia Comillas al presentar mi Trabajo Fin de Grado titulado "Alimentación sostenible: Análisis de la eficiencia de mercado mediante el uso del MF-DFA", declaro que he utilizado la herramienta de Inteligencia Artificial Generativa ChatGPT u otras similares de IAG de código sólo en el contexto de las actividades descritas a continuación:

1. **Brainstorming de ideas de investigación:** Utilizado para idear y esbozar posibles áreas de investigación.
2. **Referencias:** Usado conjuntamente con otras herramientas, como Science, para identificar referencias preliminares que luego he contrastado y validado.
3. **Corrector de estilo literario y de lenguaje:** Para mejorar la calidad lingüística y estilística del texto.
4. **Sintetizador y divulgador de libros complicados:** Para resumir y comprender literatura compleja.
5. **Revisor:** Para recibir sugerencias sobre cómo mejorar y perfeccionar el trabajo con diferentes niveles de exigencia.
6. **Traductor:** Para traducir textos de un lenguaje a otro.

Afirmo que toda la información y contenido presentados en este trabajo son producto de mi investigación y esfuerzo individual, excepto donde se ha indicado lo contrario y se han dado los créditos correspondientes (he incluido las referencias adecuadas en el TFG y he explicitado para que se ha usado ChatGPT u otras herramientas similares). Soy

consciente de las implicaciones académicas y éticas de presentar un trabajo no original y acepto las consecuencias de cualquier violación a esta declaración.

Fecha:

Firma: Gabriel Domínguez

12.- Código de Python

MF-DFA TFG Analytics

Autor: Gabriel Dominguez Lazaro

Fecha: 2025.02.19

```
In [ ]: %config InlineBackend.figure_format = 'png'
import pandas as pd
from pathlib import Path
import numpy as np
import matplotlib.pyplot as plt
from MFDFA import MFDFA
from numpy.polynomial.polynomial import polyfit
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import pandas as pd
import seaborn as sns
sns.set_theme()
from scipy.stats import skew, kurtosis, jarque_bera, shapiro
from fbm import FBM
from scipy.interpolate import CubicSpline
from scipy.optimize import minimize_scalar
```

1. Base de Datos

```
In [3]: FXVYF00D=pd.read_excel("DDBB/FXBYF00D.xlsx",index_col="date")
FXVYF00D["return FXBYF00D"]=np.log(FXVYF00D["FXBYF00D"] / FXVYF00D["FXBYF00D"].shift(1))
FXVYF00D["price"]=FXVYF00D["FXBYF00D"]
MSCI=pd.read_excel("DDBB/MSCI.xlsx", index_col="date")
MSCI["return MSCI"]=np.log(MSCI["MSCI"] / MSCI["MSCI"].shift(1))
MSCI["price"]=MSCI["MSCI"]
MVFOF=pd.read_excel("DDBB/MVFOF.xlsx", index_col="date")
MVFOF=MVFOF.drop(columns=["MVFOF"])
MVFOF["return MVOFTR"]=np.log(MVFOF["MVFOFTR"] / MVFOF["MVFOFTR"].shift(1))
SP500=pd.read_excel("DDBB/SP500.xlsx",index_col="Date")
SP500["return SP500"]=np.log(SP500["Price"] / SP500["Price"].shift(1))
MSCI.index = pd.to_datetime(MSCI.index)
FXVYF00D.index = pd.to_datetime(FXVYF00D.index)
MVFOF.index = pd.to_datetime(MVFOF.index)
SP500.index = pd.to_datetime(SP500.index)
df_analysis=pd.concat([MSCI,FXVYF00D,MVFOF,SP500],axis=1,join="outer")
df_analysis['return SP500'] = df_analysis['return SP500'].fillna(0)
```

2. Analisis descriptivo de los datos

```
In [3]: rets_analysis = df_analysis[['return MSCI', "return FXBYFOOD", "return MVOFT
R", "return SP500"]]
print(rets_analysis)
rets_analysis.describe().round(2)
```

	return MSCI	return FXBYFOOD	return MVOFTR	return SP500
2015-01-05	NaN	NaN	NaN	0.000000
2015-01-06	-0.009021	NaN	NaN	-0.007463
2015-01-07	0.001917	NaN	NaN	0.021341
2015-01-08	0.020873	NaN	NaN	0.018664
2015-01-09	0.000536	NaN	NaN	-0.011996
...	...	...	...	...
2025-02-13	0.013346	0.011244	0.017151	0.004606
2025-02-14	0.012836	NaN	-0.003663	-0.007235
2025-02-17	0.000456	NaN	0.001359	0.000000
2025-02-18	0.008594	NaN	-0.005946	0.006399
2025-02-19	-0.003817	NaN	-0.000933	0.006490

[2643 rows x 4 columns]

Out[3]:

	return MSCI	return FXBYFOOD	return MVOFTR	return SP500
count	2642.00	2563.00	1339.00	2643.00
mean	0.00	0.00	-0.00	0.00
std	0.01	0.01	0.01	0.01
min	-0.10	-0.09	-0.11	-0.13
25%	-0.01	-0.00	-0.01	-0.00
50%	0.00	0.00	0.00	0.00
75%	0.01	0.01	0.01	0.01
max	0.06	0.06	0.07	0.09

```

In [ ]: risk_free_rate = 0.025
results = []

# Analizo los 4 indices
for indice in rets_analysis.columns:
    data = rets_analysis[indice].dropna()

    if not data.empty:
        ann_return = (np.exp(data.mean() * 252) - 1) * 100
        volatility = data.std() * np.sqrt(252) * 100
        mean_log_return_annualized = data.mean() * 252
        sharpe_ratio = (mean_log_return_annualized - risk_free_rate) / (data.std() * np.sqrt(252))
        skewness = skew(data)
        kurt = kurtosis(data)
        jb_test_statistic, jb_p_value = jarque_bera(data)
        results.append({
            'Indice': indice,
            'Annualized Return (%)': f"{ann_return:.3f}%",
            'Annualized Volatility (%)': f"{volatility:.3f}%",
            'Sharpe Ratio': f"{sharpe_ratio:.3f}",
            'Skewness': f"{skewness:.3f}",
            'Kurtosis': f"{kurt:.3f}",
            'Jarque-Bera Test Statistic': f"{jb_test_statistic:.3f}",
            'Jarque-Bera p-value': f"{jb_p_value:.3f}"
        })

results_df = pd.DataFrame(results)
print("=" * 80)
print("Estadísticas Financieras Generales por Indice (2015-2025)".center(80))
print("=" * 80)

for index, row in results_df.iterrows():
    print(f"\n{'-' * 80}")
    print(f"Indice: {row['Indice']}".center(80))
    print(f"{'-' * 80}")
    for key, value in row.items():
        if key != 'Indice':
            print(f"{key:35}: {value}")

print("=" * 80)

results=[]

for indice in rets_analysis.columns:
    for year in range(2014, 2026):
        data = rets_analysis.loc[rets_analysis.index.year == year, indice].dropna()
        if not data.empty:
            ann_return = (np.exp(data.sum()) - 1) * 100
            volatility = data.std() * np.sqrt(252) * 100
            mean_log_return_annualized = data.mean() * 252
            sharpe_ratio = (mean_log_return_annualized - risk_free_rate) / (data.std() * np.sqrt(252))
            skewness = skew(data)
            kurt = kurtosis(data)
            jb_test_statistic, jb_p_value = jarque_bera(data)
            results.append({
                'Indice': indice,

```

```

        'Year': year,
        'Annualized Return (%)': f"{ann_return:.3f}%",
        'Annualized Volatility (%)': f"{volatility:.3f}%",
        'Sharpe Ratio': f"{sharpe_ratio:.3f}",
        'Skewness': f"{skewness:.3f}",
        'Kurtosis': f"{kurt:.3f}",
        'Jarque-Bera Test Statistic': f"{j_b_test_statistic:.3f}",
        'Jarque-Bera p-value': f"{j_b_p_value:.3f}"
    })

results_df = pd.DataFrame(results)

print("=" * 80)
print("Estadísticas Financieras Anuales por Índice y año (2015-2025)".center(80))
print("=" * 80)

for indice in results_df['Indice'].unique():
    print(f"\n{'-' * 80}")
    print(f"Indice: {indice}".center(80))
    print(f"{'-' * 80}")
    display_df = results_df[results_df['Indice'] == indice].drop(columns=[
        'Indice'])
    print(display_df.to_string(index=False))

print("=" * 80)

```

```

=====
=====
                          Estadísticas Financieras Generales por Indice (2015-2025)
=====
=====

-----
-----
                          Indice: return MSCI
-----
-----
Annualized Return (%)           : 5.705%
Annualized Volatility (%)       : 16.353%
Sharpe Ratio                    : 0.186
Skewness                        : -0.591
Kurtosis                        : 7.085
Jarque-Bera Test Statistic      : 5679.919
Jarque-Bera p-value             : 0.000

-----
-----
                          Indice: return FXBYF00D
-----
-----
Annualized Return (%)           : 2.414%
Annualized Volatility (%)       : 16.033%
Sharpe Ratio                    : -0.007
Skewness                        : -0.591
Kurtosis                        : 6.825
Jarque-Bera Test Statistic      : 5123.229
Jarque-Bera p-value             : 0.000

-----
-----
                          Indice: return MVOFTR
-----
-----
Annualized Return (%)           : -1.970%
Annualized Volatility (%)       : 16.975%
Sharpe Ratio                    : -0.264
Skewness                        : -0.724
Kurtosis                        : 11.444
Jarque-Bera Test Statistic      : 7423.612
Jarque-Bera p-value             : 0.000

-----
-----
                          Indice: return SP500
-----
-----
Annualized Return (%)           : 12.632%
Annualized Volatility (%)       : 18.719%
Sharpe Ratio                    : 0.502
Skewness                        : -0.750
Kurtosis                        : 13.284
Jarque-Bera Test Statistic      : 19680.235
Jarque-Bera p-value             : 0.000
=====
=====
=====
=====

```

Estadísticas Financieras Anuales por Índice y año (2015-2025)

=====

Indice: return MSCI

Year	Annualized Return (%)	Annualized Volatility (%)	Sharpe Ratio	Skewness	Kurtosis	Jarque-Bera Test Statistic	Jarque-Bera p-value	
2015	3.703%	14.695%	0.072	-0.30	5	0.844	11.661	0.003
2016	2.085%	16.421%	-0.031	-0.77	4	2.593	99.171	0.000
2017	35.962%	8.045%	3.390	0.22	7	1.131	16.107	0.000
2018	-9.596%	14.552%	-0.841	-0.16	3	0.665	5.969	0.051
2019	23.224%	12.980%	1.361	-0.25	1	1.855	40.139	0.000
2020	34.071%	25.849%	0.994	-1.40	2	8.806	932.434	0.000
2021	5.185%	14.446%	0.165	0.04	2	0.912	9.115	0.010
2022	-22.602%	21.999%	-1.242	0.30	7	0.622	8.265	0.016
2023	-6.324%	15.325%	-0.576	0.32	8	0.429	6.671	0.036
2024	0.646%	12.707%	-0.148	-0.28	5	0.275	4.374	0.112
2025	8.424%	11.869%	4.560	-0.53	9	-0.289	1.871	0.392

Indice: return FXBYFOOD

Year	Annualized Return (%)	Annualized Volatility (%)	Sharpe Ratio	Skewness	Kurtosis	Jarque-Bera Test Statistic	Jarque-Bera p-value	
2015	-1.755%	14.030%	-0.334	-0.35	7	0.735	8.940	0.011
2016	9.783%	14.531%	0.448	-0.95	4	4.965	307.670	0.000
2017	26.325%	8.202%	2.457	0.25	5	0.170	3.136	0.208
2018	-11.004%	13.614%	-1.057	-0.48	1	0.493	12.019	0.002
2019	24.078%	12.800%	1.477	-0.43	7	1.549	33.452	0.000
2020	20.504%	25.662%	0.602	-1.08	3	7.291	631.608	0.000
2021	3.071%	12.393%	0.034	-0.27	6	0.522	6.278	0.043
2022	-27.034%	22.815%	-1.448	0.13	9	0.372	2.336	0.311
2023	-1.476%	15.373%	-0.256	0.19	5	0.472	4.061	0.131
2024	-6.312%	12.985%	-0.675	-0.30	1	0.457	6.243	0.044

2025	1.267%	12.505%	0.593	-0.20
1 -0.852	1.184	0.553		

Indice: return MVOFTR

Year	Annualized Return (%)	Annualized Volatility (%)	Sharpe Ratio	Skewness
s Kurtosis	Jarque-Bera Test	Statistic	Jarque-Bera p-value	
2020	17.401%	25.312%	0.511	-1.16
5 10.381		1235.734	0.000	
2021	-10.669%	12.552%	-1.067	-0.07
8 0.506		3.056	0.217	
2022	-26.706%	19.259%	-1.693	0.13
1 0.682		5.785	0.055	
2023	-2.473%	13.170%	-0.374	-0.06
4 0.420		2.084	0.353	
2024	17.269%	10.729%	1.201	-0.11
9 0.824		8.006	0.018	
2025	2.338%	11.952%	1.183	-0.35
3 0.611		1.273	0.529	

Indice: return SP500

Year	Annualized Return (%)	Annualized Volatility (%)	Sharpe Ratio	Skewness
s Kurtosis	Jarque-Bera Test	Statistic	Jarque-Bera p-value	
2015	11.157%	20.578%	0.379	-0.57
5 2.720		94.133	0.000	
2016	12.811%	15.716%	0.582	0.05
5 1.306		18.689	0.000	
2017	4.895%	10.747%	0.198	-0.05
2 1.255		17.179	0.000	
2018	-1.509%	17.444%	-0.227	-0.49
2 2.657		87.324	0.000	
2019	31.249%	12.935%	1.836	-0.76
9 3.490		158.187	0.000	
2020	6.657%	34.449%	0.107	-0.83
0 8.926		899.890	0.000	
2021	36.528%	13.889%	1.984	-0.31
2 2.045		49.706	0.000	
2022	-14.163%	22.896%	-0.756	-0.14
6 0.281		1.772	0.412	
2023	20.025%	13.555%	1.121	-0.15
3 -0.068		1.068	0.586	
2024	31.543%	13.763%	1.734	-0.33
8 3.090		109.231	0.000	
2025	3.772%	12.222%	1.916	0.01
6 -0.509		0.391	0.822	

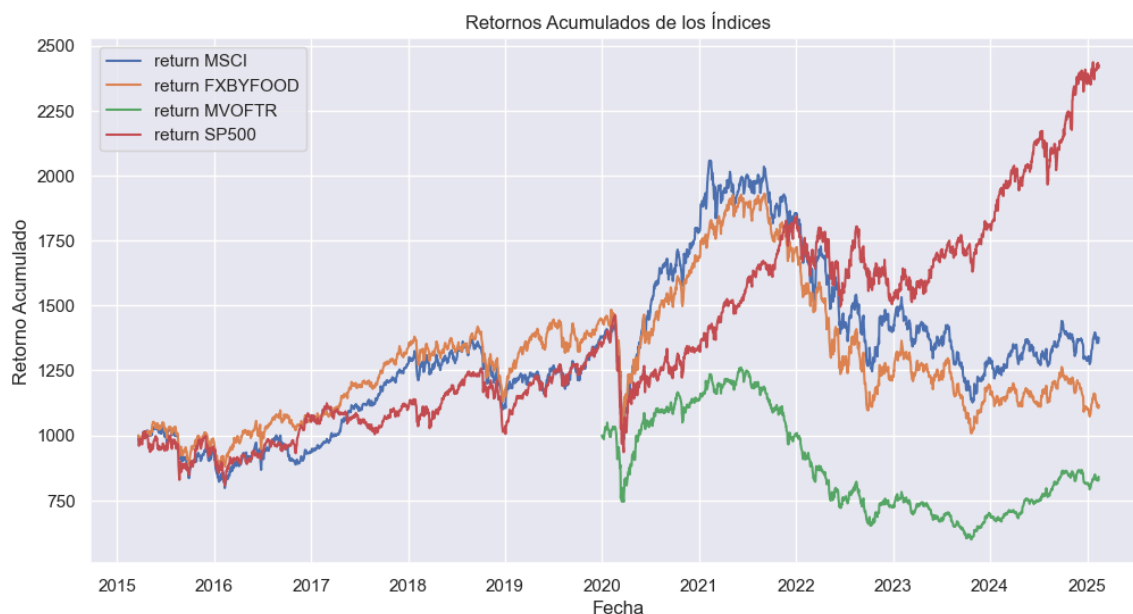
=====
=====


```
In [ ]: df_filtered=df_analysis.dropna(subset=["return FXBYFOOD"]).copy()

df_cumulative = 1000 * (1 + df_filtered[['return MSCI', "return FXBYFOOD",
"return MVOFTR", "return SP500"]]).cumprod()

plt.figure(figsize=(12, 6))
for column in df_cumulative.columns:
    plt.plot(df_filtered.index, df_cumulative[column], label=column)

plt.title("Retornos Acumulados de los Índices")
plt.xlabel("Fecha")
plt.ylabel("Retorno Acumulado")
plt.legend()
plt.grid(True)
#plt.savefig("Imagenes/Retornos_Acumulados.svg",format="svg")
plt.show()
```



3. Generación de White Noise O GAUSSIANO

Genero white noise con distribución normal (ruido gaussiano):

$\text{noise}(t) \sim N(\mu, \sigma)$

```
In [6]: def generate_fractional_gaussian_noise(hurst, length):
        """
        Generate fractional Gaussian noise with a given Hurst exponent
        """
        fbm_instance = FBM(n=length, hurst=hurst, length=1, method='daviessharte')

        fgn = fbm_instance.fgn()

        return fgn
```

4. DFA HURST

```

In [ ]: def get_dfa_HURST(data,
                        lag_start=None,
                        lag_end=None,
                        lag_steps=None,
                        q=None,
                        order=1,
                        integrate=True,
                        window=None):
    """
    Wrapper around the MFDFA package

    :param data: time series
    :param lag_start: minimum lag
    :param lag_end: maximum lag (if not provided, len(data)//6)
    :param lag_steps: how many lags to generate
    :param q: power q
    :param order: order of polynomial fitting
    :return:
    """

    def _integrate(data):
        mean = data.mean()
        return (data - mean).cumsum()

    def _power_variations(q_min, q_max, q_steps):
        q = np.linspace(q_min, q_max, q_steps)
        q = q[q != .0]
        return q

    data_length = len(data)

    if lag_start is None:
        # if lag_start is not provided explicitly,
        lag_start = 3

    if lag_end is None:
        # if lag_end is not provided explicitly,
        lag_end = data_length // 6

    if lag_steps is None:
        # if lag_steps is not provided explicitly,
        lag_steps = min(100, data_length // 6)

    lags = np.unique(np.logspace(np.log10(lag_start), np.log10(lag_end), lag_steps).astype(int))

    # q power variations, removing the 0 power
    if q is None:
        q = _power_variations(-5, +5, 21)

    # integrate series
    if integrate:
        integr_data = _integrate(data)
    else:
        # passthrough
        integr_data = data

    # Obtain the (MF)DFA as
    extensions = {}

```

```

if window:
    extensions['window'] = window

# using MFDFA package to calculate the multifractal DFA
lag, dfa = MFDFA(integr_data, lag=lags, q=q, order=order, extensions=ex
tensions)

# This "fine-tuning" should be done in a more robust way,
# by truncating the lag_start and lag_end values before and not here.
start = int(0.00 * lag.size)
end   = int(0.60 * lag.size)

# scales, you need to fit the logs of the results
slope = polyfit(np.log10(lag[start:end]), np.log10(dfa[start:end]), 1)
[1]

# Hurst exponent
hq = slope - 1

# q-order mass exponent
tau = q * hq - 1

# q-order singularity exponent (aka alpha)
alpha = np.gradient(tau) / np.gradient(q)

# Dimension (aka D(alpha))
Dalpha = q * alpha - tau

metrics = pd.DataFrame(index=q,
                        data={'slope': slope,
                              'hq': hq,
                              'tau': tau,
                              'alpha': alpha,
                              'Dalpha': Dalpha})

spectral_width = metrics['Dalpha'].max() - metrics['Dalpha'].min()

return q, lag, dfa, slope, hq, tau, alpha, Dalpha, metrics, spectral_wi
dth

```

```

In [ ]: def hurst_rolling(log_returns, window, step):

    def _calculate_hq(x):
        q, lag, dfa, slope, hq, tau, alpha, Dalpha, metrics, spectral_width
= get_dfa_HURST(
            x.values,
            q=[-2, 2],
            window=False)
        return metrics.loc[2, 'hq']

    result = log_returns.rolling(window=window, min_periods=window, step=ste
p).apply(_calculate_hq)

    return result

```

5. MF-DFA

```

In [ ]: def get_mfdfa(data,
                    lag_start=None,
                    lag_end=None,
                    lag_steps=None,
                    q=None,
                    order=1,
                    integrate=True,
                    window=None,
                    mode="normal",
                    iterations=1000,
                    verbose=False):
    """
    Wrapper around the MFDFA package

    :param data: time series
    :param lag_start: minimum lag
    :param lag_end: maximum lag (if not provided, len(data)//6)
    :param lag_steps: how many lags to generate
    :param q: power q
    :param order: order of polynomial fitting
    :return:
    """
    if mode not in ["fgn", "normal", "shuffle"]:
        raise Exception["Invalid mode {mode}"]

    def _integrate(data):
        mean = data.mean()
        return (data - mean).cumsum()

    def _power_variations(q_min, q_max, q_steps):
        q = np.linspace(q_min, q_max, q_steps)
        q = q[q != 0.0]
        return q

    data_length = len(data)

    if lag_start is None:
        # if lag_start is not provided explicitly,
        lag_start = 3

    if lag_end is None:
        # if lag_end is not provided explicitly,
        lag_end = data_length // 6

    if lag_steps is None:
        # if lag_steps is not provided explicitly,
        lag_steps = min(100, data_length // 6)

    # Logspace uses Log10
    lags = np.unique(np.logspace(np.log10(lag_start), np.log10(lag_end), la
g_steps).astype(int))

    if q is None:
        minq=-6
        maxq=6
        steps=4

        q = _power_variations(minq, maxq, 1+(maxq-minq)*steps)

```

```

# integrate series

if mode == "normal":
    iterations=1

# Obtain the (MF)DFA as
extensions = {}
if window:
    extensions['window'] = window

dfa_list=[]
for iter in range(iterations):
    if mode == "shuffle":
        data = np.random.permutation (data)

    if mode == "fgn":
        data = generate_fractional_gaussian_noise(0.5,len(data))

    if integrate:
        integr_data = _integrate(data)
    else:
        # passthrough
        integr_data = data

    lag, dfa = MFDFA(integr_data, lag=lags, q=q, order=order, extension
s=extensions)

    dfa_list.append(dfa)

if verbose:
    print(f'get_mfdfa():{mode=}{iterations=}')
dfa = np.mean(dfa_list,axis=0)

# This "fine-tuning" should be done in a more robust way,
# by truncating the lag_start and lag_end values before and not here.
start = int(0.00 * lag.size)
end   = int(1 * lag.size)

slope = polyfit(np.log10(lag[start:end]), np.log10(dfa[start:end]), 1)
[1]
# Hurst exponent
hq = slope - 1

# q-order mass exponent
tau = q * hq - 1 # slopes or hq??

# q-order singularity exponent (aka alpha)
alpha = np.gradient(tau) / np.gradient(q)

# Dimension (aka D(alpha))
Dalpha = q * alpha - tau

metrics = pd.DataFrame(index=q,
                        data={'slope': slope,
                              'hq': hq,
                              'tau': tau,
                              'alpha': alpha,
                              'Dalpha': Dalpha})

```

```

spectral_width = metrics['Dalpha'].max() - metrics['Dalpha'].min()
MDM=(abs(metrics["hq"].max()-0.5)+abs(metrics["hq"].min()-0.5))/2

# Encontrar el alpha que maximiza la interpolación de D(alpha)
alpha_1 = alpha[::-1]
Dalpaha_1 = Dalpaha[:::-1]
sorted_indices = np.argsort(alpha_1)
alpha_1 = alpha_1[sorted_indices]
Dalpaha_1 = Dalpaha_1[sorted_indices]

# Remove duplicates (if necessary)
alpha_1, unique_indices = np.unique(alpha_1, return_index=True)
Dalpaha_1 = Dalpaha_1[unique_indices]

print(alpha_1)
print(Dalpaha_1)
spline = CubicSpline(alpha_1, Dalpaha_1)
res = minimize_scalar(lambda x: -spline(x), bounds=(min(alpha_1), max(alpha_1)), method='bounded')
max_alpha = res.x # Alpha con D(alpha) máximo
slope=hq +1
return q, lag, dfa, slope, hq, tau, alpha, Dalpaha, metrics, spectral_width, max_alpha, MDM

```

6. Graficar Rolling Hurst

```

In [ ]: def plot_hurst_timeseries(rolling_hurst, filename=None, figsize=(12, 6), confidence_interval=[0.4, 0.6],name=None):

    figure, ax = plt.subplots(1, 1, figsize=figsize)

    sns.lineplot(rolling_hurst, ax=ax)
    ax.axhline(y=0.5, color='gray', linestyle='--')
    for value in confidence_interval:
        ax.axhline(y=value, color='gray', linestyle=':')
    ax.set(title=f'Rolling Hurst {name}', xlabel=r'date', ylabel=r'$h(2)$', ylim=[0, 1])

    if filename:
        plt.savefig(filename, format="svg", bbox_inches='tight')

    plt.tight_layout()
    plt.show()

    return figure

```

7. Graficar MF-DFA

```

In [ ]: def plot_multifractal(results_dict, filename=None, tittle=None):
    fig, axes = plt.subplots(1, 3, figsize=(18, 5)) # 1 fila, 3 columnas

    cmap = plt.cm.get_cmap('tab10')
    series_names = list(results_dict.keys())

    for idx, series_name in enumerate(series_names):
        res = results_dict[series_name]
        color = 'gray' if series_name == "fgn" else cmap(idx % 10)

        # 1.  $h(q)$  vs.  $q$ 
        axes[0].plot(res['q'], res['hq'], '--', color=color, label=series_name)
        axes[0].set(title='Generalized Hurst exponents:  $h(q)$ ',
                    xlabel=r'$q$', ylabel=r'$h(q)$', ylim=[-0.1, 1.5])

        # 2.  $\tau(q)$  vs.  $q$ 
        axes[1].plot(res['q'], res['tau'], '--', color=color, label=series_name)
        axes[1].set(title=r'Multifractal scaling exponent:  $\tau(q)$ ',
                    xlabel=r'$q$', ylabel=r'$\tau(q)$')

        # 3.  $D(\alpha)$  vs.  $\alpha$ 
        axes[2].plot(res['alpha'], res['Dalpha'], '--', color=color, label=series_name)
        axes[2].set(title='Singularity spectrum',
                    xlabel=r'$\alpha$', ylabel=r'$D(\alpha)$')

        # Línea vertical en  $h = 2$  en gráfico 1
        axes[0].axvline(x=2, color='black', linestyle=':', linewidth=0.8)

        # Línea vertical en  $\alpha = 0.5$  en gráfico 3
        axes[2].axvline(x=0.5, color='black', linestyle=':', linewidth=0.8)
        fig.suptitle(tittle, fontsize=12)

    for ax in axes:
        ax.legend(loc='best')

    if filename:
        plt.savefig(filename, format="svg")
    plt.tight_layout()

```

8. Cálculos


```
In [13]: #ruido_gaussiano.index = pd.to_datetime(ruido_gaussiano.index)
MSCI.index = pd.to_datetime(MSCI.index)
FXVYFOOD.index = pd.to_datetime(FXVYFOOD.index)
MVFOF.index = pd.to_datetime(MVFOF.index)
SP500.index = pd.to_datetime(SP500.index)
df_final=pd.concat([MSCI,FXVYFOOD,MVFOF,SP500],axis=1,join="outer")
log_rets = df_final[['return MSCI',"return FXBYFOOD","return MVOFTR","return SP500"]]
log_rets=log_rets[1:]
log_rets['return SP500'] = log_rets['return SP500'].fillna(0)
print(log_rets)
```

	return MSCI	return FXBYFOOD	return MVOFTR	return SP500
2015-01-06	-0.009021	NaN	NaN	-0.007463
2015-01-07	0.001917	NaN	NaN	0.021341
2015-01-08	0.020873	NaN	NaN	0.018664
2015-01-09	0.000536	NaN	NaN	-0.011996
2015-01-12	0.000860	NaN	NaN	-0.006729
...	...	...	...	...
2025-02-13	0.013346	0.011244	0.017151	0.004606
2025-02-14	0.012836	NaN	-0.003663	-0.007235
2025-02-17	0.000456	NaN	0.001359	0.000000
2025-02-18	0.008594	NaN	-0.005946	0.006399
2025-02-19	-0.003817	NaN	-0.000933	0.006490

[2642 rows x 4 columns]

8.1. DFA

```
In [ ]: for series_name in log_rets:

    data = log_rets[series_name].dropna()
    print(data)

    print(f'processing time series: {series_name}')

    WINDOW = 252
    STEP = 5
    roll_hurst = hurst_rolling(data, WINDOW, STEP)
    roll_hurst = roll_hurst.clip(lower=0.0, upper=1.0)

    plot_hurst_timeseries(roll_hurst, filename=f"{series_name}_Rolling_Hurst.svg", name=series_name)
```

```

2015-01-06    -0.009021
2015-01-07     0.001917
2015-01-08     0.020873
2015-01-09     0.000536
2015-01-12     0.000860

```

```

...
2025-02-13     0.013346
2025-02-14     0.012836
2025-02-17     0.000456
2025-02-18     0.008594
2025-02-19    -0.003817

```

Freq: B, Name: return MSCI, Length: 2642, dtype: float64
processing time series: return MSCI

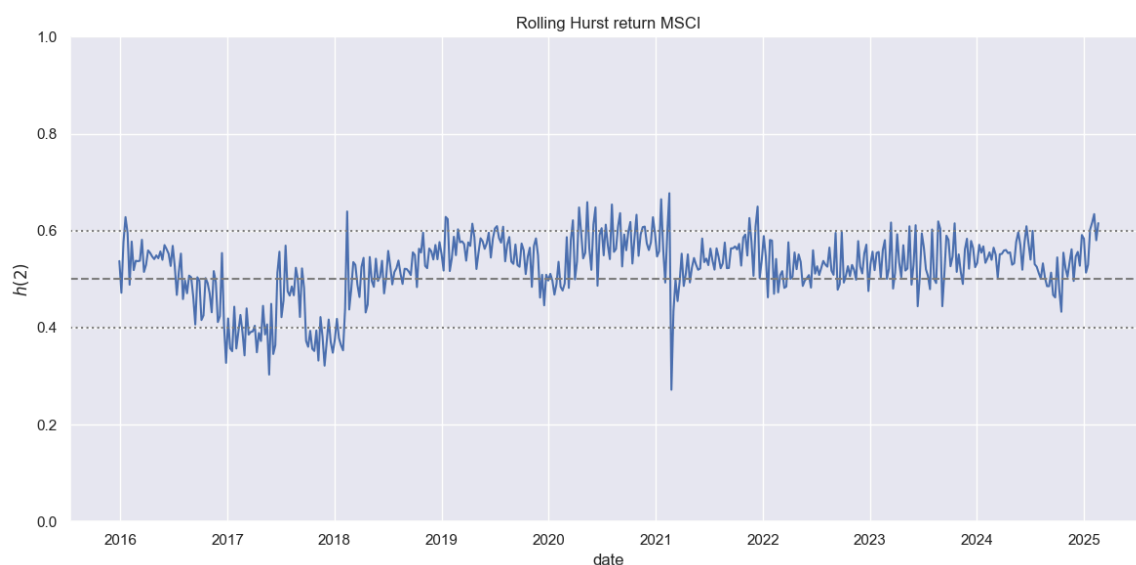
Exception ignored in: <function ZipFile.__del__ at 0x000002A66A40C720>
Traceback (most recent call last):

```

  File "C:\Program Files\WindowsApps\PythonSoftwareFoundation.Python.3.11_
3.11.2544.0_x64__qbz5n2kfra8p0\Lib\zipfile.py", line 1894, in __del__
    self.close()
  File "C:\Program Files\WindowsApps\PythonSoftwareFoundation.Python.3.11_
3.11.2544.0_x64__qbz5n2kfra8p0\Lib\zipfile.py", line 1896, in close
    def close(self):

```

KeyboardInterrupt:



```

2015-03-23     0.000490
2015-03-24    -0.003535
2015-03-25    -0.009524
2015-03-26    -0.006512
2015-03-27     0.001314

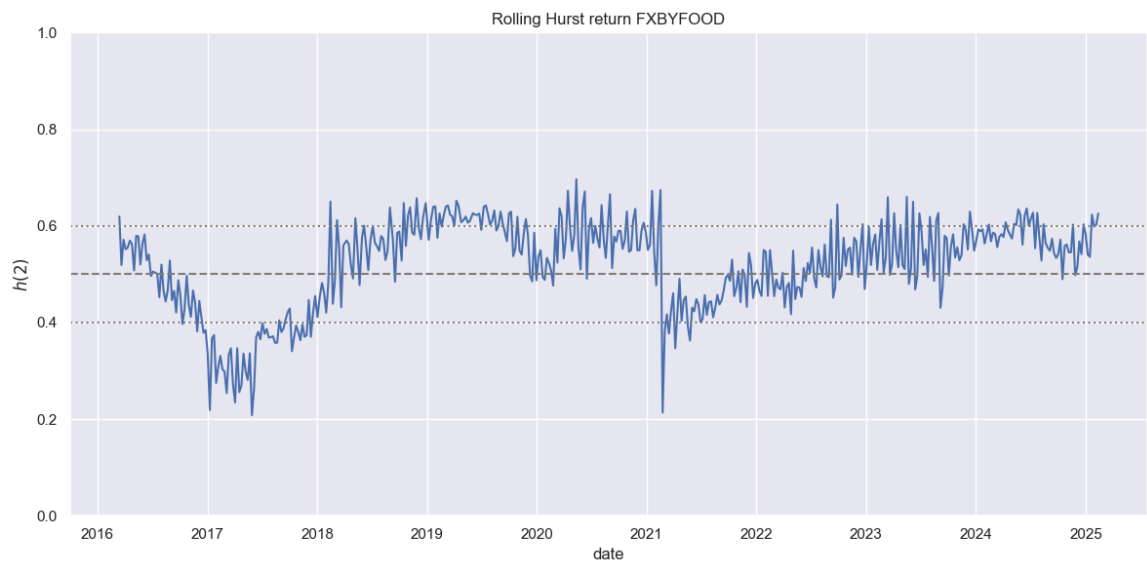
```

```

...
2025-02-07    -0.013035
2025-02-10    -0.000261
2025-02-11     0.004014
2025-02-12    -0.008289
2025-02-13     0.011244

```

Name: return FXBYFOOD, Length: 2563, dtype: float64
processing time series: return FXBYFOOD



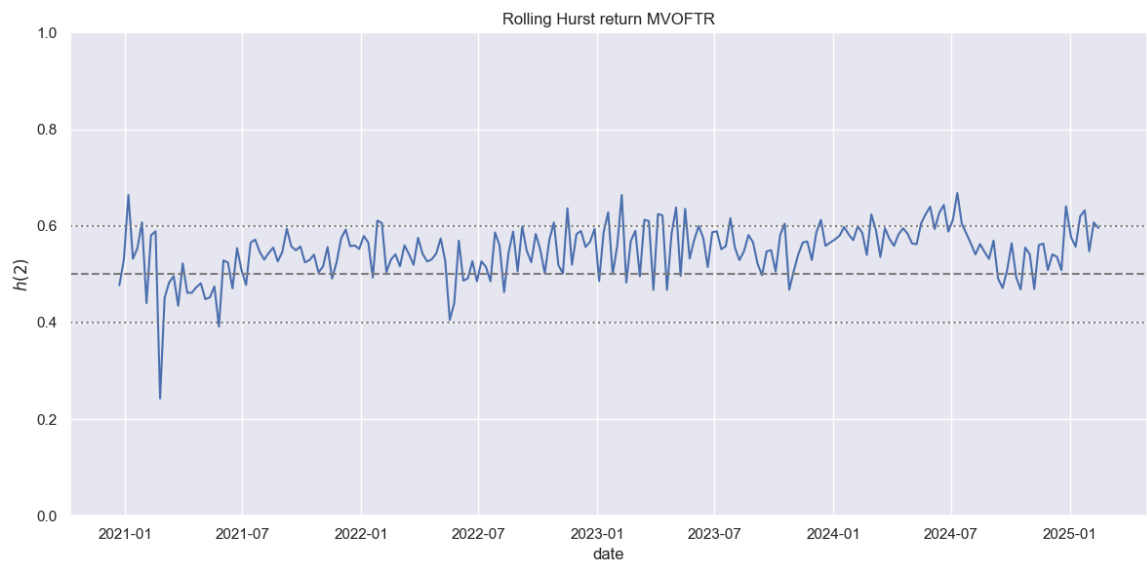
```
2020-01-01    0.000000
2020-01-02   -0.005435
2020-01-03   -0.000161
2020-01-06   -0.004445
2020-01-07    0.003418
```

...

```
2025-02-13    0.017151
2025-02-14   -0.003663
2025-02-17    0.001359
2025-02-18   -0.005946
2025-02-19   -0.000933
```

Name: return MVOFTR, Length: 1339, dtype: float64

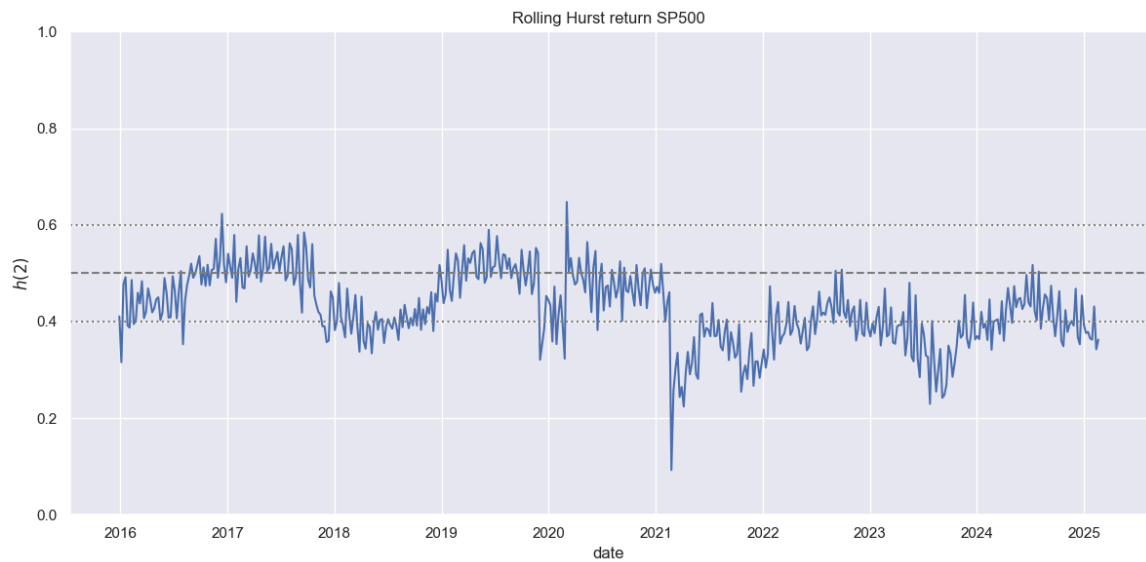
processing time series: return MVOFTR



```

2015-01-06    -0.007463
2015-01-07     0.021341
2015-01-08     0.018664
2015-01-09    -0.011996
2015-01-12    -0.006729
...
2025-02-13     0.004606
2025-02-14    -0.007235
2025-02-17     0.000000
2025-02-18     0.006399
2025-02-19     0.006490
Freq: B, Name: return SP500, Length: 2642, dtype: float64
processing time series: return SP500

```



8.2. MF-DFA

```
In [ ]: calculo_1 = log_rets.loc["2015-03-23":"2020-01-01"]
calculo_1=calculo_1.drop(columns=["return MVOFTR"])
calculo_1['return FXBYFOOD'] = calculo_1['return FXBYFOOD'].fillna(0)
calculo_1
```

Out[]:

	return MSCI	return FXBYFOOD	return SP500
2015-03-23	-0.002509	0.000490	-0.013900
2015-03-24	-0.004101	-0.003535	-0.004325
2015-03-25	-0.020324	-0.009524	-0.022067
2015-03-26	-0.010977	-0.006512	0.003920
2015-03-27	0.010156	0.001314	0.004935
...	...	...	...
2019-12-26	0.002390	0.002132	0.004349
2019-12-27	0.005190	0.001392	-0.005760
2019-12-30	-0.004662	-0.003671	-0.009419
2019-12-31	0.001672	0.000558	0.001025
2020-01-01	0.000203	0.000000	0.000000

1248 rows × 3 columns

```
In [36]: calculo_3 = log_rets.loc["2020-01-01":"2025-02-13"]
calculo_3['return FXBYFOOD'] = calculo_3['return FXBYFOOD'].fillna(0)
calculo_3
```

C:\Users\gabid\AppData\Local\Temp\ipykernel_23480\386565952.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
calculo_3['return FXBYFOOD'] = calculo_3['return FXBYFOOD'].fillna(0)

Out[36]:

	return MSCI	return FXBYFOOD	return MVOFTR	return SP500
2020-01-01	0.000203	0.000000	0.000000	0.000000
2020-01-02	0.001993	0.000292	-0.005435	0.010483
2020-01-03	-0.001380	-0.005691	-0.000161	-0.003865
2020-01-06	0.000898	-0.005233	-0.004445	0.001336
2020-01-07	-0.002757	0.000960	0.003418	0.001492
...	...	...	...	...
2025-02-07	-0.006139	-0.013035	-0.008754	-0.007095
2025-02-10	0.000888	-0.000261	-0.001399	0.009498
2025-02-11	0.000009	0.004014	0.003891	-0.002129
2025-02-12	0.001317	-0.008289	-0.004025	-0.005867
2025-02-13	0.013346	0.011244	0.017151	0.004606

1337 rows × 4 columns

```

In [ ]: all_results={}
i=0
for series_name in calculo_3:
    # Elimino valores NaN para cada una de las columnas
    data = calculo_3[series_name].dropna()

    print(f'processing time series: {series_name}')
    print(data)

    # Métricas fractales para el periodo
    q, lag, dfa, slope, hq, tau, alpha, Dalpha, metrics, spectral_width, ma
x_alpha, MDM = get_mfdfa(
    data.values,
    lag_start=None,
    lag_end=None,
    lag_steps=None,
    q=None,
    order=1,
    integrate=True,
    window=None,
    mode="shuffle",
    iterations=10000)
    print(f'{spectral_width=}')
    all_results[series_name] = {
        'q': q,
        'lag': lag,
        'dfa': dfa,
        'slope': slope,
        'hq': hq,
        'tau': tau,
        'alpha': alpha,
        'Dalpha': Dalpha,
        'metrics': metrics,
        'spectral_width': spectral_width,
        "max_alpha":max_alpha,
        "MDM":MDM
    }
    if i==0:
        series_name="fgn"
        print(f'processing time series: {series_name}')
        q, lag, dfa, slope, hq, tau, alpha, Dalpha, metrics, spectral_widt
h, max_alpha, MDM = get_mfdfa(
    data.values,
    lag_start=None,
    lag_end=None,
    lag_steps=None,
    q=None,
    order=2,
    integrate=True,
    window=None,
    mode="fgn",
    iterations=10000)
    print(f'{spectral_width=}')
    all_results[series_name] = {
        'q': q,
        'lag': lag,
        'dfa': dfa,
        'slope': slope,
        'hq': hq,
        'tau': tau,

```



```
'alpha': alpha,  
'Dalpha': Dalpha,  
'metrics': metrics,  
'spectral_width': spectral_width,  
"max_alpha":max_alpha,  
"MDM":MDM  
}  
i=i+1
```

Dibujar las gráficas

```
plot_multifractal(all_results,filename="MF-DFA 3rd period_final_shuffled.v  
2.svg",tittle="Post-COVID19 shuffled")
```

```

processing time series: return MSCI
2020-01-01    0.000203
2020-01-02    0.001993
2020-01-03   -0.001380
2020-01-06    0.000898
2020-01-07   -0.002757
...
2025-02-07   -0.006139
2025-02-10    0.000888
2025-02-11    0.000009
2025-02-12    0.001317
2025-02-13    0.013346
Freq: B, Name: return MSCI, Length: 1337, dtype: float64
[0.25109029 0.2534174 0.25843434 0.26422907 0.27090879 0.27858291
 0.28735551 0.29731428 0.3085156 0.32096669 0.33460734 0.34929569
 0.36480407 0.38083037 0.39702681 0.41304216 0.42856795 0.44337833
 0.4573573 0.47051468 0.48300049 0.49513497 0.507487 0.52981218
 0.55037242 0.59724574 0.64763726 0.70435903 0.75467659 0.793923
 0.82462456 0.84978509 0.87108852 0.8894155 0.90529756 0.91910505
 0.93112048 0.94157337 0.95065939 0.95855066 0.96540067 0.9713465
 0.97650984 0.98099789 0.98490424 0.98831008 0.9912854 0.99267352]
[0.26265274 0.27603362 0.3042086 0.33530338 0.36947821 0.40682395
 0.44732548 0.49081859 0.53694523 0.58511579 0.63449189 0.68400538
 0.73242624 0.77847992 0.82099624 0.85905467 0.89208619 0.91990587
 0.9426754 0.96081579 0.97489729 0.98552391 0.99320779 1.00036923
 0.99923031 0.98207263 0.94971835 0.9001551 0.84428007 0.79096791
 0.74149432 0.69459451 0.64953068 0.60617021 0.56461931 0.52504258
 0.48759856 0.45241118 0.41955409 0.38904505 0.36084931 0.33488879
 0.31105364 0.28921349 0.26922705 0.25094959 0.23423831 0.22625662]
spectral_width=np.float64(0.7741126100567812)
processing time series: fgn
[0.42031811 0.42204288 0.42554123 0.42912809 0.43278345 0.43648352
 0.44020098 0.4439055 0.4475645 0.45114439 0.45461222 0.45793779
 0.4610962 0.46407093 0.46685735 0.46946667 0.47193052 0.47430634
 0.47668443 0.47919798 0.48203948 0.48548992 0.48997649 0.50114343
 0.51353911 0.54531138 0.58120538 0.62131163 0.65747446 0.68686002
 0.71047151 0.72999668 0.74664849 0.76116522 0.7740054 0.7854733
 0.79578117 0.80508215 0.81349056 0.8210956 0.82797065 0.8341793
 0.83977899 0.84482298 0.84936126 0.85344088 0.85710596 0.8588402 ]
[0.76722228 0.77713971 0.79681177 0.81608622 0.83481633 0.85285216
 0.87004527 0.88625433 0.90135158 0.91522965 0.92780856 0.93904212
 0.94892327 0.95748747 0.96481351 0.97102109 0.97626451 0.98072243
 0.98458328 0.98802603 0.99119512 0.99416574 0.99688884 1.0003224
 0.9993734 0.98753518 0.96450983 0.92948193 0.88922659 0.84924076
 0.81117454 0.77477319 0.73953892 0.70517775 0.67156578 0.6386736
 0.60652849 0.57519655 0.54476875 0.51534695 0.4870309 0.45990783
 0.43404561 0.40948929 0.38626068 0.36435994 0.34376844 0.33379661]
spectral_width=np.float64(0.6665257887354685)
processing time series: return FXBYFOOD
2020-01-01    0.000000
2020-01-02    0.000292
2020-01-03   -0.005691
2020-01-06   -0.005233
2020-01-07    0.000960
...
2025-02-07   -0.013035
2025-02-10   -0.000261
2025-02-11    0.004014
2025-02-12   -0.008289
2025-02-13    0.011244
Freq: B, Name: return FXBYFOOD, Length: 1337, dtype: float64

```

```

[0.27396581 0.27621061 0.28100703 0.28645631 0.29263826 0.29963332
 0.30751741 0.31635448 0.32618677 0.33702327 0.34882785 0.36150969
 0.3749197 0.3888561 0.40308122 0.41734814 0.4314327 0.44516503
 0.45845604 0.47131817 0.48388606 0.4964501 0.5095261 0.53296196
 0.55298949 0.59330284 0.63104244 0.67406822 0.71716972 0.75609118
 0.78922801 0.81685011 0.83990812 0.85936907 0.8759873 0.89029747
 0.90267734 0.91340836 0.92271518 0.93078708 0.93778833 0.94386313
 0.94913811 0.95372399 0.95771705 0.96120054 0.96424607 0.96566761]
[0.33363576 0.34654334 0.37348483 0.40273147 0.43436564 0.46841326
 0.50481936 0.54342023 0.58391533 0.62584434 0.66857733 0.7113275
 0.75319384 0.79323511 0.83056582 0.86445476 0.89440223 0.9201766
 0.94180341 0.95951352 0.9736645 0.98464642 0.99276654 1.00031238
 0.99947758 0.98516259 0.96109454 0.92326701 0.87494833 0.82178332
 0.76830706 0.71683388 0.66808786 0.6220655 0.57860059 0.53758941
 0.49901306 0.46289123 0.42923665 0.39802975 0.3692117 0.34268814
 0.31833768 0.29602135 0.27559115 0.25689683 0.23979106 0.23161719]
spectral_width=np.float64(0.7686951894717411)
processing time series: return MVOFTR
2020-01-01    0.000000
2020-01-02   -0.005435
2020-01-03   -0.000161
2020-01-06   -0.004445
2020-01-07    0.003418
...
2025-02-07   -0.008754
2025-02-10   -0.001399
2025-02-11    0.003891
2025-02-12   -0.004025
2025-02-13    0.017151
Name: return MVOFTR, Length: 1335, dtype: float64
[0.21480415 0.21697711 0.22172642 0.22735248 0.23400229 0.24183436
 0.25101146 0.26168857 0.27399487 0.28800974 0.30373394 0.3210602
 0.33975033 0.35942983 0.37961147 0.39975322 0.41934158 0.43797593
 0.45542745 0.47166215 0.48684006 0.50131706 0.51568621 0.54017471
 0.56140862 0.60752251 0.65501133 0.70685934 0.7520323 0.78754258
 0.81590256 0.83956762 0.85992879 0.8777192 0.89335093 0.90709213
 0.91914843 0.92969762 0.93890316 0.94691812 0.9538854 0.9599366
 0.96519114 0.96975585 0.9737251 0.97718143 0.98019644 0.98160192]
[0.15983232 0.17232684 0.19899126 0.22917217 0.26318367 0.30128602
 0.34364097 0.39025494 0.44091315 0.49511248 0.55200734 0.6103886
 0.66871962 0.72525013 0.77821223 0.82606859 0.86774332 0.90275251
 0.9311893 0.95358604 0.97071926 0.98341618 0.99238084 1.00031517
 0.99931661 0.9825667 0.95217152 0.90697428 0.85681931 0.8085359
 0.7628015 0.71866581 0.67557501 0.63346744 0.59255933 0.55316452
 0.51558806 0.48007345 0.44678244 0.41579421 0.38711538 0.36069472
 0.33643871 0.31422572 0.29391778 0.27536967 0.25843572 0.25035422]
spectral_width=np.float64(0.8404828439686385)
processing time series: return SP500
2020-01-01    0.000000
2020-01-02    0.010483
2020-01-03   -0.003865
2020-01-06    0.001336
2020-01-07    0.001492
...
2025-02-07   -0.007095
2025-02-10    0.009498
2025-02-11   -0.002129
2025-02-12   -0.005867
2025-02-13    0.004606
Freq: B, Name: return SP500, Length: 1337, dtype: float64
[0.19968338 0.20149188 0.20543648 0.21010001 0.21562097 0.22216022

```

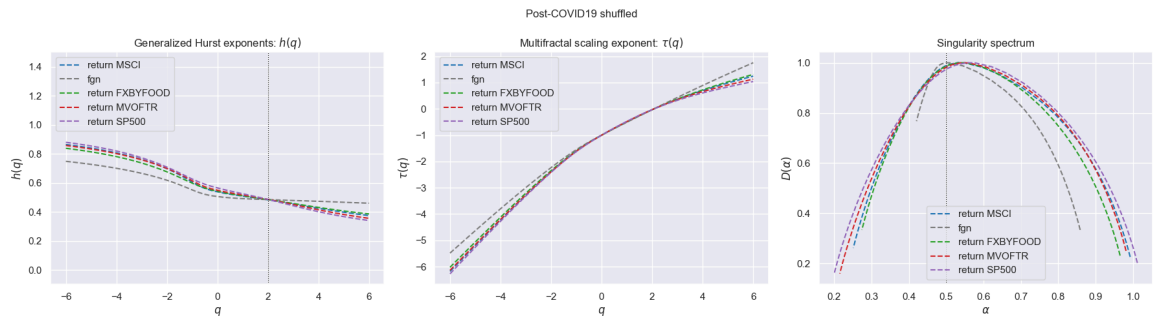
```

0.22990045 0.2390422 0.24979447 0.26235744 0.27689516 0.29349778
0.31213699 0.33262493 0.35459327 0.37751063 0.40074722 0.42367673
0.44578802 0.46677674 0.4866032 0.50552523 0.5241296 0.55430607
0.57787753 0.62190334 0.66126414 0.70636703 0.75201082 0.79318921
0.82806113 0.85719693 0.88172633 0.90256812 0.92037692 0.9356331
0.94871192 0.9599225 0.96952824 0.97775749 0.98480915 0.99085618
0.99604811 1.00051338 1.00436166 1.00768611 1.01056553 1.01190301]
[0.15902524 0.16942411 0.19157154 0.21658911 0.24482573 0.27663555
0.31235303 0.35225411 0.39650001 0.4450624 0.49763391 0.55353736
0.61166068 0.67045661 0.72804537 0.78243362 0.83181194 0.87484259
0.91083791 0.93977325 0.96214745 0.97874784 0.99037166 1.00024524
0.99956702 0.9841252 0.95903997 0.91934191 0.86815807 0.81193054
0.75565923 0.70135106 0.64948042 0.60018779 0.5536122 0.50989726
0.46915154 0.43142376 0.3966956 0.36488666 0.33586636 0.30946838
0.28550484 0.26377835 0.24409131 0.2262525 0.21008129 0.20239077]
spectral_width=np.float64(0.8412199976723533)

```

C:\Users\gabid\AppData\Local\Temp\ipykernel_23480\2494155285.py:4: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be removed in 3.11. Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get\_cmap()`` or ``pyplot.get\_cmap()`` instead.

```
cmap = plt.cm.get_cmap('tab10')
```



```
In [38]: all_results
```

```

Out[38]: {'return MSCI': {'q': array([-6.   , -5.75, -5.5  , -5.25, -5.   , -4.75, -4.
5   , -4.25, -4.   ,
      -3.75, -3.5  , -3.25, -3.   , -2.75, -2.5  , -2.25, -2.   , -1.75,
      -1.5  , -1.25, -1.   , -0.75, -0.5  , -0.25,  0.25,  0.5  ,  0.75,
       1.   ,  1.25,  1.5  ,  1.75,  2.   ,  2.25,  2.5  ,  2.75,  3.   ,
       3.25,  3.5  ,  3.75,  4.   ,  4.25,  4.5  ,  4.75,  5.   ,  5.25,
       5.5  ,  5.75,  6.   ]),
  'lag': array([ 3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 1
4, 15,
      16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28,
      30, 31, 32, 34, 35, 37, 39, 40, 42, 44, 46, 48, 50,
      52, 55, 57, 60, 62, 65, 68, 71, 74, 78, 81, 85, 89,
      93, 97, 101, 106, 110, 115, 120, 126, 131, 137, 143, 150, 156,
      163, 171, 178, 186, 194, 203, 212, 222]),
  'dfa': array([[4.61570845e-06, 4.83883634e-06, 5.09372416e-06, ...,
      6.73417420e-03, 7.02854839e-03, 7.31637028e-03],
      [2.20569319e-04, 2.29778650e-04, 2.40176916e-04, ...,
      9.21511763e-03, 9.55737797e-03, 9.89229586e-03],
      [8.77657518e-04, 9.09212871e-04, 9.44310617e-04, ...,
      1.21983442e-02, 1.26213917e-02, 1.30357871e-02],
      ...,
      [6.55848679e-01, 6.63105601e-01, 6.70906520e-01, ...,
      1.99663584e+00, 2.01928357e+00, 2.04089636e+00],
      [7.14491568e-01, 7.22246168e-01, 7.30579294e-01, ...,
      2.11220946e+00, 2.13519621e+00, 2.15713403e+00],
      [8.57884983e-01, 8.64759178e-01, 8.72129641e-01, ...,
      2.14976812e+00, 2.16848740e+00, 2.18623735e+00]]], shape=(73, 4
8)),
  'slope': array([1.86371629, 1.85810946, 1.8521191 , 1.8457094 , 1.838840
59,
      1.8314685 , 1.823544 , 1.81501227, 1.80581192, 1.79587382,
      1.78511942, 1.7734585 , 1.76078591, 1.74697731, 1.73188358,
      1.71532438, 1.69708235, 1.67690703, 1.65456828, 1.63010064,
      1.60451413, 1.58059505, 1.561391 , 1.54729367, 1.52833527,
      1.52107143, 1.51443643, 1.5081032 , 1.50186205, 1.49557369,
      1.48914641, 1.48252485, 1.47568453, 1.46862831, 1.46138312,
      1.45399532, 1.44652481, 1.43903823, 1.43160248, 1.42427929,
      1.41712167, 1.41017207, 1.40346207, 1.39701315, 1.39083795,
      1.38494187, 1.37932459, 1.3739815 ]),
  'hq': array([0.86371629, 0.85810946, 0.8521191 , 0.8457094 , 0.83884059,
      0.8314685 , 0.823544 , 0.81501227, 0.80581192, 0.79587382,
      0.78511942, 0.7734585 , 0.76078591, 0.74697731, 0.73188358,
      0.71532438, 0.69708235, 0.67690703, 0.65456828, 0.63010064,
      0.60451413, 0.58059505, 0.561391 , 0.54729367, 0.52833527,
      0.52107143, 0.51443643, 0.5081032 , 0.50186205, 0.49557369,
      0.48914641, 0.48252485, 0.47568453, 0.46862831, 0.46138312,
      0.45399532, 0.44652481, 0.43903823, 0.43160248, 0.42427929,
      0.41712167, 0.41017207, 0.40346207, 0.39701315, 0.39083795,
      0.38494187, 0.37932459, 0.3739815 ]),
  'tau': array([-6.18229776, -5.93412938, -5.68665506, -5.43997434, -5.194
20294,
      -4.94947539, -4.70594801, -4.46380215, -4.22324768, -3.98452682,
      -3.74791798, -3.51374013, -3.28235774, -3.0541876 , -2.82970896,
      -2.60947985, -2.3941647 , -2.18458731, -1.98185242, -1.7876258 ,
      -1.60451413, -1.43544629, -1.2806955 , -1.13682342, -0.86791618,
      -0.73946428, -0.61417268, -0.4918968 , -0.37267244, -0.25663946,
      -0.14399379, -0.03495029,  0.07029018,  0.17157079,  0.26880359,
      0.36198597,  0.45120562,  0.53663382,  0.6185093 ,  0.69711716,
      0.77276709,  0.8457743 ,  0.91644485,  0.98506575,  1.05189924,
      1.11718029,  1.18111641,  1.24388899]),
  'alpha': array([0.99267352, 0.9912854 , 0.98831008, 0.98490424, 0.980997

```

89,

```
0.97650984, 0.9713465 , 0.96540067, 0.95855066, 0.95065939,  
0.94157337, 0.93112048, 0.91910505, 0.90529756, 0.8894155 ,  
0.87108852, 0.84978509, 0.82462456, 0.793923 , 0.75467659,  
0.70435903, 0.64763726, 0.59724574, 0.55037242, 0.52981218,  
0.507487 , 0.49513497, 0.48300049, 0.47051468, 0.4573573 ,  
0.44337833, 0.42856795, 0.41304216, 0.39702681, 0.38083037,  
0.36480407, 0.34929569, 0.33460734, 0.32096669, 0.3085156 ,  
0.29731428, 0.28735551, 0.27858291, 0.27090879, 0.26422907,  
0.25843434, 0.2534174 , 0.25109029]]),
```

```
'Dalphi': array([0.22625662, 0.23423831, 0.25094959, 0.26922705, 0.28921  
349,
```

```
0.31105364, 0.33488879, 0.36084931, 0.38904505, 0.41955409,  
0.45241118, 0.48759856, 0.52504258, 0.56461931, 0.60617021,  
0.64953068, 0.69459451, 0.74149432, 0.79096791, 0.84428007,  
0.9001551 , 0.94971835, 0.98207263, 0.99923031, 1.00036923,  
0.99320779, 0.98552391, 0.97489729, 0.96081579, 0.9426754 ,  
0.91990587, 0.89208619, 0.85905467, 0.82099624, 0.77847992,  
0.73242624, 0.68400538, 0.63449189, 0.58511579, 0.53694523,  
0.49081859, 0.44732548, 0.40682395, 0.36947821, 0.33530338,  
0.3042086 , 0.27603362, 0.26265274]]),
```

	'metrics':	slope	hq	tau	alpha	Dalphi
-6.00	1.863716	0.863716	-6.182298	0.992674	0.226257	
-5.75	1.858109	0.858109	-5.934129	0.991285	0.234238	
-5.50	1.852119	0.852119	-5.686655	0.988310	0.250950	
-5.25	1.845709	0.845709	-5.439974	0.984904	0.269227	
-5.00	1.838841	0.838841	-5.194203	0.980998	0.289213	
-4.75	1.831469	0.831469	-4.949475	0.976510	0.311054	
-4.50	1.823544	0.823544	-4.705948	0.971346	0.334889	
-4.25	1.815012	0.815012	-4.463802	0.965401	0.360849	
-4.00	1.805812	0.805812	-4.223248	0.958551	0.389045	
-3.75	1.795874	0.795874	-3.984527	0.950659	0.419554	
-3.50	1.785119	0.785119	-3.747918	0.941573	0.452411	
-3.25	1.773459	0.773459	-3.513740	0.931120	0.487599	
-3.00	1.760786	0.760786	-3.282358	0.919105	0.525043	
-2.75	1.746977	0.746977	-3.054188	0.905298	0.564619	
-2.50	1.731884	0.731884	-2.829709	0.889415	0.606170	
-2.25	1.715324	0.715324	-2.609480	0.871089	0.649531	
-2.00	1.697082	0.697082	-2.394165	0.849785	0.694595	
-1.75	1.676907	0.676907	-2.184587	0.824625	0.741494	
-1.50	1.654568	0.654568	-1.981852	0.793923	0.790968	
-1.25	1.630101	0.630101	-1.787626	0.754677	0.844280	
-1.00	1.604514	0.604514	-1.604514	0.704359	0.900155	
-0.75	1.580595	0.580595	-1.435446	0.647637	0.949718	
-0.50	1.561391	0.561391	-1.280695	0.597246	0.982073	
-0.25	1.547294	0.547294	-1.136823	0.550372	0.999230	
0.25	1.528335	0.528335	-0.867916	0.529812	1.000369	
0.50	1.521071	0.521071	-0.739464	0.507487	0.993208	
0.75	1.514436	0.514436	-0.614173	0.495135	0.985524	
1.00	1.508103	0.508103	-0.491897	0.483000	0.974897	
1.25	1.501862	0.501862	-0.372672	0.470515	0.960816	
1.50	1.495574	0.495574	-0.256639	0.457357	0.942675	
1.75	1.489146	0.489146	-0.143994	0.443378	0.919906	
2.00	1.482525	0.482525	-0.034950	0.428568	0.892086	
2.25	1.475685	0.475685	0.070290	0.413042	0.859055	
2.50	1.468628	0.468628	0.171571	0.397027	0.820996	
2.75	1.461383	0.461383	0.268804	0.380830	0.778480	
3.00	1.453995	0.453995	0.361986	0.364804	0.732426	
3.25	1.446525	0.446525	0.451206	0.349296	0.684005	
3.50	1.439038	0.439038	0.536634	0.334607	0.634492	
3.75	1.431602	0.431602	0.618509	0.320967	0.585116	

```

4.00  1.424279  0.424279  0.697117  0.308516  0.536945
4.25  1.417122  0.417122  0.772767  0.297314  0.490819
4.50  1.410172  0.410172  0.845774  0.287356  0.447325
4.75  1.403462  0.403462  0.916445  0.278583  0.406824
5.00  1.397013  0.397013  0.985066  0.270909  0.369478
5.25  1.390838  0.390838  1.051899  0.264229  0.335303
5.50  1.384942  0.384942  1.117180  0.258434  0.304209
5.75  1.379325  0.379325  1.181116  0.253417  0.276034
6.00  1.373981  0.373981  1.243889  0.251090  0.262653,
'spectral_width': np.float64(0.7741126100567812),
'max_alpha': np.float64(0.5359383115040464),
'MDM': np.float64(0.2448673978875806)},
'fqn': {'q': array([-6. , -5.75, -5.5 , -5.25, -5. , -4.75, -4.5 , -4.2
5, -4. ,
-3.75, -3.5 , -3.25, -3. , -2.75, -2.5 , -2.25, -2. , -1.75,
-1.5 , -1.25, -1. , -0.75, -0.5 , -0.25, 0.25, 0.5 , 0.75,
1. , 1.25, 1.5 , 1.75, 2. , 2.25, 2.5 , 2.75, 3. ,
3.25, 3.5 , 3.75, 4. , 4.25, 4.5 , 4.75, 5. , 5.25,
5.5 , 5.75, 6. ]),
'lag': array([ 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 1
5, 16,
17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 30,
31, 32, 34, 35, 37, 39, 40, 42, 44, 46, 48, 50, 52,
55, 57, 60, 62, 65, 68, 71, 74, 78, 81, 85, 89, 93,
97, 101, 106, 110, 115, 120, 126, 131, 137, 143, 150, 156, 163,
171, 178, 186, 194, 203, 212, 222])),
'dfa': array([[2.28999435e-05, 2.39070827e-05, 2.50511458e-05, ...,
6.50638168e-03, 6.63254714e-03, 6.75577149e-03],
[6.39637496e-04, 6.64494402e-04, 6.92402538e-04, ...,
8.81522045e-03, 8.95628394e-03, 9.09485360e-03],
[2.04784352e-03, 2.11587928e-03, 2.19099094e-03, ...,
1.12819797e-02, 1.14522653e-02, 1.16200023e-02],
...,
[8.48560463e-01, 8.56766022e-01, 8.65511452e-01, ...,
1.91050926e+00, 1.92929963e+00, 1.94733231e+00],
[9.11985796e-01, 9.20648155e-01, 9.29873188e-01, ...,
2.04592975e+00, 2.06644091e+00, 2.08612522e+00],
[1.04973870e+00, 1.05730213e+00, 1.06535179e+00, ...,
2.10652669e+00, 2.12328456e+00, 2.13926060e+00]]], shape=(72, 4
8)),
'slope': array([1.7478063 , 1.74297874, 1.73786996, 1.73245853, 1.726720
84,
1.7206307 , 1.71415882, 1.70727204, 1.69993234, 1.69209556,
1.68370974, 1.67471301, 1.66503116, 1.65457478, 1.64323632,
1.63088801, 1.61738327, 1.60257125, 1.58635386, 1.56885574,
1.55079356, 1.53388515, 1.52038174, 1.51103269, 1.49985385,
1.4961988 , 1.49326894, 1.49084436, 1.48877716, 1.48696224,
1.48532209, 1.48379826, 1.48234619, 1.48093194, 1.47953003,
1.47812177, 1.47669406, 1.47523835, 1.47374981, 1.4722266 ,
1.47066919, 1.46907981, 1.46746201, 1.46582018, 1.46415929,
1.46248454, 1.4608012 , 1.4591144 ]),
'hq': array([0.7478063 , 0.74297874, 0.73786996, 0.73245853, 0.72672084,
0.7206307 , 0.71415882, 0.70727204, 0.69993234, 0.69209556,
0.68370974, 0.67471301, 0.66503116, 0.65457478, 0.64323632,
0.63088801, 0.61738327, 0.60257125, 0.58635386, 0.56885574,
0.55079356, 0.53388515, 0.52038174, 0.51103269, 0.49985385,
0.4961988 , 0.49326894, 0.49084436, 0.48877716, 0.48696224,
0.48532209, 0.48379826, 0.48234619, 0.48093194, 0.47953003,
0.47812177, 0.47669406, 0.47523835, 0.47374981, 0.4722266 ,
0.47066919, 0.46907981, 0.46746201, 0.46582018, 0.46415929,
0.46248454, 0.4608012 , 0.4591144 ]),

```



```

'tau': array([-5.48683779, -5.27212774, -5.05828481, -4.8454073 , -4.633
60418,
-4.42299581, -4.21371468, -4.00590616, -3.79972936, -3.59535836,
-3.39298407, -3.19281728, -2.99509349, -2.80008063, -2.60809079,
-2.41949802, -2.23476654, -2.05449968, -1.87953079, -1.71106967,
-1.55079356, -1.40041386, -1.26019087, -1.12775817, -0.87503654,
-0.7519006 , -0.63004829, -0.50915564, -0.38902855, -0.26955665,
-0.15068634, -0.03240348, 0.08527892, 0.20232986, 0.31870759,
0.43436532, 0.54925569, 0.66333422, 0.7765618 , 0.88890641,
1.00034405, 1.11085916, 1.22044454, 1.32910092, 1.43683627,
1.54366497, 1.64960688, 1.75468641]),
'alpha': array([0.8588402 , 0.85710596, 0.85344088, 0.84936126, 0.844822
98,
0.83977899, 0.8341793 , 0.82797065, 0.8210956 , 0.81349056,
0.80508215, 0.79578117, 0.7854733 , 0.7740054 , 0.76116522,
0.74664849, 0.72999668, 0.71047151, 0.68686002, 0.65747446,
0.62131163, 0.58120538, 0.54531138, 0.51353911, 0.50114343,
0.48997649, 0.48548992, 0.48203948, 0.47919798, 0.47668443,
0.47430634, 0.47193052, 0.46946667, 0.46685735, 0.46407093,
0.4610962 , 0.45793779, 0.45461222, 0.45114439, 0.4475645 ,
0.4439055 , 0.44020098, 0.43648352, 0.43278345, 0.42912809,
0.42554123, 0.42204288, 0.42031811]),
'Dalpha': array([0.33379661, 0.34376844, 0.36435994, 0.38626068, 0.40948
929,
0.43404561, 0.45990783, 0.4870309 , 0.51534695, 0.54476875,
0.57519655, 0.60652849, 0.6386736 , 0.67156578, 0.70517775,
0.73953892, 0.77477319, 0.81117454, 0.84924076, 0.88922659,
0.92948193, 0.96450983, 0.98753518, 0.9993734 , 1.0003224 ,
0.99688884, 0.99416574, 0.99119512, 0.98802603, 0.98458328,
0.98072243, 0.97626451, 0.97102109, 0.96481351, 0.95748747,
0.94892327, 0.93904212, 0.92780856, 0.91522965, 0.90135158,
0.88625433, 0.87004527, 0.85285216, 0.83481633, 0.81608622,
0.79681177, 0.77713971, 0.76722228]),
'metrics':      slope      hq      tau      alpha      Dalpha
-6.00  1.747806  0.747806 -5.486838  0.858840  0.333797
-5.75  1.742979  0.742979 -5.272128  0.857106  0.343768
-5.50  1.737870  0.737870 -5.058285  0.853441  0.364360
-5.25  1.732459  0.732459 -4.845407  0.849361  0.386261
-5.00  1.726721  0.726721 -4.633604  0.844823  0.409489
-4.75  1.720631  0.720631 -4.422996  0.839779  0.434046
-4.50  1.714159  0.714159 -4.213715  0.834179  0.459908
-4.25  1.707272  0.707272 -4.005906  0.827971  0.487031
-4.00  1.699932  0.699932 -3.799729  0.821096  0.515347
-3.75  1.692096  0.692096 -3.595358  0.813491  0.544769
-3.50  1.683710  0.683710 -3.392984  0.805082  0.575197
-3.25  1.674713  0.674713 -3.192817  0.795781  0.606528
-3.00  1.665031  0.665031 -2.995093  0.785473  0.638674
-2.75  1.654575  0.654575 -2.800081  0.774005  0.671566
-2.50  1.643236  0.643236 -2.608091  0.761165  0.705178
-2.25  1.630888  0.630888 -2.419498  0.746648  0.739539
-2.00  1.617383  0.617383 -2.234767  0.729997  0.774773
-1.75  1.602571  0.602571 -2.054500  0.710472  0.811175
-1.50  1.586354  0.586354 -1.879531  0.686860  0.849241
-1.25  1.568856  0.568856 -1.711070  0.657474  0.889227
-1.00  1.550794  0.550794 -1.550794  0.621312  0.929482
-0.75  1.533885  0.533885 -1.400414  0.581205  0.964510
-0.50  1.520382  0.520382 -1.260191  0.545311  0.987535
-0.25  1.511033  0.511033 -1.127758  0.513539  0.999373
0.25  1.499854  0.499854 -0.875037  0.501143  1.000322
0.50  1.496199  0.496199 -0.751901  0.489976  0.996889
0.75  1.493269  0.493269 -0.630048  0.485490  0.994166

```

```

1.00  1.490844  0.490844 -0.509156  0.482039  0.991195
1.25  1.488777  0.488777 -0.389029  0.479198  0.988026
1.50  1.486962  0.486962 -0.269557  0.476684  0.984583
1.75  1.485322  0.485322 -0.150686  0.474306  0.980722
2.00  1.483798  0.483798 -0.032403  0.471931  0.976265
2.25  1.482346  0.482346  0.085279  0.469467  0.971021
2.50  1.480932  0.480932  0.202330  0.466857  0.964814
2.75  1.479530  0.479530  0.318708  0.464071  0.957487
3.00  1.478122  0.478122  0.434365  0.461096  0.948923
3.25  1.476694  0.476694  0.549256  0.457938  0.939042
3.50  1.475238  0.475238  0.663334  0.454612  0.927809
3.75  1.473750  0.473750  0.776562  0.451144  0.915230
4.00  1.472227  0.472227  0.888906  0.447564  0.901352
4.25  1.470669  0.470669  1.000344  0.443906  0.886254
4.50  1.469080  0.469080  1.110859  0.440201  0.870045
4.75  1.467462  0.467462  1.220445  0.436484  0.852852
5.00  1.465820  0.465820  1.329101  0.432783  0.834816
5.25  1.464159  0.464159  1.436836  0.429128  0.816086
5.50  1.462485  0.462485  1.543665  0.425541  0.796812
5.75  1.460801  0.460801  1.649607  0.422043  0.777140
6.00  1.459114  0.459114  1.754686  0.420318  0.767222,
'spectral_width': np.float64(0.6665257887354685),
'max_alpha': np.float64(0.5041556606584148),
'MDM': np.float64(0.14434594802065848)},
'return FXBYFOOD': {'q': array([-6. , -5.75, -5.5 , -5.25, -5. , -4.75,
-4.5 , -4.25, -4. ,
-3.75, -3.5 , -3.25, -3. , -2.75, -2.5 , -2.25, -2. , -1.75,
-1.5 , -1.25, -1. , -0.75, -0.5 , -0.25,  0.25,  0.5 ,  0.75,
 1. ,  1.25,  1.5 ,  1.75,  2. ,  2.25,  2.5 ,  2.75,  3. ,
 3.25,  3.5 ,  3.75,  4. ,  4.25,  4.5 ,  4.75,  5. ,  5.25,
 5.5 ,  5.75,  6. ]),
'lag': array([ 3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 1
4, 15,
16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28,
30, 31, 32, 34, 35, 37, 39, 40, 42, 44, 46, 48, 50,
52, 55, 57, 60, 62, 65, 68, 71, 74, 78, 81, 85, 89,
93, 97, 101, 106, 110, 115, 120, 126, 131, 137, 143, 150, 156,
163, 171, 178, 186, 194, 203, 212, 222]),
'dfa': array([[1.38862025e-05, 1.44954751e-05, 1.51866319e-05, ...,
6.44051569e-03, 6.70415786e-03, 6.96173644e-03],
[2.11178932e-04, 2.20111135e-04, 2.30205202e-04, ...,
8.80824105e-03, 9.11123814e-03, 9.40759115e-03],
[8.57391121e-04, 8.88252841e-04, 9.22538888e-04, ...,
1.16835096e-02, 1.20592694e-02, 1.24272887e-02],
...,
[6.65668912e-01, 6.72990504e-01, 6.80859757e-01, ...,
1.99385535e+00, 2.01662661e+00, 2.03835766e+00],
[7.18386415e-01, 7.26050614e-01, 7.34277228e-01, ...,
2.10702686e+00, 2.13077032e+00, 2.15343876e+00],
[8.54015484e-01, 8.60891433e-01, 8.68258091e-01, ...,
2.13967953e+00, 2.15865769e+00, 2.17665750e+00]]], shape=(73, 4
8)),
'slope': array([1.83760381, 1.83203582, 1.82609088, 1.81973441, 1.812928
26,
1.80563025, 1.79779383, 1.78936756, 1.78029452, 1.77051162,
1.75994871, 1.74852751, 1.7361606 , 1.72275115, 1.70819526,
1.69239162, 1.67526705, 1.65683204, 1.63728006, 1.61712839,
1.59733523, 1.5791685 , 1.56362802, 1.55089982, 1.53171242,
1.52399302, 1.51692154, 1.51022156, 1.50370735, 1.49725376,
1.4907784 , 1.48423158, 1.47759047, 1.47085489, 1.46404333,
1.45718843, 1.450332 , 1.44352004, 1.43679812, 1.43020794,

```

```

1.42378502, 1.41755756, 1.41154632, 1.40576513, 1.40022174,
1.39491888, 1.38985525, 1.38502652]),
'hq': array([0.83760381, 0.83203582, 0.82609088, 0.81973441, 0.81292826,
0.80563025, 0.79779383, 0.78936756, 0.78029452, 0.77051162,
0.75994871, 0.74852751, 0.7361606 , 0.72275115, 0.70819526,
0.69239162, 0.67526705, 0.65683204, 0.63728006, 0.61712839,
0.59733523, 0.5791685 , 0.56362802, 0.55089982, 0.53171242,
0.52399302, 0.51692154, 0.51022156, 0.50370735, 0.49725376,
0.4907784 , 0.48423158, 0.47759047, 0.47085489, 0.46404333,
0.45718843, 0.450332 , 0.44352004, 0.43679812, 0.43020794,
0.42378502, 0.41755756, 0.41154632, 0.40576513, 0.40022174,
0.39491888, 0.38985525, 0.38502652]),
'tau': array([-6.02562285, -5.78420595, -5.54349982, -5.30360568, -5.064
64129,
-4.82674368, -4.59007224, -4.35481212, -4.12117807, -3.88941858,
-3.65982048, -3.4327144 , -3.20848181, -2.98756567, -2.77048816,
-2.55788113, -2.3505341 , -2.14945608, -1.9559201 , -1.77141049,
-1.59733523, -1.43437638, -1.28181401, -1.13772496, -0.8670719 ,
-0.73800349, -0.61230885, -0.48977844, -0.37036582, -0.25411936,
-0.1411378 , -0.03153684, 0.07457855, 0.17713723, 0.27611916,
0.37156528, 0.46357901, 0.55232012, 0.63799293, 0.72083176,
0.80108632, 0.879009 , 0.95484503, 1.02882566, 1.10116416,
1.17205382, 1.24166767, 1.31015912]),
'alpha': array([0.96566761, 0.96424607, 0.96120054, 0.95771705, 0.953723
99,
0.94913811, 0.94386313, 0.93778833, 0.93078708, 0.92271518,
0.91340836, 0.90267734, 0.89029747, 0.8759873 , 0.85936907,
0.83990812, 0.81685011, 0.78922801, 0.75609118, 0.71716972,
0.67406822, 0.63104244, 0.59330284, 0.55298949, 0.53296196,
0.5095261 , 0.4964501 , 0.48388606, 0.47131817, 0.45845604,
0.44516503, 0.4314327 , 0.41734814, 0.40308122, 0.3888561 ,
0.3749197 , 0.36150969, 0.34882785, 0.33702327, 0.32618677,
0.31635448, 0.30751741, 0.29963332, 0.29263826, 0.28645631,
0.28100703, 0.27621061, 0.27396581]),
'Dalpha': array([0.23161719, 0.23979106, 0.25689683, 0.27559115, 0.29602
135,
0.31833768, 0.34268814, 0.3692117 , 0.39802975, 0.42923665,
0.46289123, 0.49901306, 0.53758941, 0.57860059, 0.6220655 ,
0.66808786, 0.71683388, 0.76830706, 0.82178332, 0.87494833,
0.92326701, 0.96109454, 0.98516259, 0.99947758, 1.00031238,
0.99276654, 0.98464642, 0.9736645 , 0.95951352, 0.94180341,
0.9201766 , 0.89440223, 0.86445476, 0.83056582, 0.79323511,
0.75319384, 0.7113275 , 0.66857733, 0.62584434, 0.58391533,
0.54342023, 0.50481936, 0.46841326, 0.43436564, 0.40273147,
0.37348483, 0.34654334, 0.33363576]),
'metrics':      slope      hq      tau      alpha      Dalpha
-6.00  1.837604  0.837604 -6.025623  0.965668  0.231617
-5.75  1.832036  0.832036 -5.784206  0.964246  0.239791
-5.50  1.826091  0.826091 -5.543500  0.961201  0.256897
-5.25  1.819734  0.819734 -5.303606  0.957717  0.275591
-5.00  1.812928  0.812928 -5.064641  0.953724  0.296021
-4.75  1.805630  0.805630 -4.826744  0.949138  0.318338
-4.50  1.797794  0.797794 -4.590072  0.943863  0.342688
-4.25  1.789368  0.789368 -4.354812  0.937788  0.369212
-4.00  1.780295  0.780295 -4.121178  0.930787  0.398030
-3.75  1.770512  0.770512 -3.889419  0.922715  0.429237
-3.50  1.759949  0.759949 -3.659820  0.913408  0.462891
-3.25  1.748528  0.748528 -3.432714  0.902677  0.499013
-3.00  1.736161  0.736161 -3.208482  0.890297  0.537589
-2.75  1.722751  0.722751 -2.987566  0.875987  0.578601
-2.50  1.708195  0.708195 -2.770488  0.859369  0.622065

```

```

-2.25 1.692392 0.692392 -2.557881 0.839908 0.668088
-2.00 1.675267 0.675267 -2.350534 0.816850 0.716834
-1.75 1.656832 0.656832 -2.149456 0.789228 0.768307
-1.50 1.637280 0.637280 -1.955920 0.756091 0.821783
-1.25 1.617128 0.617128 -1.771410 0.717170 0.874948
-1.00 1.597335 0.597335 -1.597335 0.674068 0.923267
-0.75 1.579169 0.579169 -1.434376 0.631042 0.961095
-0.50 1.563628 0.563628 -1.281814 0.593303 0.985163
-0.25 1.550900 0.550900 -1.137725 0.552989 0.999478
0.25 1.531712 0.531712 -0.867072 0.532962 1.000312
0.50 1.523993 0.523993 -0.738003 0.509526 0.992767
0.75 1.516922 0.516922 -0.612309 0.496450 0.984646
1.00 1.510222 0.510222 -0.489778 0.483886 0.973665
1.25 1.503707 0.503707 -0.370366 0.471318 0.959514
1.50 1.497254 0.497254 -0.254119 0.458456 0.941803
1.75 1.490778 0.490778 -0.141138 0.445165 0.920177
2.00 1.484232 0.484232 -0.031537 0.431433 0.894402
2.25 1.477590 0.477590 0.074579 0.417348 0.864455
2.50 1.470855 0.470855 0.177137 0.403081 0.830566
2.75 1.464043 0.464043 0.276119 0.388856 0.793235
3.00 1.457188 0.457188 0.371565 0.374920 0.753194
3.25 1.450332 0.450332 0.463579 0.361510 0.711327
3.50 1.443520 0.443520 0.552320 0.348828 0.668577
3.75 1.436798 0.436798 0.637993 0.337023 0.625844
4.00 1.430208 0.430208 0.720832 0.326187 0.583915
4.25 1.423785 0.423785 0.801086 0.316354 0.543420
4.50 1.417558 0.417558 0.879009 0.307517 0.504819
4.75 1.411546 0.411546 0.954845 0.299633 0.468413
5.00 1.405765 0.405765 1.028826 0.292638 0.434366
5.25 1.400222 0.400222 1.101164 0.286456 0.402731
5.50 1.394919 0.394919 1.172054 0.281007 0.373485
5.75 1.389855 0.389855 1.241668 0.276211 0.346543
6.00 1.385027 0.385027 1.310159 0.273966 0.333636,
'spectral_width': np.float64(0.7686951894717411),
'max_alpha': np.float64(0.5397554267276626),
'MDM': np.float64(0.22628864393742287)},
'return MVOFTR': {'q': array([-6. , -5.75, -5.5 , -5.25, -5. , -4.75, -
4.5 , -4.25, -4. ,
-3.75, -3.5 , -3.25, -3. , -2.75, -2.5 , -2.25, -2. , -1.75,
-1.5 , -1.25, -1. , -0.75, -0.5 , -0.25, 0.25, 0.5 , 0.75,
1. , 1.25, 1.5 , 1.75, 2. , 2.25, 2.5 , 2.75, 3. ,
3.25, 3.5 , 3.75, 4. , 4.25, 4.5 , 4.75, 5. , 5.25,
5.5 , 5.75, 6. ]),
'lag': array([ 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 1
4, 15,
16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28,
30, 31, 32, 34, 35, 37, 39, 40, 42, 44, 46, 48, 50,
52, 55, 57, 60, 62, 65, 68, 71, 74, 78, 81, 85, 89,
93, 97, 101, 106, 110, 115, 120, 126, 131, 137, 143, 150, 156,
163, 171, 178, 186, 194, 203, 212, 222])),
'dfa': array([[7.23234921e-06, 7.53635018e-06, 7.88143776e-06, ...,
6.21559997e-03, 6.46332964e-03, 6.70336700e-03],
[2.24049150e-04, 2.33193806e-04, 2.43490619e-04, ...,
9.31172153e-03, 9.69237498e-03, 1.00639806e-02],
[8.95932528e-04, 9.23703508e-04, 9.54390790e-04, ...,
1.16212017e-02, 1.20160546e-02, 1.23994789e-02],
...,
[6.07564082e-01, 6.14274342e-01, 6.21486621e-01, ...,
1.84356184e+00, 1.86455089e+00, 1.88458095e+00],
[6.64595935e-01, 6.71611402e-01, 6.79144167e-01, ...,
1.93300117e+00, 1.95434732e+00, 1.97472217e+00],

```

```

[7.88727452e-01, 7.94936086e-01, 8.01592082e-01, ...,
 1.97108757e+00, 1.98838485e+00, 2.00478186e+00]], shape=(73, 4
8)),
'slope': array([1.85666096, 1.85122874, 1.84543046, 1.83923325, 1.832601
,
1.82549403, 1.81786876, 1.80967725, 1.80086668, 1.79137847,
1.78114718, 1.7700986 , 1.75814697, 1.74519069, 1.73110618,
1.71573991, 1.69890053, 1.68036056, 1.65989985, 1.63748775,
1.61383362, 1.59124002, 1.57265591, 1.55867505, 1.53891404,
1.53092454, 1.52342882, 1.5161208 , 1.50879332, 1.50130125,
1.49354593, 1.48546992, 1.47705607, 1.46832658, 1.45933887,
1.45017712, 1.44094063, 1.43173185, 1.42264641, 1.41376658,
1.40515799, 1.39686903, 1.38893204, 1.38136556, 1.37417683,
1.36736437, 1.36092027, 1.3548321 ]),
'hq': array([0.85666096, 0.85122874, 0.84543046, 0.83923325, 0.832601 ,
0.82549403, 0.81786876, 0.80967725, 0.80086668, 0.79137847,
0.78114718, 0.7700986 , 0.75814697, 0.74519069, 0.73110618,
0.71573991, 0.69890053, 0.68036056, 0.65989985, 0.63748775,
0.61383362, 0.59124002, 0.57265591, 0.55867505, 0.53891404,
0.53092454, 0.52342882, 0.5161208 , 0.50879332, 0.50130125,
0.49354593, 0.48546992, 0.47705607, 0.46832658, 0.45933887,
0.45017712, 0.44094063, 0.43173185, 0.42264641, 0.41376658,
0.40515799, 0.39686903, 0.38893204, 0.38136556, 0.37417683,
0.36736437, 0.36092027, 0.3548321 ]),
'tau': array([-6.13996575, -5.89456527, -5.64986753, -5.40597456, -5.163
00498,
-4.92109663, -4.68040941, -4.44112833, -4.20346671, -3.96766927,
-3.73401513, -3.50282046, -3.27444091, -3.04927439, -2.82776545,
-2.61041479, -2.39780105, -2.19063098, -1.98984977, -1.79685969,
-1.61383362, -1.44343002, -1.28632796, -1.13966876, -0.86527149,
-0.73453773, -0.60742838, -0.4838792 , -0.36400835, -0.24804813,
-0.13629463, -0.02906016, 0.07337616, 0.17081645, 0.2631819 ,
0.35053136, 0.43305706, 0.51106146, 0.58492403, 0.65506633,
0.72192147, 0.78591061, 0.8474272 , 0.9068278 , 0.96442834,
1.02050404, 1.07529155, 1.12899259]),
'alpha': array([0.98160192, 0.98019644, 0.97718143, 0.9737251 , 0.969755
85,
0.96519114, 0.9599366 , 0.9538854 , 0.94691812, 0.93890316,
0.92969762, 0.91914843, 0.90709213, 0.89335093, 0.8777192 ,
0.85992879, 0.83956762, 0.81590256, 0.78754258, 0.7520323 ,
0.70685934, 0.65501133, 0.60752251, 0.56140862, 0.54017471,
0.51568621, 0.50131706, 0.48684006, 0.47166215, 0.45542745,
0.43797593, 0.41934158, 0.39975322, 0.37961147, 0.35942983,
0.33975033, 0.3210602 , 0.30373394, 0.28800974, 0.27399487,
0.26168857, 0.25101146, 0.24183436, 0.23400229, 0.22735248,
0.22172642, 0.21697711, 0.21480415]),
'Dalpha': array([0.25035422, 0.25843572, 0.27536967, 0.29391778, 0.31422
572,
0.33643871, 0.36069472, 0.38711538, 0.41579421, 0.44678244,
0.48007345, 0.51558806, 0.55316452, 0.59255933, 0.63346744,
0.67557501, 0.71866581, 0.7628015 , 0.8085359 , 0.85681931,
0.90697428, 0.95217152, 0.9825667 , 0.99931661, 1.00031517,
0.99238084, 0.98341618, 0.97071926, 0.95358604, 0.9311893 ,
0.90275251, 0.86774332, 0.82606859, 0.77821223, 0.72525013,
0.66871962, 0.6103886 , 0.55200734, 0.49511248, 0.44091315,
0.39025494, 0.34364097, 0.30128602, 0.26318367, 0.22917217,
0.19899126, 0.17232684, 0.15983232]),
'metrics':      slope      hq      tau      alpha      Dalpha
-6.00  1.856661  0.856661 -6.139966  0.981602  0.250354
-5.75  1.851229  0.851229 -5.894565  0.980196  0.258436
-5.50  1.845430  0.845430 -5.649868  0.977181  0.275370

```

```

-5.25  1.839233  0.839233 -5.405975  0.973725  0.293918
-5.00  1.832601  0.832601 -5.163005  0.969756  0.314226
-4.75  1.825494  0.825494 -4.921097  0.965191  0.336439
-4.50  1.817869  0.817869 -4.680409  0.959937  0.360695
-4.25  1.809677  0.809677 -4.441128  0.953885  0.387115
-4.00  1.800867  0.800867 -4.203467  0.946918  0.415794
-3.75  1.791378  0.791378 -3.967669  0.938903  0.446782
-3.50  1.781147  0.781147 -3.734015  0.929698  0.480073
-3.25  1.770099  0.770099 -3.502820  0.919148  0.515588
-3.00  1.758147  0.758147 -3.274441  0.907092  0.553165
-2.75  1.745191  0.745191 -3.049274  0.893351  0.592559
-2.50  1.731106  0.731106 -2.827765  0.877719  0.633467
-2.25  1.715740  0.715740 -2.610415  0.859929  0.675575
-2.00  1.698901  0.698901 -2.397801  0.839568  0.718666
-1.75  1.680361  0.680361 -2.190631  0.815903  0.762801
-1.50  1.659900  0.659900 -1.989850  0.787543  0.808536
-1.25  1.637488  0.637488 -1.796860  0.752032  0.856819
-1.00  1.613834  0.613834 -1.613834  0.706859  0.906974
-0.75  1.591240  0.591240 -1.443430  0.655011  0.952172
-0.50  1.572656  0.572656 -1.286328  0.607523  0.982567
-0.25  1.558675  0.558675 -1.139669  0.561409  0.999317
 0.25  1.538914  0.538914 -0.865271  0.540175  1.000315
 0.50  1.530925  0.530925 -0.734538  0.515686  0.992381
 0.75  1.523429  0.523429 -0.607428  0.501317  0.983416
 1.00  1.516121  0.516121 -0.483879  0.486840  0.970719
 1.25  1.508793  0.508793 -0.364008  0.471662  0.953586
 1.50  1.501301  0.501301 -0.248048  0.455427  0.931189
 1.75  1.493546  0.493546 -0.136295  0.437976  0.902753
 2.00  1.485470  0.485470 -0.029060  0.419342  0.867743
 2.25  1.477056  0.477056  0.073376  0.399753  0.826069
 2.50  1.468327  0.468327  0.170816  0.379611  0.778212
 2.75  1.459339  0.459339  0.263182  0.359430  0.725250
 3.00  1.450177  0.450177  0.350531  0.339750  0.668720
 3.25  1.440941  0.440941  0.433057  0.321060  0.610389
 3.50  1.431732  0.431732  0.511061  0.303734  0.552007
 3.75  1.422646  0.422646  0.584924  0.288010  0.495112
 4.00  1.413767  0.413767  0.655066  0.273995  0.440913
 4.25  1.405158  0.405158  0.721921  0.261689  0.390255
 4.50  1.396869  0.396869  0.785911  0.251011  0.343641
 4.75  1.388932  0.388932  0.847427  0.241834  0.301286
 5.00  1.381366  0.381366  0.906828  0.234002  0.263184
 5.25  1.374177  0.374177  0.964428  0.227352  0.229172
 5.50  1.367364  0.367364  1.020504  0.221726  0.198991
 5.75  1.360920  0.360920  1.075292  0.216977  0.172327
 6.00  1.354832  0.354832  1.128993  0.214804  0.159832,
'spectral_width': np.float64(0.8404828439686385),
'max_alpha': np.float64(0.547040525554207),
'MDM': np.float64(0.2509144298254673)},
'return SP500': {'q': array([-6. , -5.75, -5.5 , -5.25, -5. , -4.75, -
4.5 , -4.25, -4. ,
        -3.75, -3.5 , -3.25, -3. , -2.75, -2.5 , -2.25, -2. , -1.75,
        -1.5 , -1.25, -1. , -0.75, -0.5 , -0.25,  0.25,  0.5 ,  0.75,
         1. ,  1.25,  1.5 ,  1.75,  2. ,  2.25,  2.5 ,  2.75,  3. ,
         3.25,  3.5 ,  3.75,  4. ,  4.25,  4.5 ,  4.75,  5. ,  5.25,
         5.5 ,  5.75,  6. ]),
'lag': array([ 3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 1
4, 15,
        16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28,
        30, 31, 32, 34, 35, 37, 39, 40, 42, 44, 46, 48, 50,
        52, 55, 57, 60, 62, 65, 68, 71, 74, 78, 81, 85, 89,
        93, 97, 101, 106, 110, 115, 120, 126, 131, 137, 143, 150, 156,

```

```

163, 171, 178, 186, 194, 203, 212, 222]),
'dfa': array([[1.16932250e-05, 1.22371710e-05, 1.28568830e-05, ...,
8.97601724e-03, 9.37902795e-03, 9.77088341e-03],
[2.42134555e-04, 2.51833107e-04, 2.62746331e-04, ...,
1.20646629e-02, 1.25342226e-02, 1.29905190e-02],
[7.61040914e-04, 7.89145602e-04, 8.20480650e-04, ...,
1.58028833e-02, 1.63881791e-02, 1.69574054e-02],
...,
[7.40746680e-01, 7.48938489e-01, 7.57749097e-01, ...,
2.29040925e+00, 2.31662085e+00, 2.34162207e+00],
[8.02556349e-01, 8.11246077e-01, 8.20582902e-01, ...,
2.42560464e+00, 2.45296958e+00, 2.47908061e+00],
[9.48222941e-01, 9.56068989e-01, 9.64487760e-01, ...,
2.48261837e+00, 2.50500070e+00, 2.52621496e+00]], shape=(73, 4
8)),
'slope': array([1.87896814, 1.87318836, 1.86700474, 1.86037906, 1.853269
05,
1.84562807, 1.83740471, 1.82854241, 1.81897916, 1.80864707,
1.79747214, 1.78537393, 1.77226551, 1.75805408, 1.74264323,
1.72593985, 1.70787246, 1.68843783, 1.66780957, 1.64653728,
1.62570894, 1.60665077, 1.59015374, 1.57614563, 1.55332512,
1.54338628, 1.53386144, 1.52445576, 1.51495814, 1.50522942,
1.49519525, 1.48484125, 1.4742068 , 1.46337512, 1.45245889,
1.44158342, 1.4308709 , 1.42042833, 1.4103408 , 1.40066947,
1.391453 , 1.38271089, 1.37444747, 1.36665582, 1.35932113,
1.35242347, 1.34593986, 1.33984584]),
'hq': array([0.87896814, 0.87318836, 0.86700474, 0.86037906, 0.85326905,
0.84562807, 0.83740471, 0.82854241, 0.81897916, 0.80864707,
0.79747214, 0.78537393, 0.77226551, 0.75805408, 0.74264323,
0.72593985, 0.70787246, 0.68843783, 0.66780957, 0.64653728,
0.62570894, 0.60665077, 0.59015374, 0.57614563, 0.55332512,
0.54338628, 0.53386144, 0.52445576, 0.51495814, 0.50522942,
0.49519525, 0.48484125, 0.4742068 , 0.46337512, 0.45245889,
0.44158342, 0.4308709 , 0.42042833, 0.4103408 , 0.40066947,
0.391453 , 0.38271089, 0.37444747, 0.36665582, 0.35932113,
0.35242347, 0.34593986, 0.33984584]),
'tau': array([-6.27380885, -6.0208331 , -5.76852609, -5.51699004, -5.266
34525,
-5.01673335, -4.7683212 , -4.52130526, -4.27591662, -4.03242652,
-3.7911525 , -3.55246527, -3.31679654, -3.08464872, -2.85660808,
-2.63336466, -2.41574492, -2.2047662 , -2.00171435, -1.8081716 ,
-1.62570894, -1.45498808, -1.29507687, -1.14403641, -0.86166872,
-0.72830686, -0.59960392, -0.47554424, -0.35630232, -0.24215587,
-0.13340831, -0.03031751, 0.0669653 , 0.15843781, 0.24426194,
0.32475027, 0.40033043, 0.47149916, 0.53877801, 0.60267788,
0.66367525, 0.72219899, 0.77862547, 0.83327909, 0.88643595,
0.9383291 , 0.98915419, 1.03907504]),
'alpha': array([1.01190301, 1.01056553, 1.00768611, 1.00436166, 1.000513
38,
0.99604811, 0.99085618, 0.98480915, 0.97775749, 0.96952824,
0.9599225 , 0.94871192, 0.9356331 , 0.92037692, 0.90256812,
0.88172633, 0.85719693, 0.82806113, 0.79318921, 0.75201082,
0.70636703, 0.66126414, 0.62190334, 0.57787753, 0.55430607,
0.5241296 , 0.50552523, 0.4866032 , 0.46677674, 0.44578802,
0.42367673, 0.40074722, 0.37751063, 0.35459327, 0.33262493,
0.31213699, 0.29349778, 0.27689516, 0.26235744, 0.24979447,
0.2390422 , 0.22990045, 0.22216022, 0.21562097, 0.21010001,
0.20543648, 0.20149188, 0.19968338]),
'Dalpha': array([0.20239077, 0.21008129, 0.2262525 , 0.24409131, 0.26377
835,
0.28550484, 0.30946838, 0.33586636, 0.36488666, 0.3966956 ,

```

```

0.43142376, 0.46915154, 0.50989726, 0.5536122 , 0.60018779,
0.64948042, 0.70135106, 0.75565923, 0.81193054, 0.86815807,
0.91934191, 0.95903997, 0.9841252 , 0.99956702, 1.00024524,
0.99037166, 0.97874784, 0.96214745, 0.93977325, 0.91083791,
0.87484259, 0.83181194, 0.78243362, 0.72804537, 0.67045661,
0.61166068, 0.55353736, 0.49763391, 0.4450624 , 0.39650001,
0.35225411, 0.31235303, 0.27663555, 0.24482573, 0.21658911,
0.19157154, 0.16942411, 0.15902524]),
'metrics':      slope      hq      tau      alpha      Dalpha
-6.00  1.878968  0.878968 -6.273809  1.011903  0.202391
-5.75  1.873188  0.873188 -6.020833  1.010566  0.210081
-5.50  1.867005  0.867005 -5.768526  1.007686  0.226252
-5.25  1.860379  0.860379 -5.516990  1.004362  0.244091
-5.00  1.853269  0.853269 -5.266345  1.000513  0.263778
-4.75  1.845628  0.845628 -5.016733  0.996048  0.285505
-4.50  1.837405  0.837405 -4.768321  0.990856  0.309468
-4.25  1.828542  0.828542 -4.521305  0.984809  0.335866
-4.00  1.818979  0.818979 -4.275917  0.977757  0.364887
-3.75  1.808647  0.808647 -4.032427  0.969528  0.396696
-3.50  1.797472  0.797472 -3.791153  0.959922  0.431424
-3.25  1.785374  0.785374 -3.552465  0.948712  0.469152
-3.00  1.772266  0.772266 -3.316797  0.935633  0.509897
-2.75  1.758054  0.758054 -3.084649  0.920377  0.553612
-2.50  1.742643  0.742643 -2.856608  0.902568  0.600188
-2.25  1.725940  0.725940 -2.633365  0.881726  0.649480
-2.00  1.707872  0.707872 -2.415745  0.857197  0.701351
-1.75  1.688438  0.688438 -2.204766  0.828061  0.755659
-1.50  1.667810  0.667810 -2.001714  0.793189  0.811931
-1.25  1.646537  0.646537 -1.808172  0.752011  0.868158
-1.00  1.625709  0.625709 -1.625709  0.706367  0.919342
-0.75  1.606651  0.606651 -1.454988  0.661264  0.959040
-0.50  1.590154  0.590154 -1.295077  0.621903  0.984125
-0.25  1.576146  0.576146 -1.144036  0.577878  0.999567
 0.25  1.553325  0.553325 -0.861669  0.554306  1.000245
 0.50  1.543386  0.543386 -0.728307  0.524130  0.990372
 0.75  1.533861  0.533861 -0.599604  0.505525  0.978748
 1.00  1.524456  0.524456 -0.475544  0.486603  0.962147
 1.25  1.514958  0.514958 -0.356302  0.466777  0.939773
 1.50  1.505229  0.505229 -0.242156  0.445788  0.910838
 1.75  1.495195  0.495195 -0.133408  0.423677  0.874843
 2.00  1.484841  0.484841 -0.030318  0.400747  0.831812
 2.25  1.474207  0.474207  0.066965  0.377511  0.782434
 2.50  1.463375  0.463375  0.158438  0.354593  0.728045
 2.75  1.452459  0.452459  0.244262  0.332625  0.670457
 3.00  1.441583  0.441583  0.324750  0.312137  0.611661
 3.25  1.430871  0.430871  0.400330  0.293498  0.553537
 3.50  1.420428  0.420428  0.471499  0.276895  0.497634
 3.75  1.410341  0.410341  0.538778  0.262357  0.445062
 4.00  1.400669  0.400669  0.602678  0.249794  0.396500
 4.25  1.391453  0.391453  0.663675  0.239042  0.352254
 4.50  1.382711  0.382711  0.722199  0.229900  0.312353
 4.75  1.374447  0.374447  0.778625  0.222160  0.276636
 5.00  1.366656  0.366656  0.833279  0.215621  0.244826
 5.25  1.359321  0.359321  0.886436  0.210100  0.216589
 5.50  1.352423  0.352423  0.938329  0.205436  0.191572
 5.75  1.345940  0.345940  0.989154  0.201492  0.169424
 6.00  1.339846  0.339846  1.039075  0.199683  0.159025,
'spectral_width': np.float64(0.8412199976723533),
'max_alpha': np.float64(0.5633819110490189),
'MDM': np.float64(0.269561151085305)}}

```



```
In [39]: prueba=pd.DataFrame(all_results)
prueba.to_excel("Results_2nd period shuffled final.xlsx")
```

9. Gráficas comparativas una a una MF-DFA

```
In [ ]: def limpiar_y_convertir_a_array(valor):
    if isinstance(valor, str):
        try:
            valor = valor.replace("\n", " ").replace(",,", ",")
            return np.array(eval(valor), dtype=np.float64)
        except:
            return valor
    return valor
```

```
In [ ]: def plot_multifractal_unico(results_dict, filename=None, title=None):
    plt.figure(figsize=(10, 6))
    series_names = list(results_dict.keys())
    cmap = plt.cm.get_cmap('tab10', len(series_names))

    for indice, series_name in enumerate(series_names):
        res = results_dict[series_name]
        color = cmap(indice)
        plt.plot(res['alpha'], res['Dalpha'], '--', color=color, label=series_name)

    plt.axvline(0.5, linestyle='--', linewidth=0.8, color='gray', alpha=0.8)

    plt.title(f'Singularity spectrum {title}')
    plt.xlabel(r'$\alpha$')
    plt.ylabel(r'$D(\alpha)$')
    plt.legend()

    if filename:
        plt.savefig(filename, format="svg", bbox_inches='tight')

    plt.tight_layout()
    plt.show()
```

```
In [ ]: grafica=pd.read_excel("Results/Results_to_analysis.xlsx",index_col=0)

data_dict = {}

for indice in grafica.columns:
    data_dict[indice] = {}
    for subindice in grafica.index:
        valor = grafica.loc[subindice, indice]
        data_dict[indice][subindice] = limpiar_y_convertir_a_array(valor)

data_dict
```

```

Out[ ]: {'1st return MSCI shuffle': {'alpha': array([0.92686131, 0.92557217, 0.922
79888, 0.91960166, 0.91590664,
0.91162701, 0.90666136, 0.90089216, 0.89418466, 0.88638655,
0.87732851, 0.86682563, 0.85467899, 0.84067468, 0.82457401,
0.80608079, 0.78476058, 0.75988369, 0.73025607, 0.69459263,
0.65353695, 0.61181866, 0.57574407, 0.5383634 , 0.52015562,
0.49949947, 0.48884156, 0.479338 , 0.47052242, 0.46209877,
0.45388528, 0.44577966, 0.43773588, 0.42974763, 0.42183585,
0.41403882, 0.40640418, 0.39898242, 0.39182187, 0.38496501,
0.3784461 , 0.37228998, 0.36651192, 0.36111816, 0.35610703,
0.35147029, 0.34719456, 0.34514501])},
'Dalpha': array([0.27485743, 0.28226997, 0.29784534, 0.3150018 , 0.33390
519,
0.35472889, 0.37764872, 0.40283483, 0.43044014, 0.46058461,
0.49333573, 0.52868664, 0.56653573, 0.60667505, 0.64880031,
0.69256164, 0.73767379, 0.78406669, 0.831869 , 0.88049432,
0.92641984, 0.96310263, 0.98617543, 0.99950374, 1.00031382,
0.99372761, 0.98715635, 0.9788951 , 0.9690092 , 0.95744403,
0.94410603, 0.92891257, 0.91182269, 0.89285439, 0.87209173,
0.84968385, 0.82583731, 0.80080375, 0.77486454, 0.74831441,
0.72144595, 0.69453622, 0.66783664, 0.64156615, 0.61590787,
0.59100842, 0.56697954, 0.55519461])},
'1st return FXBYFOOD shuffle': {'alpha': array([0.91568877, 0.91432825,
0.91140326, 0.90803531, 0.90414809,
0.89965207, 0.89444268, 0.88839856, 0.8813801 , 0.87322831,
0.86376423, 0.8527892 , 0.84008637, 0.82542486, 0.80857124,
0.78932078, 0.76757855, 0.74354201, 0.71797346, 0.69185943,
0.66317971, 0.62546062, 0.58312975, 0.54064884, 0.52237616,
0.50264706, 0.49232722, 0.4829244 , 0.47397007, 0.46515897,
0.45630177, 0.44729611, 0.43810782, 0.42875689, 0.41930508,
0.40984355, 0.4004801 , 0.3913268 , 0.382489 , 0.37405699,
0.36610074, 0.35866798, 0.35178479, 0.34545812, 0.33967927,
0.33442768, 0.32967456, 0.3274186 ]),
'Dalpha': array([0.28709526, 0.29491822, 0.31134579, 0.32941869, 0.34930
564,
0.37118265, 0.39522801, 0.4216148 , 0.45050036, 0.48201245,
0.51623183, 0.5531716 , 0.59275292, 0.63477494, 0.67887148,
0.72443595, 0.77048209, 0.8154199 , 0.85690801, 0.8928074 ,
0.92475877, 0.95694636, 0.98364329, 0.99931474, 1.00031558,
0.99401836, 0.98764343, 0.97945562, 0.9693984 , 0.95728465,
0.94288442, 0.92598751, 0.90645087, 0.88423361, 0.8594188 ,
0.83221949, 0.80296838, 0.7720926 , 0.74007842, 0.70743224,
0.6746443 , 0.64215984, 0.61035993, 0.57955213, 0.54996926,
0.52177414, 0.49506798, 0.48209625])},
'1st return SP500 shuffle': {'alpha': array([0.97258206, 0.97139305, 0.96
883095, 0.96586734, 0.96242921,
0.95842961, 0.95376539, 0.94831455, 0.94193358, 0.93445448,
0.92568152, 0.91538732, 0.90330726, 0.88913114, 0.87249057,
0.85294134, 0.82993784, 0.80278075, 0.77052654, 0.73216188,
0.68801736, 0.64200539, 0.60054016, 0.55594423, 0.53401336,
0.50908309, 0.49609754, 0.48437256, 0.47336201, 0.46274442,
0.45234191, 0.44207184, 0.43191628, 0.42190079, 0.41207831,
0.4025161 , 0.39328522, 0.38445221, 0.37607324, 0.36819054,
0.36083106, 0.35400678, 0.34771631, 0.3419472 , 0.33667846,
0.33188307, 0.32753017, 0.3254603 ]),
'Dalpha': array([0.25477517, 0.26161196, 0.27600078, 0.29190296, 0.30949
126,
0.32895125, 0.35047826, 0.37427238, 0.4005309 , 0.42943815,
0.46115267, 0.49579288, 0.53342254, 0.57403739, 0.61755234,
0.66378474, 0.71243242, 0.76306753, 0.8151279 , 0.86746822,
0.91681941, 0.95715784, 0.983564 , 0.99940575, 1.00037925,

```

```

0.99242333, 0.98440752, 0.97420557, 0.9618506 , 0.94726864,
0.93037421, 0.91112473, 0.88955158, 0.8657749 , 0.84000491,
0.81253207, 0.78370847, 0.75392389, 0.72358004, 0.6930667 ,
0.66274214, 0.6329195 , 0.60385921, 0.57576684, 0.54879507,
0.52304852, 0.49859007, 0.48668835])),
'1st return MSCI': {'alpha': array([0.94854491, 0.94686872, 0.9432584 ,
0.93909294, 0.93428888,
0.92875279, 0.92238142, 0.9150625 , 0.90667648, 0.89709911,
0.88620433, 0.87386558, 0.85995208, 0.84431467, 0.82675682,
0.80699238, 0.78461048, 0.75910664, 0.73008839, 0.69772193,
0.66319327, 0.62861521, 0.59616195, 0.55474593, 0.53092136,
0.4999671 , 0.48225985, 0.46598374, 0.45078103, 0.43641208,
0.42273475, 0.40968247, 0.39724198, 0.38543245, 0.37428742,
0.3638411 , 0.35411907, 0.34513325, 0.33688022, 0.32934197,
0.32248825, 0.31627955, 0.31067035, 0.30561202, 0.30105526,
0.29695197, 0.29325655, 0.29150529])),
'Dalpha': array([0.21104566, 0.22068377, 0.24095959, 0.26331177, 0.28788
99 ,
0.3148295 , 0.34424152, 0.37619893, 0.41072073, 0.44775463,
0.48716195, 0.52871101, 0.57208806, 0.61693275, 0.66289493,
0.70968684, 0.75706983, 0.80467785, 0.85160488, 0.89591786,
0.93468322, 0.96501223, 0.98548792, 0.99970617, 1.00023978,
0.99018519, 0.97922078, 0.96505545, 0.94801031, 0.92829932,
0.90611379, 0.88167875, 0.85528121, 0.82727395, 0.79806093,
0.76807242, 0.73773695, 0.70745596, 0.67758415, 0.64841734,
0.6201874 , 0.59306332, 0.56715676, 0.54253027, 0.51920667,
0.49717838, 0.47641576, 0.46634603])),
'1st return FXBYFOOD': {'alpha': array([0.88932922, 0.88792806, 0.8849836
2, 0.88173984, 0.87816505,
0.87422466, 0.86988123, 0.86509462, 0.85982249, 0.85402109,
0.84764636, 0.84065498, 0.83300466, 0.8246526 , 0.81554901,
0.80561254, 0.79462006, 0.78173659, 0.76385721, 0.73214781,
0.67566526, 0.60444819, 0.54696286, 0.50256212, 0.483312 ,
0.46035369, 0.44646618, 0.43272562, 0.41893394, 0.40506201,
0.39117235, 0.37737151, 0.36378147, 0.35052392, 0.33771239,
0.32544857, 0.31382018, 0.30289886, 0.29273786, 0.28337001,
0.27480684, 0.26703937, 0.26004072, 0.25376983, 0.24817585,
0.24320231, 0.23879066, 0.23671797])),
'Dalpha': array([0.39647177, 0.40452844, 0.42107313, 0.4384888 , 0.45678
79 ,
0.47597331, 0.49603529, 0.51694772, 0.53866355, 0.5611095 ,
0.58418068, 0.60773674, 0.63160148, 0.65556843, 0.67941665,
0.70296036, 0.72624278, 0.75023951, 0.77882878, 0.82116518,
0.88287544, 0.94518117, 0.98283517, 0.99939535, 1.000223 ,
0.99272873, 0.98406779, 0.97204441, 0.95652277, 0.93744484,
0.91487597, 0.88900866, 0.86014689, 0.82868471, 0.79508569,
0.75986442, 0.72356793, 0.68675464, 0.6499699 , 0.61371975,
0.57844697, 0.54451346, 0.51219164, 0.48166506, 0.45303654,
0.42634071, 0.40155847, 0.38964047])),
'1st return SP500': {'alpha': array([1.08235566, 1.08132447, 1.07912085,
1.07660724, 1.07372201,
1.07038841, 1.06651063, 1.06196883, 1.05661284, 1.05025386,
1.04265265, 1.03350062, 1.02238631, 1.00873241, 0.99168338,
0.96994228, 0.94165875, 0.90467879, 0.85742369, 0.79997958,
0.7350032 , 0.66844901, 0.60771818, 0.53699132, 0.4986468 ,
0.44991245, 0.42114423, 0.39338972, 0.36599072, 0.3386273 ,
0.31125162, 0.284044 , 0.25736193, 0.23166554, 0.20742499,
0.18503345, 0.164752 , 0.14669559, 0.13085122, 0.11711129,
0.10530847, 0.09524482, 0.08671329, 0.079512 , 0.07345292,
0.06836634, 0.06410263, 0.06215906])),
'Dalpha': array([0.14727856, 0.15320787, 0.16558559, 0.17907515, 0.19383

```

```

659,
    0.21005721, 0.2279546 , 0.24777931, 0.26981666, 0.29438844,
    0.32185683, 0.35263707, 0.38723184, 0.42630683, 0.47081612,
    0.52210913, 0.58173595, 0.65046199, 0.7265785 , 0.80496355,
    0.87772105, 0.93609968, 0.97464066, 0.99932951, 1.00041796,
    0.98454847, 0.96665821, 0.94240984, 0.91159358, 0.87396572,
    0.82948185, 0.77848696, 0.72183386, 0.66088185, 0.59735547,
    0.53310586, 0.46986402, 0.40906406, 0.3517643 , 0.29864906,
    0.25007757, 0.20615134, 0.1667823 , 0.13175303, 0.10076598,
    0.07348142, 0.04954513, 0.03836959]]},
    '2nd return MSCI shuffle': {'alpha': array([0.99267352, 0.9912854 , 0.988
31008, 0.98490424, 0.98099789,
    0.97650984, 0.9713465 , 0.96540067, 0.95855066, 0.95065939,
    0.94157337, 0.93112048, 0.91910505, 0.90529756, 0.8894155 ,
    0.87108852, 0.84978509, 0.82462456, 0.793923 , 0.75467659,
    0.70435903, 0.64763726, 0.59724574, 0.55037242, 0.52981218,
    0.507487 , 0.49513497, 0.48300049, 0.47051468, 0.4573573 ,
    0.44337833, 0.42856795, 0.41304216, 0.39702681, 0.38083037,
    0.36480407, 0.34929569, 0.33460734, 0.32096669, 0.3085156 ,
    0.29731428, 0.28735551, 0.27858291, 0.27090879, 0.26422907,
    0.25843434, 0.2534174 , 0.25109029])},
    'Dalpha': array([0.22625662, 0.23423831, 0.25094959, 0.26922705, 0.28921
349,
    0.31105364, 0.33488879, 0.36084931, 0.38904505, 0.41955409,
    0.45241118, 0.48759856, 0.52504258, 0.56461931, 0.60617021,
    0.64953068, 0.69459451, 0.74149432, 0.79096791, 0.84428007,
    0.9001551 , 0.94971835, 0.98207263, 0.99923031, 1.00036923,
    0.99320779, 0.98552391, 0.97489729, 0.96081579, 0.9426754 ,
    0.91990587, 0.89208619, 0.85905467, 0.82099624, 0.77847992,
    0.73242624, 0.68400538, 0.63449189, 0.58511579, 0.53694523,
    0.49081859, 0.44732548, 0.40682395, 0.36947821, 0.33530338,
    0.3042086 , 0.27603362, 0.26265274])},
    '2nd return FXBYFOOD shuffle': {'alpha': array([0.96566761, 0.96424607,
0.96120054, 0.95771705, 0.95372399,
    0.94913811, 0.94386313, 0.93778833, 0.93078708, 0.92271518,
    0.91340836, 0.90267734, 0.89029747, 0.8759873 , 0.85936907,
    0.83990812, 0.81685011, 0.78922801, 0.75609118, 0.71716972,
    0.67406822, 0.63104244, 0.59330284, 0.55298949, 0.53296196,
    0.5095261 , 0.4964501 , 0.48388606, 0.47131817, 0.45845604,
    0.44516503, 0.4314327 , 0.41734814, 0.40308122, 0.3888561 ,
    0.3749197 , 0.36150969, 0.34882785, 0.33702327, 0.32618677,
    0.31635448, 0.30751741, 0.29963332, 0.29263826, 0.28645631,
    0.28100703, 0.27621061, 0.27396581])},
    'Dalpha': array([0.23161719, 0.23979106, 0.25689683, 0.27559115, 0.29602
135,
    0.31833768, 0.34268814, 0.3692117 , 0.39802975, 0.42923665,
    0.46289123, 0.49901306, 0.53758941, 0.57860059, 0.6220655 ,
    0.66808786, 0.71683388, 0.76830706, 0.82178332, 0.87494833,
    0.92326701, 0.96109454, 0.98516259, 0.99947758, 1.00031238,
    0.99276654, 0.98464642, 0.9736645 , 0.95951352, 0.94180341,
    0.9201766 , 0.89440223, 0.86445476, 0.83056582, 0.79323511,
    0.75319384, 0.7113275 , 0.66857733, 0.62584434, 0.58391533,
    0.54342023, 0.50481936, 0.46841326, 0.43436564, 0.40273147,
    0.37348483, 0.34654334, 0.33363576])},
    '2nd return MVOFTR shuffle': {'alpha': array([0.98160192, 0.98019644, 0.9
7718143, 0.9737251 , 0.96975585,
    0.96519114, 0.9599366 , 0.9538854 , 0.94691812, 0.93890316,
    0.92969762, 0.91914843, 0.90709213, 0.89335093, 0.8777192 ,
    0.85992879, 0.83956762, 0.81590256, 0.78754258, 0.7520323 ,
    0.70685934, 0.65501133, 0.60752251, 0.56140862, 0.54017471,
    0.51568621, 0.50131706, 0.48684006, 0.47166215, 0.45542745,

```

```

0.43797593, 0.41934158, 0.39975322, 0.37961147, 0.35942983,
0.33975033, 0.3210602 , 0.30373394, 0.28800974, 0.27399487,
0.26168857, 0.25101146, 0.24183436, 0.23400229, 0.22735248,
0.22172642, 0.21697711, 0.21480415]),
'Dalpha': array([0.25035422, 0.25843572, 0.27536967, 0.29391778, 0.31422
572,
0.33643871, 0.36069472, 0.38711538, 0.41579421, 0.44678244,
0.48007345, 0.51558806, 0.55316452, 0.59255933, 0.63346744,
0.67557501, 0.71866581, 0.7628015 , 0.8085359 , 0.85681931,
0.90697428, 0.95217152, 0.9825667 , 0.99931661, 1.00031517,
0.99238084, 0.98341618, 0.97071926, 0.95358604, 0.9311893 ,
0.90275251, 0.86774332, 0.82606859, 0.77821223, 0.72525013,
0.66871962, 0.6103886 , 0.55200734, 0.49511248, 0.44091315,
0.39025494, 0.34364097, 0.30128602, 0.26318367, 0.22917217,
0.19899126, 0.17232684, 0.15983232])),
'2nd return SP500 shuffle': {'alpha': array([1.01190301, 1.01056553, 1.00
768611, 1.00436166, 1.00051338,
0.99604811, 0.99085618, 0.98480915, 0.97775749, 0.96952824,
0.9599225 , 0.94871192, 0.9356331 , 0.92037692, 0.90256812,
0.88172633, 0.85719693, 0.82806113, 0.79318921, 0.75201082,
0.70636703, 0.66126414, 0.62190334, 0.57787753, 0.55430607,
0.5241296 , 0.50552523, 0.4866032 , 0.46677674, 0.44578802,
0.42367673, 0.40074722, 0.37751063, 0.35459327, 0.33262493,
0.31213699, 0.29349778, 0.27689516, 0.26235744, 0.24979447,
0.2390422 , 0.22990045, 0.22216022, 0.21562097, 0.21010001,
0.20543648, 0.20149188, 0.19968338])),
'Dalpha': array([0.20239077, 0.21008129, 0.2262525 , 0.24409131, 0.26377
835,
0.28550484, 0.30946838, 0.33586636, 0.36488666, 0.3966956 ,
0.43142376, 0.46915154, 0.50989726, 0.5536122 , 0.60018779,
0.64948042, 0.70135106, 0.75565923, 0.81193054, 0.86815807,
0.91934191, 0.95903997, 0.9841252 , 0.99956702, 1.00024524,
0.99037166, 0.97874784, 0.96214745, 0.93977325, 0.91083791,
0.87484259, 0.83181194, 0.78243362, 0.72804537, 0.67045661,
0.61166068, 0.55353736, 0.49763391, 0.4450624 , 0.39650001,
0.35225411, 0.31235303, 0.27663555, 0.24482573, 0.21658911,
0.19157154, 0.16942411, 0.15902524])),
'2nd return MSCI': {'alpha': array([0.99207861, 0.9909409 , 0.98852088,
0.98579012, 0.98270224,
0.97920387, 0.97523411, 0.97072451, 0.96560017, 0.95978315,
0.95319967, 0.94579287, 0.93753953, 0.92845834, 0.91856746,
0.90767741, 0.89474816, 0.87636694, 0.84493712, 0.79184646,
0.71853032, 0.64174454, 0.58231842, 0.53960048, 0.52517267,
0.51244228, 0.50499692, 0.49609148, 0.48486728, 0.47083655,
0.45382903, 0.43397881, 0.41171202, 0.38771125, 0.36284246,
0.33804653, 0.31421543, 0.29208319, 0.27215745, 0.25470239,
0.23976563, 0.2272312 , 0.21687924, 0.20843863, 0.20162636,
0.19617284, 0.1918356 , 0.18991771])),
'Dalpha': array([0.31045293, 0.31699474, 0.33058929, 0.34524637, 0.36104
788,
0.37807498, 0.39640363, 0.41609716, 0.4371942 , 0.45968943,
0.48350448, 0.50844958, 0.5341883 , 0.56024621, 0.58615906,
0.61194872, 0.63924268, 0.67320668, 0.72314986, 0.79457219,
0.87610198, 0.9438067 , 0.98260082, 0.99905577, 1.00035284,
0.99629753, 0.99158622, 0.98366941, 0.97087689, 0.95139911,
0.92357533, 0.88618739, 0.83873716, 0.78165189, 0.71634626,
0.6450921 , 0.57070638, 0.49613597, 0.42405507, 0.35657563,
0.29511734, 0.24042368, 0.19267421, 0.15163678, 0.11681692,
0.08758108, 0.06324679, 0.05221896])),
'2nd return FXBYFOOD': {'alpha': array([0.94329536, 0.9420374 , 0.9393466
6, 0.93627881, 0.93277567,

```

```

0.92877325, 0.92420291, 0.91899271, 0.91306772, 0.90634606,
0.8987242 , 0.89004105, 0.8800084 , 0.86810989, 0.8535158 ,
0.83511632, 0.81173966, 0.78251634, 0.74740103, 0.70772568,
0.66648448, 0.62822517, 0.5972468 , 0.5662177 , 0.54974164,
0.52626749, 0.50852128, 0.48841397, 0.46595961, 0.44142265,
0.41522237, 0.38789478, 0.36006735, 0.33242213, 0.30564057,
0.28033845, 0.25700693, 0.23597482, 0.2173996 , 0.20128473,
0.18751384, 0.17589087, 0.16617728, 0.15812158, 0.15147975,
0.14602745, 0.14156603, 0.1395599 ]),
'Dalpha': array([ 0.29956267, 0.30679598, 0.32190951, 0.33837393, 0.
35629841,
0.37577692, 0.39687705, 0.41962938, 0.44402288, 0.47001682,
0.49758603, 0.52681901, 0.55807501, 0.59214602, 0.63025576,
0.6736786 , 0.72300778, 0.77741691, 0.83412739, 0.88846287,
0.93488162, 0.96870886, 0.98863011, 0.99969992, 1.00002565,
0.99208186, 0.98084745, 0.96310143, 0.93769902, 0.90384163,
0.86117733, 0.80988603, 0.75074234, 0.68511813, 0.61489129,
0.54225787, 0.46948304, 0.39865091, 0.3314716 , 0.26917818,
0.21251456, 0.16179123, 0.11697736, 0.07780159, 0.04384317,
0.0146048 , -0.01043457, -0.02196976])),
'2nd return MVOFTR': {'alpha': array([1.20861664, 1.20730268, 1.20444076,
1.20107376, 1.19712277,
1.19249902, 1.18710319, 1.18082446, 1.17353869, 1.16510434,
1.15535347, 1.14407238, 1.13096258, 1.11556991, 1.09717931,
1.0747258 , 1.0468832 , 1.01244778, 0.97046661, 0.91928506,
0.8571628 , 0.78843742, 0.72499331, 0.65688778, 0.6224083 ,
0.58042251, 0.55607727, 0.53246348, 0.50869947, 0.48416691,
0.45849087, 0.43156803, 0.40360256, 0.3751114 , 0.34686621,
0.31976189, 0.29464141, 0.27213794, 0.25259025, 0.23604703,
0.22233408, 0.21114284, 0.20210895, 0.19486751, 0.18908489,
0.18447288, 0.18079168, 0.1791568 ]),
'Dalpha': array([ 0.08647689, 0.09403215, 0.1101012 , 0.12816494, 0.
14837465,
0.17087046, 0.19577465, 0.22318524, 0.25317201, 0.2857786 ,
0.32103745, 0.35900789, 0.39985068, 0.44394459, 0.49200518,
0.54503915, 0.60382416, 0.66794698, 0.73566676, 0.80539096,
0.87456134, 0.93458782, 0.97500877, 0.99919728, 1.00044167,
0.98682426, 0.97168629, 0.95103785, 0.92427093, 0.89047499,
0.84867216, 0.79811525, 0.73863488, 0.67095642, 0.5968555 ,
0.51903047, 0.44067707, 0.36490691, 0.29423695, 0.23031711,
0.17391984, 0.12510472, 0.0834461 , 0.04824502, 0.01869047,
-0.00603413, -0.02668944, -0.03609004])),
'2nd return SP500': {'alpha': array([0.97296512, 0.97169825, 0.96895445,
0.96574662, 0.961978 ,
0.95753311, 0.95227644, 0.94605239, 0.93868699, 0.92999148,
0.91976714, 0.90780982, 0.89391119, 0.87785236, 0.85938216,
0.83816238, 0.81364391, 0.78484074, 0.75014712, 0.70793424,
0.65881805, 0.60759963, 0.56065843, 0.50514179, 0.474235 ,
0.43354138, 0.40819433, 0.38245762, 0.35542906, 0.32661252,
0.29606859, 0.26446134, 0.23291139, 0.20269703, 0.17493803,
0.15038569, 0.12936396, 0.11183018, 0.09749252, 0.08592993,
0.07668526, 0.06932424, 0.06346547, 0.05879029, 0.05504114,
0.05201441, 0.04955134, 0.04844339])),
'Dalpha': array([0.17038726, 0.17767179, 0.19307941, 0.21028974, 0.22956
556,
0.25118821, 0.27544503, 0.30260961, 0.33291484, 0.36652074,
0.40348209, 0.44372328, 0.48702859, 0.53305563, 0.58138059,
0.63159318, 0.68346698, 0.73716529, 0.79311375, 0.85064522,
0.90554925, 0.95045428, 0.98023828, 0.99953933, 1.0002699 ,
0.98691922, 0.97108621, 0.94850898, 0.91799798, 0.87825561,
0.82852541, 0.76922522, 0.70222533, 0.63058942, 0.55790579,

```

```

0.48753491, 0.42206623, 0.36310148, 0.31131522, 0.2666693 ,
0.22866563, 0.19656605, 0.16955216, 0.14682565, 0.12766204,
0.1114329 , 0.09760902, 0.09123832]}}},
'fgn': {'alpha': array([0.85936075, 0.85763151, 0.85397713, 0.84990975,
0.8453854 ,
0.84035717, 0.83477514, 0.828586 , 0.82173198, 0.81414864,
0.8057608 , 0.79647595, 0.78617455, 0.77469645, 0.76182201,
0.74724141, 0.73049511, 0.71085297, 0.68712495, 0.65764393,
0.6213845 , 0.58116188, 0.54513812, 0.51320939, 0.50072943,
0.48943915, 0.48486695, 0.48133112, 0.47840758, 0.47581811,
0.47337229, 0.47093813, 0.46842583, 0.4657778 , 0.46296181,
0.45996574, 0.4567932 , 0.45345962, 0.44998884, 0.44641006,
0.44275535, 0.43905747, 0.43534828, 0.4316575 , 0.42801192,
0.42443489, 0.42094611, 0.41922601])},
'Dalpha': array([0.33323527, 0.34317841, 0.36370978, 0.38554483, 0.40870
215,
0.43318175, 0.45896241, 0.48600024, 0.51422963, 0.54356735,
0.57392043, 0.60519749, 0.63732161, 0.67024183, 0.703942 ,
0.73845289, 0.7738861 , 0.81050581, 0.84876241, 0.88888113,
0.92924337, 0.96437237, 0.98747787, 0.99937238, 1.00032224,
0.99684724, 0.99407052, 0.9910253 , 0.98776423, 0.98421756,
0.98024719, 0.9756805 , 0.97033475, 0.96403583, 0.95663269,
0.94800765, 0.93808277, 0.92682251, 0.91423318, 0.90035967,
0.88528026, 0.86910036, 0.8519456 , 0.83395509, 0.81527511,
0.79605354, 0.77643522, 0.76654466]})}}

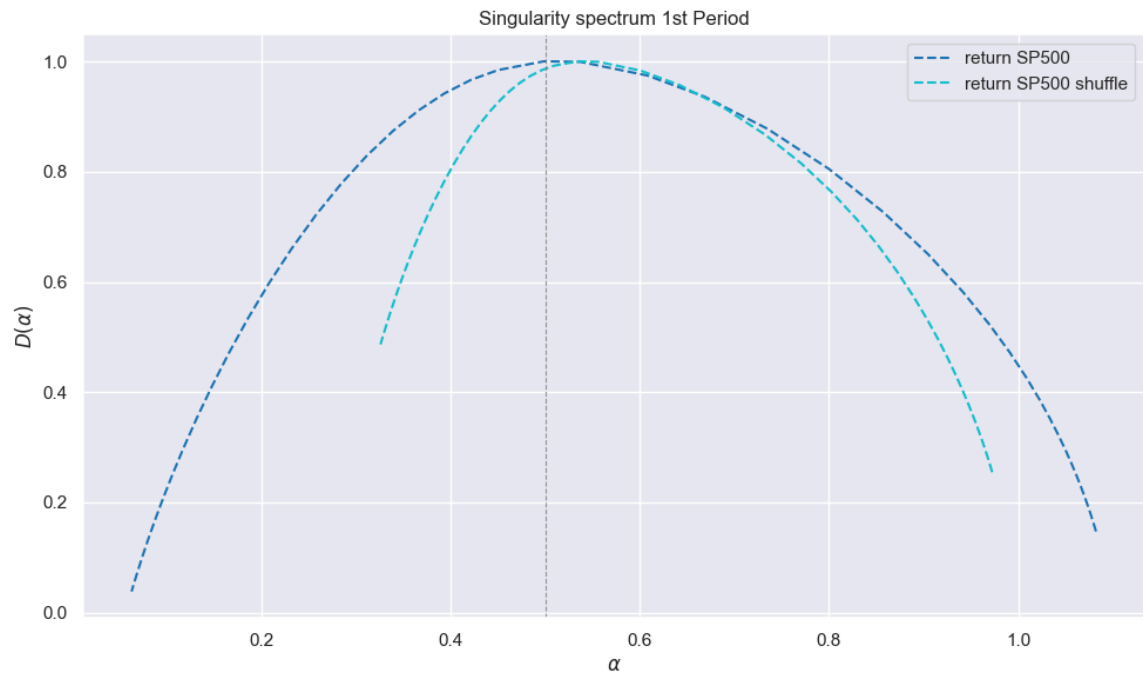
```

SP500 NORMAL VS SHUFFLE 1st period


```
In [69]: combined_dict = {
    "return SP500": data_dict.get("1st return SP500", {}),
    "return SP500 shuffle": data_dict.get("1st return SP500 shuffle", {})
}
plot_multifractal_unico(combined_dict, filename="MF-DFA 1st period comparison SP500 normal vs shuffle.svg", title="1st Period")
```

C:\Users\gabid\AppData\Local\Temp\ipykernel_21960\2145195050.py:4: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be removed in 3.11. Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get\_cmap()`` or ``pyplot.get\_cmap()`` instead.

cmap = plt.cm.get_cmap('tab10', len(series_names)) # Se limita a la cantidad de series

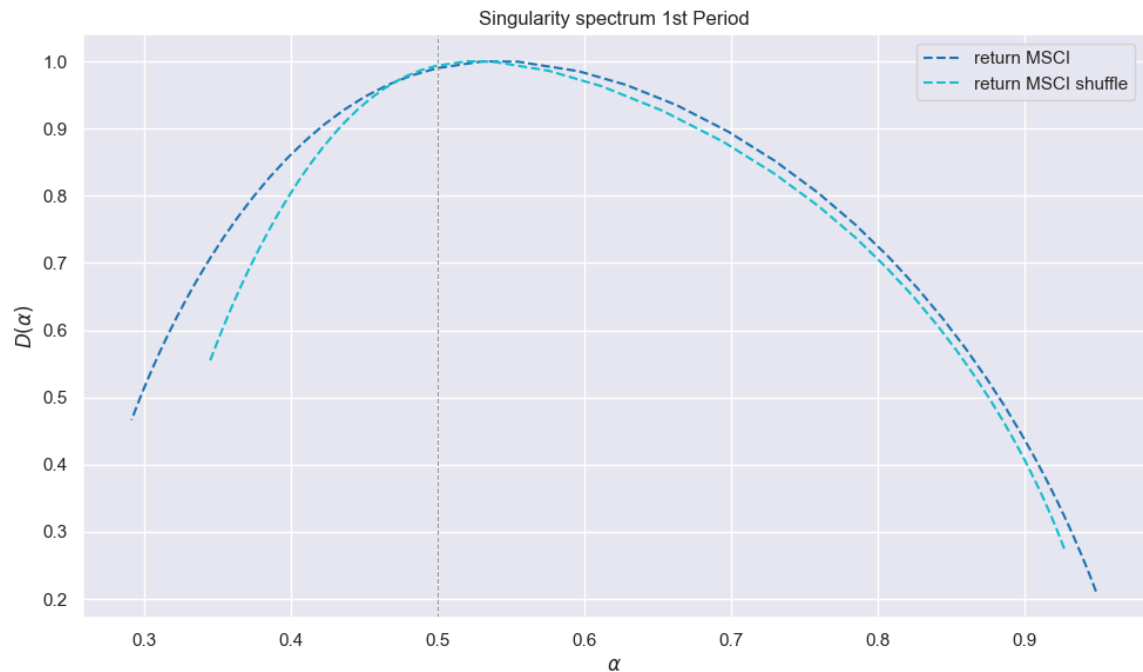


MSCI NORMAL VS SHUFFLE 1st period

```
In [70]: combined_dict = {
    "return MSCI": data_dict.get("1st return MSCI", {}),
    "return MSCI shuffle": data_dict.get("1st return MSCI shuffle", {})
}
plot_multifractal_unico(combined_dict, filename="MF-DFA 1st period comparison MSCI normal vs shuffle.svg", title="1st Period")
```

C:\Users\gabid\AppData\Local\Temp\ipykernel_21960\2145195050.py:4: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be removed in 3.11. Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get\_cmap()`` or ``pyplot.get\_cmap()`` instead.

cmap = plt.cm.get_cmap('tab10', len(series_names)) # Se limita a la cantidad de series

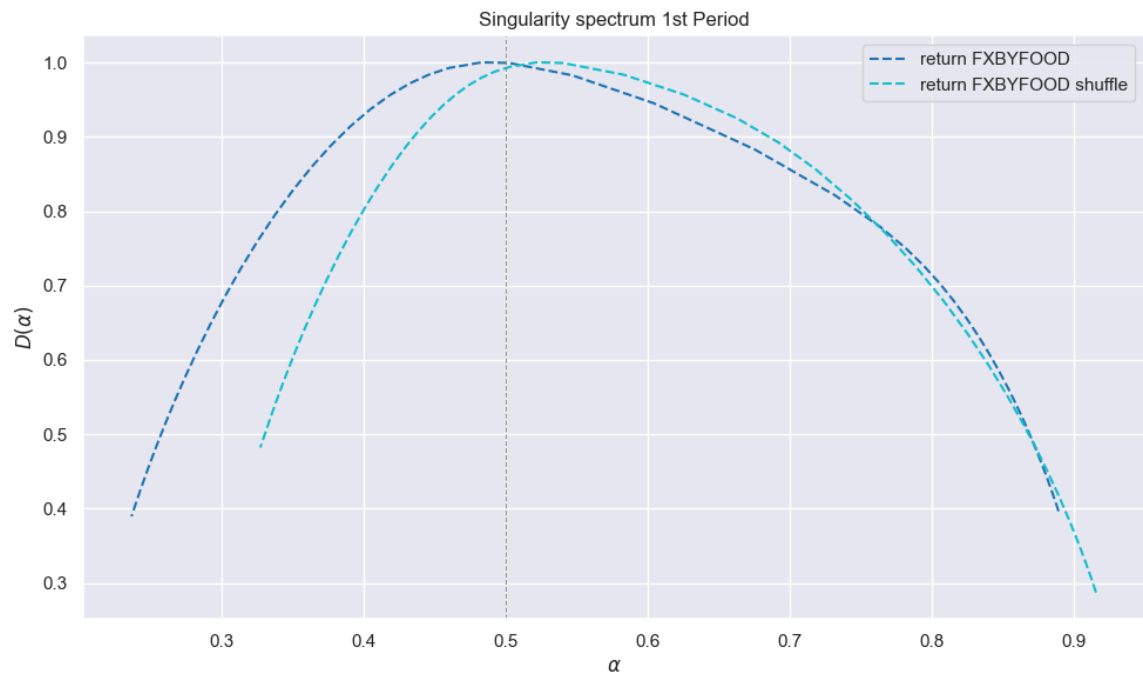


FXBYFOOD NORMAL VS SHUFFLE 1st period

```
In [71]: combined_dict = {
    "return FXBYFOOD": data_dict.get("1st return FXBYFOOD", {}),
    "return FXBYFOOD shuffle": data_dict.get("1st return FXBYFOOD shuffle",
    {}))
    }
    plot_multifractal_unico(combined_dict, filename="MF-DFA 1st period comparis
    on FXBYFOOD normal vs shuffle.svg", title="1st Period")
```

C:\Users\gabid\AppData\Local\Temp\ipykernel_21960\2145195050.py:4: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be removed in 3.11. Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get\_cmap()`` or ``pyplot.get\_cmap()`` instead.

cmap = plt.cm.get_cmap('tab10', len(series_names)) # Se limita a la cantidad de series

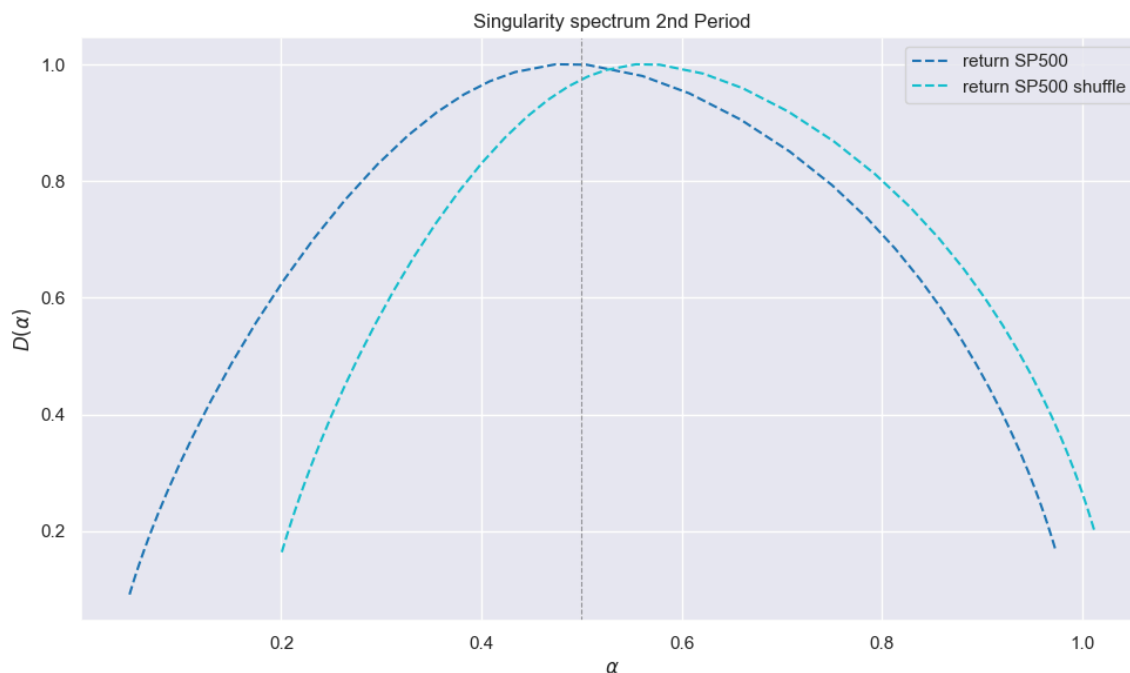


SP500 NORMAL VS SHUFFLE 2nd period

```
In [68]: combined_dict = {
    "return SP500": data_dict.get("2nd return SP500", {}),
    "return SP500 shuffle": data_dict.get("2nd return SP500 shuffle", {})
}
plot_multifractal_unico(combined_dict, filename="MF-DFA 2nd period comparis
on SP500 normal vs shuffle.svg", title="2nd Period")
```

C:\Users\gabid\AppData\Local\Temp\ipykernel_21960\2145195050.py:4: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be removed in 3.11. Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get\_cmap()`` or ``pyplot.get\_cmap()`` instead.

cmap = plt.cm.get_cmap('tab10', len(series_names)) # Se limita a la cantidad de series

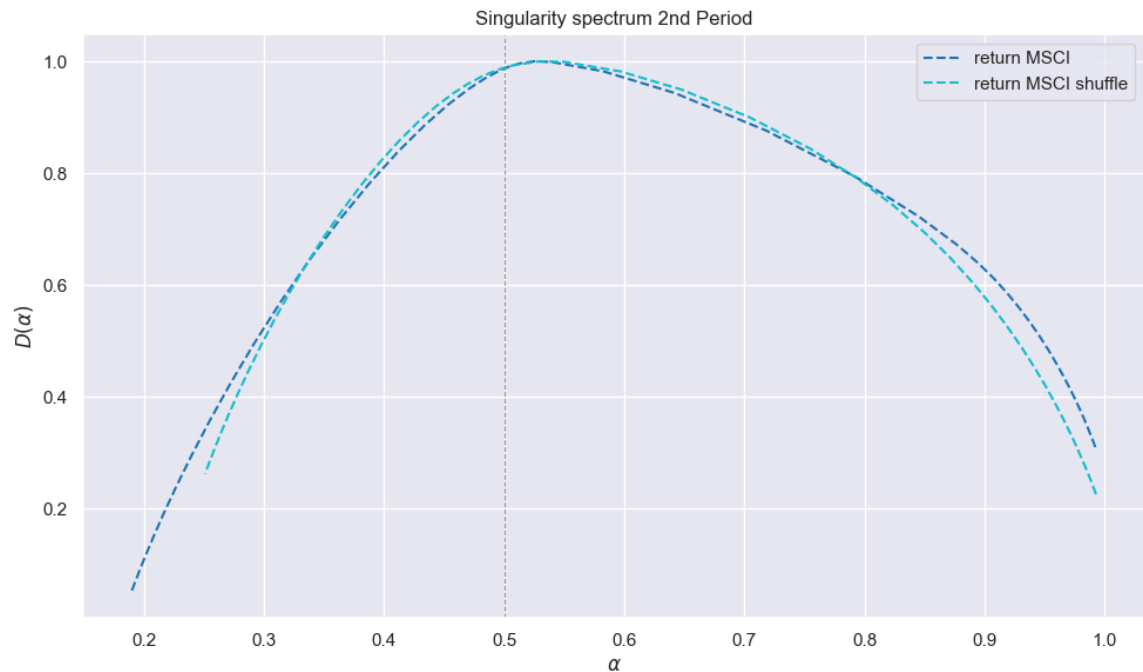


MSCI NORMAL VS SHUFFLE 2nd period

```
In [67]: combined_dict = {
    "return MSCI": data_dict.get("2nd return MSCI", {}),
    "return MSCI shuffle": data_dict.get("2nd return MSCI shuffle", {})
}
plot_multifractal_unico(combined_dict, filename="MF-DFA 2nd period comparison MSCI normal vs shuffle.svg", title="2nd Period")
```

C:\Users\gabid\AppData\Local\Temp\ipykernel_21960\2145195050.py:4: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be removed in 3.11. Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get\_cmap()`` or ``pyplot.get\_cmap()`` instead.

cmap = plt.cm.get_cmap('tab10', len(series_names)) # Se limita a la cantidad de series

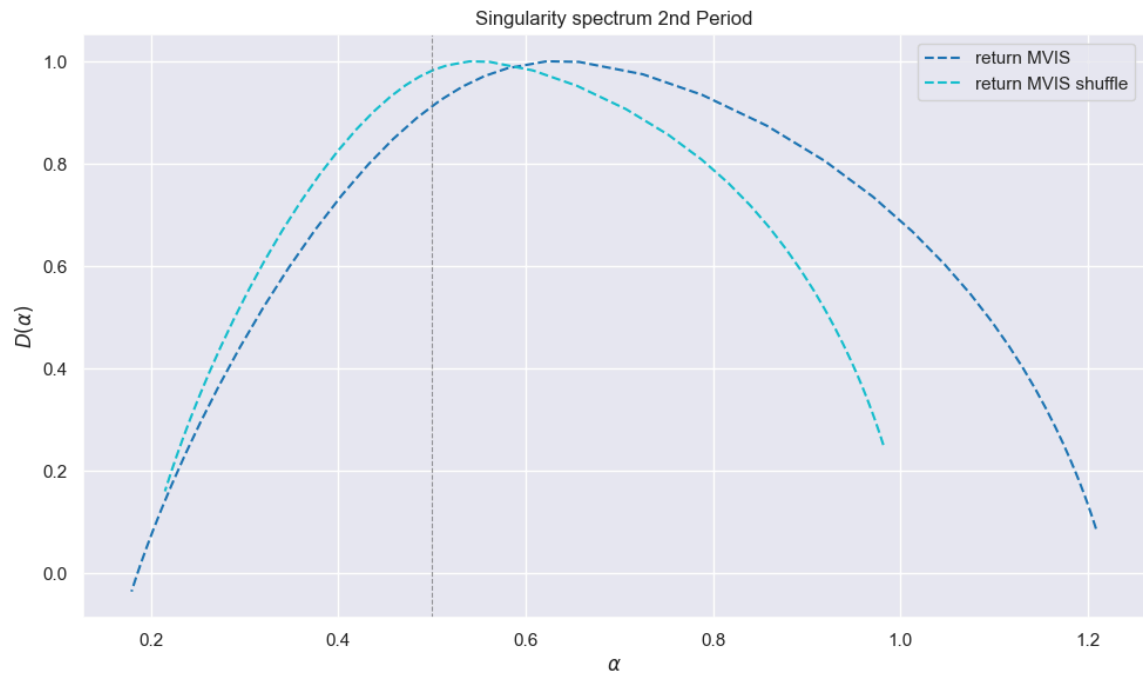


MVIS NORMAL VS SHUFFLE 2nd period

```
In [66]: combined_dict = {
    "return MVIS": data_dict.get("2nd return MVOFTR", {}),
    "return MVIS shuffle": data_dict.get("2nd return MVOFTR shuffle", {})
}
plot_multifractal_unico(combined_dict, filename="MF-DFA 2nd period comparison MVIS normal vs shuffle.svg", title="2nd Period")
```

C:\Users\gabid\AppData\Local\Temp\ipykernel_21960\2145195050.py:4: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be removed in 3.11. Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get\_cmap()`` or ``pyplot.get\_cmap()`` instead.

cmap = plt.cm.get_cmap('tab10', len(series_names)) # Se limita a la cantidad de series

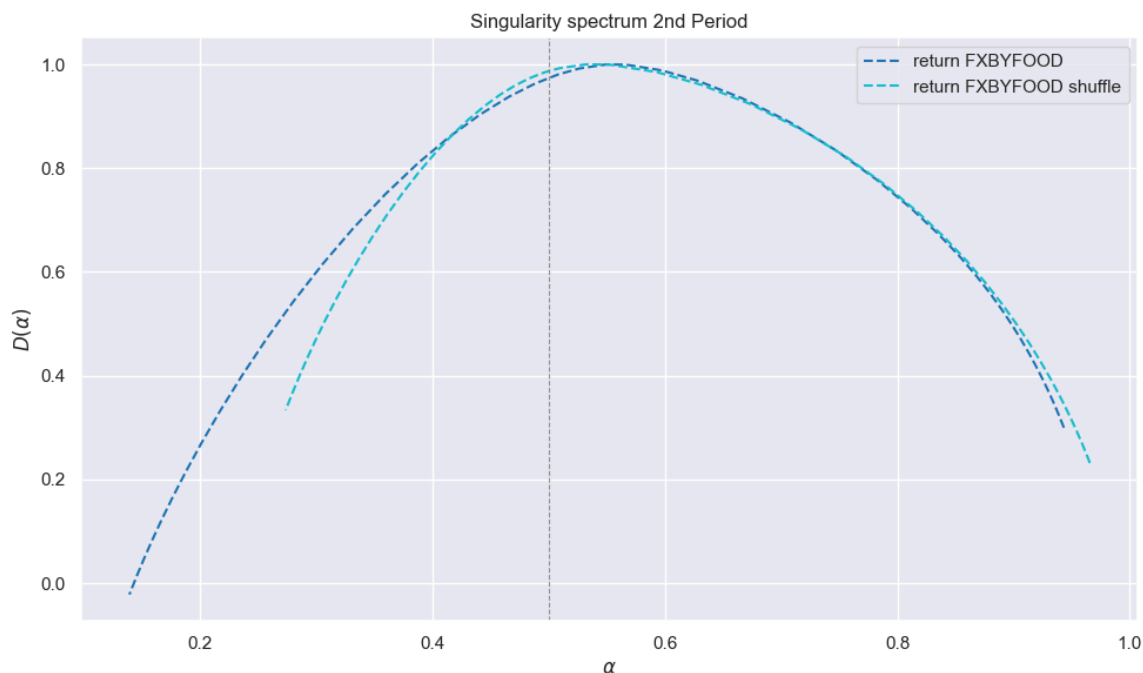


FXBYFOOD NORMAL VS SHUFFLE 2nd period

```
In [65]: combined_dict = {
    "return FXBYFOOD": data_dict.get("2nd return FXBYFOOD", {}),
    "return FXBYFOOD shuffle": data_dict.get("2nd return FXBYFOOD shuffle",
    {}))
    }
    plot_multifractal_unico(combined_dict, filename="MF-DFA 2nd period comparis
    on FXBYFOOD normal vs shuffle.svg", title="2nd Period")
```

C:\Users\gabid\AppData\Local\Temp\ipykernel_21960\2145195050.py:4: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be removed in 3.11. Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get\_cmap()`` or ``pyplot.get\_cmap()`` instead.

cmap = plt.cm.get_cmap('tab10', len(series_names)) # Se limita a la cantidad de series

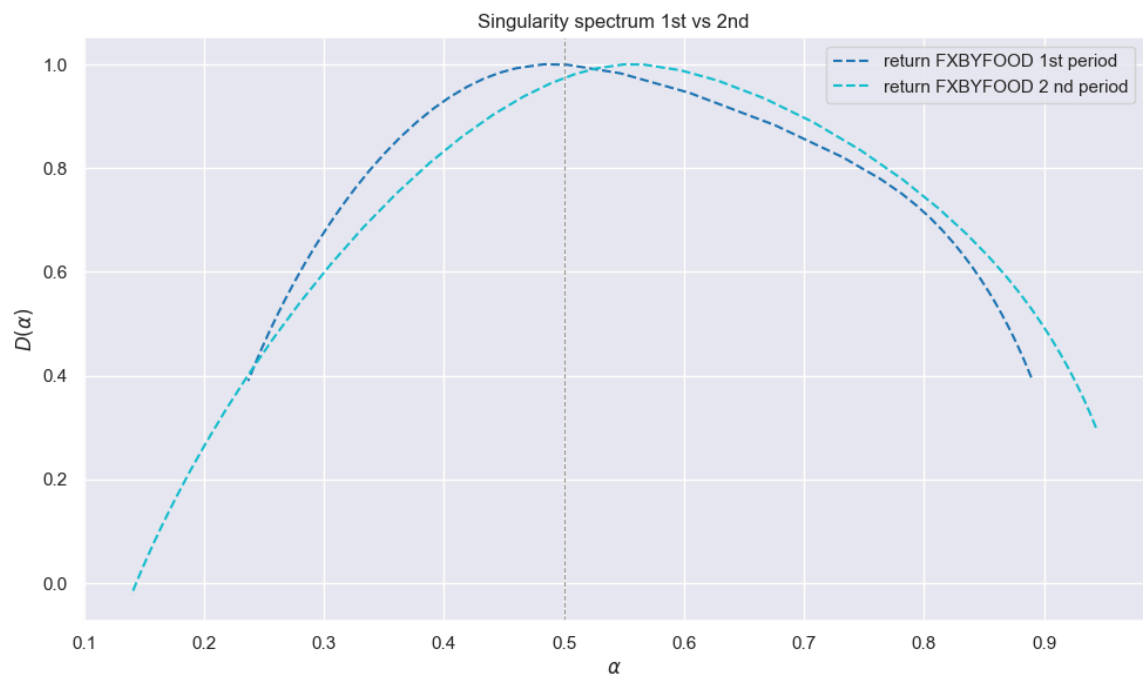


FXBYFOOD NORMAL 1st VS 2nd period

```
In [7]: combined_dict = {
    "return FXBYFOOD 1st period": data_dict.get("1st return FXBYFOOD", {}),
    "return FXBYFOOD 2 nd period": data_dict.get("2nd return FXBYFOOD", {})
}
plot_multifractal_unico(combined_dict, filename="MF-DFA period comparison F
XBYFOOD.svg", title="1st vs 2nd")
```

C:\Users\gabid\AppData\Local\Temp\ipykernel_18176\2145195050.py:4: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be removed in 3.11. Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get\_cmap()`` or ``pyplot.get\_cmap()`` instead.

cmap = plt.cm.get_cmap('tab10', len(series_names)) # Se limita a la cantidad de series

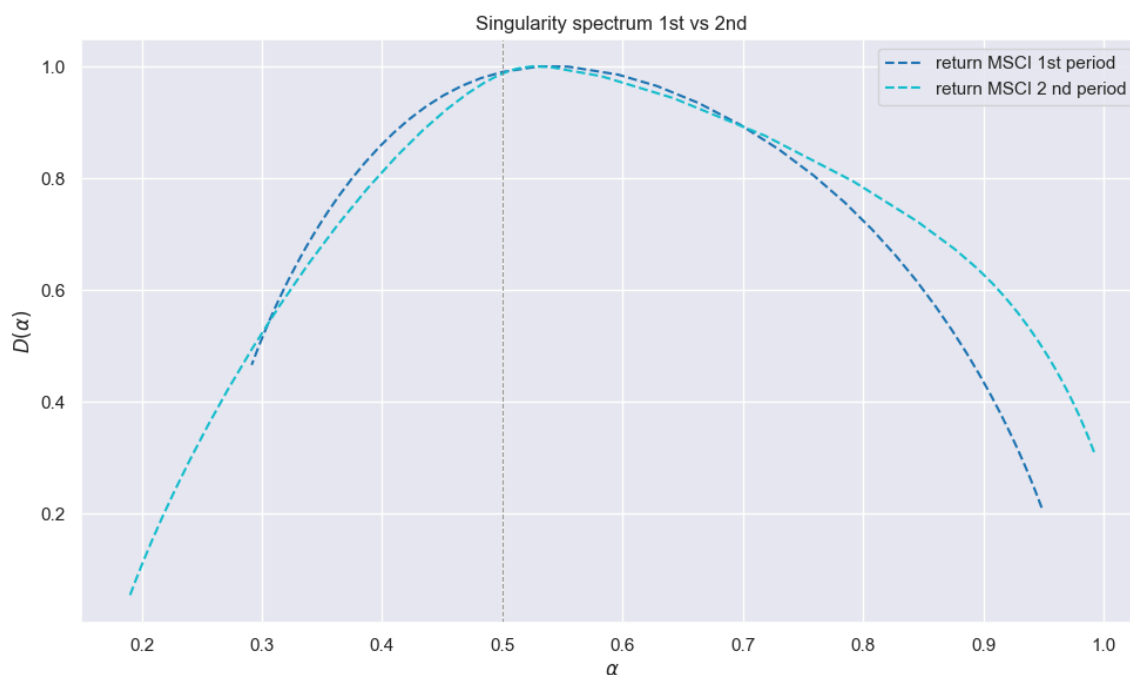


MSCI NORMAL 1st VS 2nd period


```
In [8]: combined_dict = {
    "return MSCI 1st period": data_dict.get("1st return MSCI", {}),
    "return MSCI 2 nd period": data_dict.get("2nd return MSCI", {})
}
plot_multifractal_unico(combined_dict, filename="MF-DFA period comparison M
SCI.svg", title="1st vs 2nd")
```

C:\Users\gabid\AppData\Local\Temp\ipykernel_18176\2145195050.py:4: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be removed in 3.11. Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get\_cmap()`` or ``pyplot.get\_cmap()`` instead.

cmap = plt.cm.get_cmap('tab10', len(series_names)) # Se limita a la cantidad de series

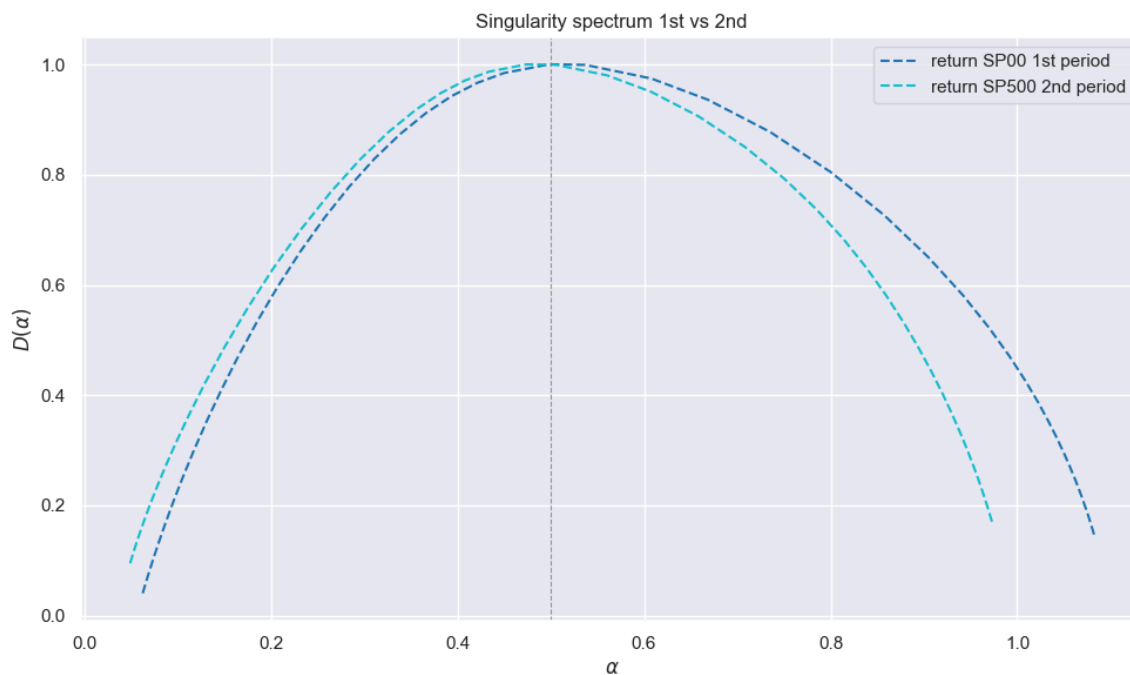


SP500 NORMAL 1st VS 2nd period

```
In [9]: combined_dict = {
    "return SP00 1st period": data_dict.get("1st return SP500", {}),
    "return SP500 2nd period": data_dict.get("2nd return SP500", {})
}
plot_multifractal_unico(combined_dict, filename="MF-DFA period comparison S
P500.svg", title="1st vs 2nd")
```

C:\Users\gabid\AppData\Local\Temp\ipykernel_18176\2145195050.py:4: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be removed in 3.11. Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get\_cmap()`` or ``pyplot.get\_cmap()`` instead.

cmap = plt.cm.get_cmap('tab10', len(series_names)) # Se limita a la cantidad de series



In []: