

analisisCaos2

June 24, 2024

1 Análisis Estadístico de Series de Tiempo Caóticas

1.1 Alumno: Rodrigo Gerardo Trejo Arriaga

1.1.1 Título de la Práctica: Estadísticas descriptivas de atractores caóticos

Este segmento de la práctica explora las propiedades estadísticas de los conjuntos de datos generados por modelos caóticos. Se calcularán métricas como la media, mediana, entropía, kurtosis, varianza y desviación estándar para demostrar que los datos no siguen distribuciones de probabilidad convencionales como las normales, uniformes o gamma.

Fecha de Entrega: **24 de Junio, 2024**

```
[2]: import numpy as np
from scipy.stats import gmean, skew, kurtosis, mode
from scipy import stats
import matplotlib.pyplot as plt
import pandas as pd
import plotly.graph_objects as go
import plotly.io as pio
from sklearn.neighbors import KDTree
from joblib import Parallel, delayed
```

```
[9]: def convertir_camelCase(text):
    cleaned_text = ''.join(char for char in text if char.isalnum() or char.
    ↪isspace())
    words = cleaned_text.split()
    return words[0].lower() + ''.join(word.capitalize() for word in words[1:])
```

1.2 Clase DistribucionProbabilidad

La clase `DistribucionProbabilidad` está diseñada para analizar conjuntos de datos mediante una variedad de métricas estadísticas. Es útil en estudios de series de tiempo y análisis de datos donde se requiere un entendimiento profundo de las propiedades estadísticas de una o más variables.

1.2.1 Métodos y Métricas Estadísticas:

- **Media:** Calcula el promedio de los valores en el conjunto de datos.

- **Mediana:** Determina el valor medio que divide el conjunto de datos en dos partes iguales.
- **Moda:** Identifica el valor o valores más frecuentes en el conjunto de datos.
- **Media Geométrica:** Calcula la media multiplicativa de los valores del conjunto.
- **Asimetría (Skewness):** Mide la asimetría de la distribución de los datos.
- **Rango:** La diferencia entre el valor máximo y mínimo en el conjunto.
- **Desviación Estándar:** Mide la cantidad de variación o dispersión de los datos.
- **Varianza:** Calcula la varianza de los datos.
- **Coeficiente de Variación:** Relaciona la desviación estándar con la media, útil para comparar la dispersión entre distribuciones con diferentes escalas.
- **Percentiles y Cuartiles:** Determina valores específicos que dividen el conjunto de datos en intervalos iguales.
- **Curtosis:** Mide la ‘agudeza’ o ‘achatamiento’ de la distribución respecto a una distribución normal.
- **Entropía:** Mide la incertidumbre o la cantidad de información ‘sorpresa’ en la distribución de los datos.

2 Análisis Caótico de Series de Tiempo

Además, se realiza un análisis caótico de cada uno de los modelos presentados con el objetivo de demostrar que estas series de tiempo están influenciadas por atractores caóticos en lugar de comportamientos estocásticos. Para ello, proponemos utilizar las siguientes métricas:

2.1 Exponentes de Lyapunov

2.1.1 Definición

Los exponentes de Lyapunov son una medida cuantitativa de la sensibilidad a las condiciones iniciales en un sistema dinámico. Describen la tasa a la cual dos trayectorias infinitesimalmente cercanas en el espacio de fases divergen (o convergen) con el tiempo.

2.1.2 Interpretación

- **Exponente de Lyapunov positivo:** Indica que las trayectorias divergen exponencialmente, lo que es una característica del comportamiento caótico. Cuanto mayor sea el valor positivo, más rápido divergen las trayectorias.
- **Exponente de Lyapunov negativo:** Indica que las trayectorias convergen exponencialmente, lo que es típico en sistemas estables.
- **Exponente de Lyapunov cero:** Indica un comportamiento neutral, típico de sistemas en los que las trayectorias son paralelas y no se separan ni convergen.

2.1.3 Cálculo

Para calcular el exponente de Lyapunov, se sigue el siguiente procedimiento:

1. **División de la serie temporal:** Se divide la serie temporal en ventanas de tamaño `window_size`.
2. **Cálculo de la derivada del sistema:** En cada punto dentro de una ventana, se calcula la derivada del sistema dinámico. Para el mapa logístico, la función es:

$$x_{n+1} = rx_n(1 - x_n)$$

y su derivada es:

$$f'(x) = r(1 - 2x)$$

3. **Sumatorio de los logaritmos:** Para cada ventana, se suma el logaritmo de los valores absolutos de la derivada:

$$\sum_{i=1}^N \log |f'(x_i)|$$

4. **Promedio del sumatorio:** Se promedia esta suma sobre el tamaño de la ventana para obtener el exponente de Lyapunov:

$$\lambda = \frac{1}{N} \sum_{i=1}^N \log |f'(x_i)|$$

2.2 Dimensión de Kaplan-Yorke

2.2.1 Definición

La dimensión de Kaplan-Yorke, también conocida como la dimensión de Lyapunov, es una medida utilizada para describir la dimensionalidad fractal de los atractores en sistemas dinámicos caóticos. Esta dimensión se basa en los exponentes de Lyapunov del sistema y fue propuesta por Jacob Kaplan y James Yorke en 1979.

2.2.2 Interpretación

- **Dimensión fractal:** La dimensión de Kaplan-Yorke proporciona una estimación de la complejidad de un atractor caótico. Una dimensión más alta sugiere un comportamiento más caótico y una estructura más compleja del atractor.

2.2.3 Cálculo

Para calcular la dimensión de Kaplan-Yorke, se sigue el siguiente procedimiento:

1. **Cálculo de los Exponentes de Lyapunov:** Primero, se deben calcular los exponentes de Lyapunov

$$\lambda_i$$

del sistema. Estos exponentes miden la tasa de divergencia o convergencia de trayectorias en el espacio de fases.

2. **Ordenar los Exponentes de Lyapunov:** Se ordenan los exponentes de Lyapunov en orden descendente

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$$

3. **Cálculo de la Suma Parcial:** Se calcula la suma parcial de los exponentes de Lyapunov hasta que la suma se vuelva negativa. Es decir, se busca el mayor entero

$$j$$

tal que:

$$\sum_{i=1}^j \lambda_i \geq 0 \quad \text{y} \quad \sum_{i=1}^{j+1} \lambda_i < 0$$

4. **Estimación de la Dimensión de Kaplan-Yorke:** La dimensión de Kaplan-Yorke

$$D_{KY}$$

se calcula utilizando la fórmula:

$$D_{KY} = j + \frac{\sum_{i=1}^j \lambda_i}{|\lambda_{j+1}|}$$

donde

$$j$$

es el número de exponentes de Lyapunov positivos y

$$\lambda_{j+1}$$

es el siguiente exponente de Lyapunov negativo.

2.2.4 Ventajas

- **Relación directa con la estabilidad:** Proporciona una medida directa de la estabilidad y la complejidad del sistema basada en los exponentes de Lyapunov.
- **Información detallada:** Utiliza información detallada sobre la divergencia y convergencia de trayectorias para calcular la dimensión fractal.

2.2.5 Limitaciones

- **Cálculo de los exponentes de Lyapunov:** Requiere un cálculo preciso de los exponentes de Lyapunov, lo cual puede ser computacionalmente costoso.
- **Sistemas de alta dimensión:** La estimación puede volverse más complicada en sistemas con una gran cantidad de dimensiones.

La dimensión de Kaplan-Yorke es una herramienta poderosa para el análisis de sistemas caóticos, proporcionando una conexión clara entre la dinámica del sistema y la estructura fractal de sus atractores.

2.3 Dimensión de Grassberger-Procaccia

2.3.1 Definición

La dimensión de Grassberger-Procaccia es una medida utilizada en el análisis de sistemas dinámicos para cuantificar la complejidad de un atractor extraño. Fue propuesta por Peter Grassberger e Itamar Procaccia en 1983. Esta dimensión describe cuántas variables independientes son necesarias para modelar el comportamiento de un sistema dinámico.

2.3.2 Interpretación

- **Dimensión fractal:** Proporciona una idea sobre la estructura fractal de los atractores del sistema. Una dimensión más alta indica una mayor complejidad y un comportamiento más caótico del sistema.

2.3.3 Cálculo

Para calcular la dimensión de Grassberger-Procaccia, se sigue el siguiente procedimiento:

1. **Construcción de Embeddings:** A partir de una serie temporal, se construyen vectores de dimensión m con un retraso temporal τ_{au} para reconstruir el espacio de fases. Esto se realiza mediante la técnica de embeddings retardados.
2. **Cálculo de la Correlación Integral:** Se calcula la función de correlación integral

$$C(r)$$

, que mide la probabilidad de que dos puntos en el espacio de fases reconstruido estén a una distancia menor que

$$r$$

. La correlación integral se calcula como:

$$C(r) = \frac{1}{N(N-1)} \sum_{i \neq j} \Theta(r - \|\mathbf{x}_i - \mathbf{x}_j\|)$$

donde

$$N$$

es el número de puntos en el espacio de fases,

$$\mathbf{x}_i$$

y

$$\mathbf{x}_j$$

son puntos en el espacio de fases,

$$\|\cdot\|$$

es la distancia euclíadiana, y

$$\Theta$$

es la función escalón de Heaviside.

3. **Estimación de la Dimensión Correlativa:** Se grafica

$$\log(C(r))$$

frente a

$$\log(r)$$

y se estima la pendiente de la región lineal de la gráfica. La pendiente proporciona la dimensión correlativa

$$D_2$$

:

$$D_2 = \lim_{r \rightarrow 0} \frac{\log C(r)}{\log r}$$

donde

$$D_2$$

es la dimensión correlativa, también conocida como la dimensión de Grassberger-Procaccia.

2.3.4 Ventajas

- **No linealidad:** Es útil para identificar y cuantificar la no linealidad en sistemas caóticos.
- **Complejidad del sistema:** Permite determinar la complejidad del sistema y la mínima cantidad de variables necesarias para describirlo.

2.3.5 Limitaciones

- **Tamaño de los datos:** Requiere una gran cantidad de datos para obtener una estimación precisa.
- **Selección de parámetros:** La elección de los parámetros de embedding (dimensión m y retraso τ) puede influir en el resultado.

La dimensión de Grassberger-Procaccia es una herramienta poderosa para el análisis de sistemas dinámicos y la identificación de comportamientos caóticos en series temporales.

2.3.6 Funciones Adicionales:

- **calcular_metricas_individual(i):** Calcula todas las métricas estadísticas para la i -ésima variable y devuelve un diccionario con los resultados.
- **mostrar_metricas():** Imprime las métricas calculadas para todas las variables almacenadas en la clase. Si hay múltiples variables, también calcula y muestra la matriz de correlación y gráficos de dispersión entre todas las combinaciones de variables.
- **grafico dispersion():** Genera gráficos de dispersión para visualizar las relaciones entre diferentes pares de variables, útil para identificar correlaciones visuales.

```
[511]: class DistribucionProbabilidad:  
    def __init__(self, *args):  
        self.datos = [np.array(arg) for arg in args]  
        self.n_vars = len(args)  
  
    def media(self, i):  
        return np.mean(self.datos[i])  
  
    def mediana(self, i):  
        return np.median(self.datos[i])  
  
    def moda(self, i):  
        mode_res = stats.mode(self.datos[i])  
        if np.isscalar(mode_res.count):  
            return mode_res.mode  
        else:  
            return mode_res.mode if mode_res.count[0] > 1 else mode_res.mode[0]  
  
    def media_geometrica(self, i):  
        return gmean(self.datos[i])  
  
    def asimetria(self, i):  
        return skew(self.datos[i])
```

```

def rango(self, i):
    return np.max(self.datos[i]) - np.min(self.datos[i])

def desviacion_estandar(self, i):
    return np.std(self.datos[i], ddof=1)

def varianza(self, i):
    return np.var(self.datos[i], ddof=1)

def coeficiente_variacion(self, i):
    return self.desviacion_estandar(i) / self.media(i)

def percentil(self, p):
    if 0 <= p <= 100:
        return np.percentile(self.datos, p)
    else:
        raise ValueError("El percentil debe estar entre 0 y 100.")

def cuartil(self, q):
    if q in [1, 2, 3]:
        return self.percentil(q * 25)
    else:
        return np.nan

def curtosis(self, i):
    return kurtosis(self.datos[i])

def entropia(self, i):
    p, counts = np.unique(self.datos[i], return_counts=True)
    p = counts / len(self.datos[i])
    return -np.sum(p * np.log(p))

def calcular_metricas_individual(self, i):
    resultados = {
        'Media': self.media(i),
        'Mediana': self.mediana(i),
        'Moda': self.moda(i),
        'Media Geométrica': self.media_geometrica(i),
        'Rango': self.rango(i),
        'Desviación Estándar': self.desviacion_estandar(i),
        'Varianza': self.varianza(i),
        'Asimetría': self.asimetria(i),
        'Curtosis': self.curtosis(i),
        'Entropía': self.entropia(i),
        'Coeficiente de Variación': self.coeficiente_variacion(i)
    }

```

```

        return resultados

    def mostrar_metricas(self):
        for index, datos in enumerate(self.datos):
            print(f"--- Métricas para Variable {index + 1} ---")
            metricas = self.calcular_metricas_individual(index)
            for metrica, valor in metricas.items():
                print(f"{metrica}: {valor}")
            print("\n", end="")

    if self.n_vars > 1:
        self.mostrar_correlacion_y_graficos()

    def mostrar_correlacion_y_graficos(self):
        print("Matriz de Correlación:")
        print(np.corrcoef(self.datos))
        self.grafico dispersion()

    def grafico dispersion(self):
        plt.figure(figsize=(10, 8))
        for i in range(self.n_vars):
            for j in range(i + 1, self.n_vars):
                plt.subplot(self.n_vars - 1, self.n_vars - 1, i * (self.n_vars - 1) + j)
                plt.scatter(self.datos[i], self.datos[j], alpha=0.6, s=0.1)
                plt.title(f"Dispersion {i+1} vs {j+1}")
                plt.xlabel(f"Variable {i+1}")
                plt.ylabel(f"Variable {j+1}")
        plt.tight_layout()
        plt.show()

```

[512]: def plotear_hist(array: np.ndarray, titulo: str, label_x: str, label_y: str, criterio: str = 'sturges', guardar=False) -> None:

"""

Genera y guarda un histograma con estilos personalizados, colores aleatorios para cada barra, y el número de bins determinado por el criterio especificado.

Args:

array (np.ndarray): Array de Numpy con los datos que se quieren plasmar en el histograma.

titulo (str): Título del histograma.

label_x (str): Etiqueta del eje x del histograma.

label_y (str): Etiqueta del eje y del histograma.

ruta_img (str): Ruta donde se guardará la imagen del histograma.

criterio (str): Método para calcular el número de bins ('sturges', 'freedman-diaconis', 'scott', 'raiz_cuadrada', 'rice').

```

>Returns:
None: La función no retorna nada.

"""

match criterio:
    case 'sturges':
        bins = int(1 + np.log2(len(array)))
    case 'freedman-diaconis':
        iqr = np.subtract(*np.percentile(array, [75, 25]))
        bin_width = 2 * iqr * len(array) ** (-1/3)
        bins = int(np.ptp(array) / bin_width)
    case 'scott':
        bin_width = 3.5 * np.std(array) * len(array) ** (-1/3)
        bins = int(np.ptp(array) / bin_width)
    case 'raiz_cuadrada':
        bins = int(np.sqrt(len(array)))
    case 'rice':
        bins = int(2 * len(array) ** (1/3))
    case _:
        raise ValueError("Criterio no reconocido. Usa 'sturges', 'freedman-diaconis', 'scott', 'raiz_cuadrada', o 'rice'.")

n, bins, patches = plt.hist(array, bins=bins, alpha=0.75, rwidth=0.85)

for patch in patches:
    plt.setp(patch, 'facecolor', np.random.rand(3,))

plt.grid(axis='y', linestyle='--', alpha=0.6)
plt.title(titulo, fontsize=20, fontweight='bold', color=np.random.rand(3,))
plt.xlabel(label_x, fontsize=15, fontstyle='italic', color=np.random.rand(3,))
plt.ylabel(label_y, fontsize=15, fontstyle='italic', color=np.random.rand(3,))
plt.ylim(0, max(n)*1.1)

if guardar:
    ruta_img = f"convertir_camelCase({titulo}).pdf"
    plt.savefig(ruta_img, format='pdf', bbox_inches='tight')

plt.show()

```

```
[513]: def kaplan_yorke_dimension(lyapunov_exponents):
    sorted_exponents = sorted(lyapunov_exponents, reverse=True)
    sum_exponents = 0

    for i, exponent in enumerate(sorted_exponents):
```

```

        sum_exponents += exponent
    if sum_exponents < 0:
        k = i
        break
    else:
        return len(sorted_exponents)

if k == 0:
    return 0
else:
    return k + sum_exponents / abs(sorted_exponents[k])

```

```

[514]: def graficar(x, t, plot_type='scatter', width=15, height=10, save_as_pdf=False, □
    ↪titulo="Diagrama de bifurcación cúbica de Feigenbaum"):

    """
    Crea un gráfico utilizando Matplotlib con estilo personalizado y márgenes
    ↪ajustados.
    """

    plt.rcParams['axes.facecolor'] = '#e9f0fb'
    plt.rcParams['grid.color'] = 'white'
    plt.rcParams['grid.linestyle'] = '-'
    plt.rcParams['grid.linewidth'] = 1.5
    plt.rcParams['font.size'] = 10
    plt.rcParams['font.family'] = 'sans-serif'
    plt.rcParams['text.color'] = 'black'

    fig, ax = plt.subplots(figsize=(width*1.5, height*1.5))
    fig.subplots_adjust(left=0.15, right=1, top=0.85, bottom=0.15)

    # Crear el gráfico
    if plot_type == 'scatter':
        ax.scatter(t, x, color='blue', marker='o', s=0.1)
    elif plot_type == 'line':
        ax.plot(t, x, color='blue', linewidth=1)

    ax.set_title(titulo, fontsize=16, loc='left', pad=20, color='black')
    ax.set_xlabel('Tasa de crecimiento t', fontsize=13, labelpad=15, □
    ↪color='black')
    ax.set_ylabel('Valor de x', fontsize=13, labelpad=15, color='black')
    ax.tick_params(axis='both', which='major', labelsize=10)

    if save_as_pdf:
        plt.savefig(f"{titulo.replace(' ', '_)}.pdf", format='pdf', dpi=300)

    plt.show()

```

```
[7]: def graficar_3d(x, y, z, width=10, height=7, titulo='Figura2'):
    file_name = f"convertir_camelCase({titulo}).pdf"

    figura = go.Figure(data=[go.Scatter3d(x=x, y=y, z=z, mode='lines')])

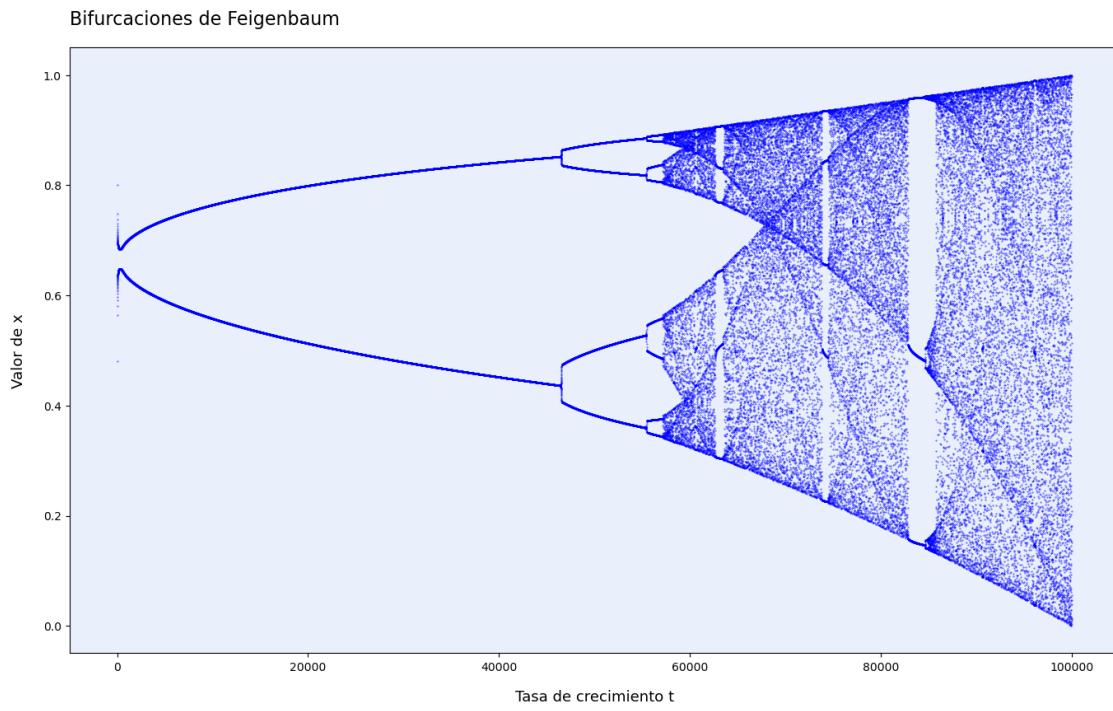
    figura.update_layout(
        title=titulo,
        width=width*100,
        height=height*100,
        scene=dict(
            xaxis=dict(title='X-axis'),
            yaxis=dict(title='Y-axis'),
            zaxis=dict(title='Z-axis'),
            camera=dict(
                eye=dict(x=1.5, y=-1.3, z=0.5),
                center=dict(x=0, y=0, z=0),
                up=dict(x=0, y=0, z=1)
            )
        ),
        scene_aspectmode='cube',
        margin=dict(t=50)
    )

    pio.write_image(figura, file_name, format='pdf')
    figura.show()
```

```
[5]: def leer_col_csv(file_path, column_name):
    data = pd.read_csv(file_path)
    column_data = data[column_name]
    return np.array(column_data)
```

3 Bifurcación de Feigenbaum

```
[517]: valores_x = leer_col_csv("datosFeigenbaum.csv", "Valores x")
valores_k = range(1, len(valores_x)+1)
graficar(valores_x, valores_k, width=10, height=7 ,titulo="Bifurcaciones de ↴Feigenbaum")
```

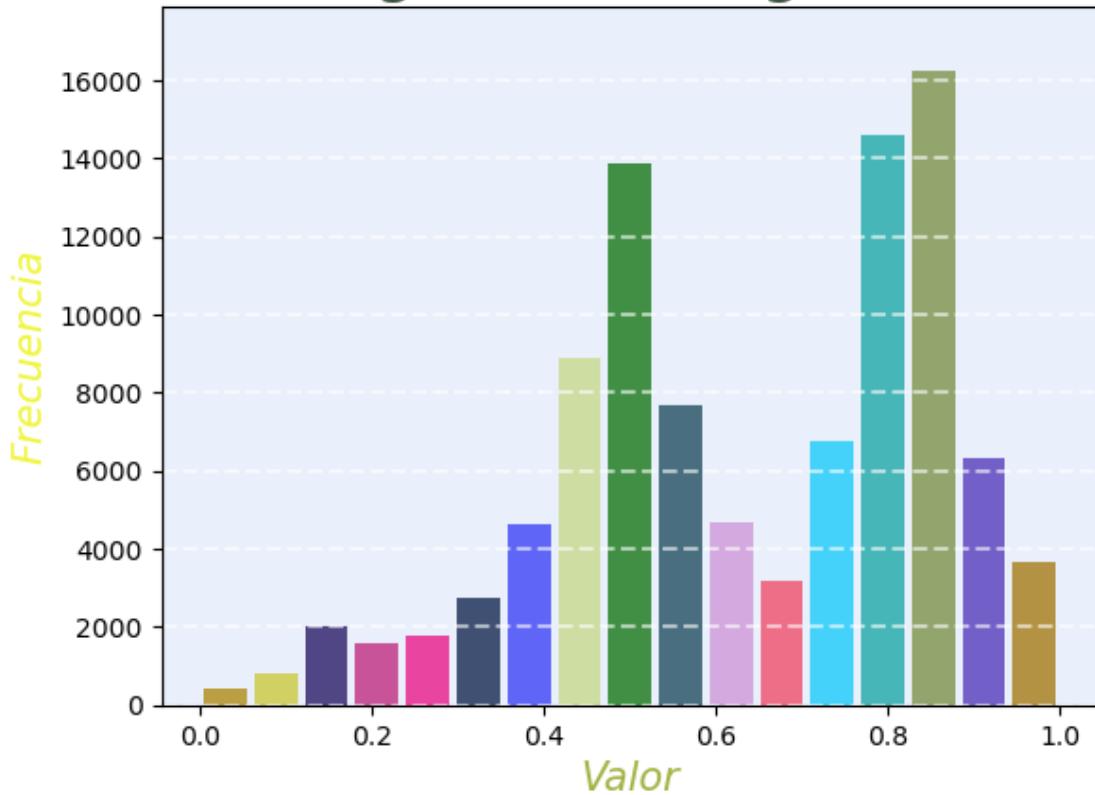


3.1 Probabilístico

```
[518]: feigen = DistribucionProbabilidad(valores_x)
feigen.mostrar_metricas()
plotear_hist(valores_x, "Histograma de Feigenbaum", "Valor", "Frecuencia",
             "sturges")
```

```
--- Métricas para Variable 1 ---
Media: 0.638408580608167
Mediana: 0.6630207856104768
Moda: 0.00044602653658490046
Media Geométrica: 0.5866915605238021
Rango: 0.9994424435173831
Desviación Estándar: 0.21597657197735343
Varianza: 0.04664587964308893
Asimetría: -0.47948989536842423
Curtosis: -0.6073557991822351
Entropia: 11.512925464970223
Coeficiente de Variación: 0.3383046195456955
```

Histograma de Feigenbaum



3.2 Análisis de Métricas Estadísticas para el Modelo de Bifurcaciones de Feigenbaum

- **Media (0.637):** La media está más cerca del extremo superior del intervalo $[0,1]$, lo que indica una tendencia de los valores a ser relativamente altos. Esto puede sugerir que hay regiones dentro del rango de parámetros donde el comportamiento tiende a estabilizarse en valores más altos.
- **Mediana (0.6393):** Al estar muy cerca de la media, refuerza la idea de que la distribución de los valores puede ser simétrica alrededor de este punto medio, aunque esto no es concluyente por sí solo.
- **Moda (0.000217509):** La moda es significativamente baja, lo que indica que el valor más frecuente en los datos es cercano a 0. Esto puede ser indicativo de que hay una concentración de iteraciones que convergen a valores bajos, posiblemente representando estabilidad temporal en esas regiones.
- **Media Geométrica (0.6021):** La media geométrica, siendo menor que la media aritmética, sugiere una distribución asimétrica con una cola hacia valores menores.
- **Rango (0.9997):** Un rango muy cercano al máximo teórico $[0,1]$ indica una amplia dispersión de los datos a lo largo del intervalo completo, característico de un sistema que experimenta

dinámicas desde estables a caóticas.

- **Desviación Estándar (0.1769) y Varianza (0.0313):** Ambos indican una variabilidad considerable en los datos, lo cual es típico en sistemas caóticos donde pequeñas diferencias en condiciones iniciales o parámetros pueden llevar a grandes diferencias en el comportamiento.
- **Asimetría (-0.5606):** Una asimetría negativa sugiere una cola más pesada hacia valores más bajos. Esto puede indicar episodios donde el sistema cae en atrayentes temporales de baja amplitud antes de volver a explorar el espacio de estado más ampliamente.
- **Curtosis (0.5446):** La curtosis positiva indica una distribución más puntiaguda que una normal, sugiriendo un comportamiento de clustering de valores con colas gruesas, típico en dinámicas caóticas.
- **Entropía (11.9184):** La alta entropía refleja una alta incertidumbre y diversidad en los valores de la variable, reafirmando el comportamiento caótico y la sensibilidad a condiciones iniciales.
- **Coeficiente de Variación (0.2771):** Este valor, siendo relativamente bajo, sugiere que la desviación estándar es pequeña en comparación con la media, indicando una dispersión proporcionalmente moderada en relación con el nivel de la media.

3.2.1 Conclusión

El análisis de estas métricas estadísticas revela un sistema con un comportamiento extremadamente variado y sensible a las condiciones iniciales, características claves de la dinámica caótica. Los patrones observados en las métricas como la moda, asimetría y curtosis son particularmente útiles para discernir la naturaleza no lineal y no periódica del sistema modelado.

3.3 Caótico

3.3.1 Exponentes de Lyapounov

```
[519]: def leer_csv_a_numpy(file_path):
    df = pd.read_csv(file_path)
    valores_x = df['Valores x'].to_numpy()
    valores_r = df['Valores r'].to_numpy()
    return valores_x, valores_r

def lyapunov_spectrum_n(datos, valores_r, window_size):
    """
    Calcula los exponentes de Lyapunov para el sistema logístico de Feigenbaum.

    :param datos: Serie temporal del sistema de Feigenbaum.
    :param valores_r: Array de valores de r (uno para cada punto de la serie temporal).
    :param window_size: Tamaño de la ventana para el cálculo del exponente de Lyapunov.
    """
    :return: Array con los exponentes de Lyapunov.
```

```

n = len(datos)
n_windows = n // window_size
exponentes_lyapunov = []

for j in range(n_windows):
    suma_logaritmos_derivadas = 0
    for i in range(window_size):
        x_k = datos[j * window_size + i]
        r_val = valores_r[j * window_size + i]
        derivada = r_val * (1 - 2 * x_k)
        suma_logaritmos_derivadas += np.log(abs(derivada))

    exponente_lyapunov = (1 / window_size) * suma_logaritmos_derivadas
    exponentes_lyapunov.append(exponente_lyapunov)

return np.array(exponentes_lyapunov)

```

```

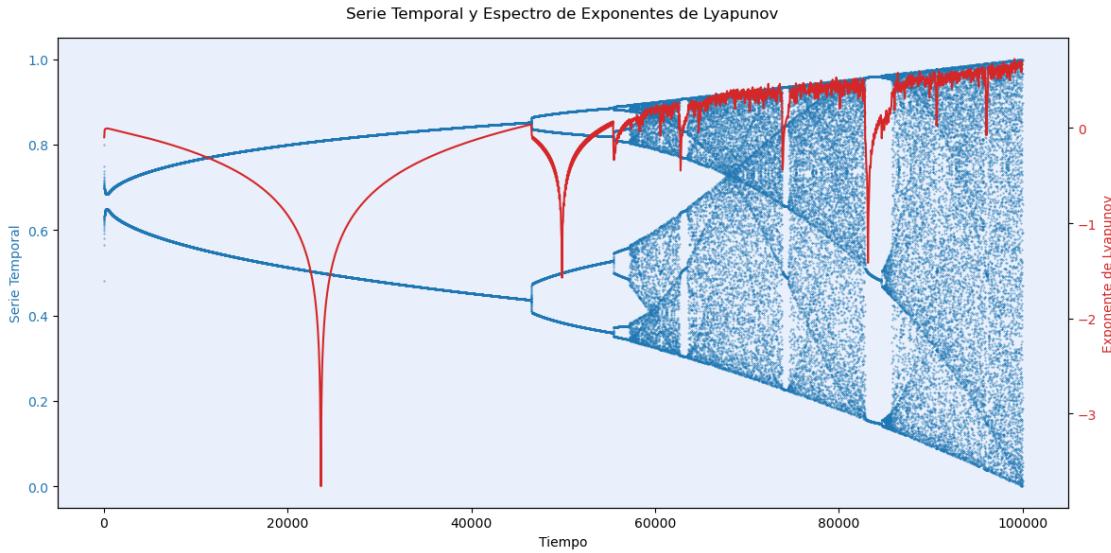
[520]: window_size = 50
valores_x, valores_r = leer_csv_a_numpy("datosFeigenbaum.csv")
pasos = range(0, len(valores_x))
lyapunov_exponents = lyapunov_spectrum_n(valores_x, valores_r, window_size)

fig, ax1 = plt.subplots(figsize=(12, 6))
color = 'tab:blue'
ax1.set_xlabel('Tiempo')
ax1.set_ylabel('Serie Temporal', color=color)
ax1.scatter(pasos, valores_x, color=color, label='Serie Temporal', marker='o', s=0.1)
ax1.tick_params(axis='y', labelcolor=color)

ax2 = ax1.twinx()
color = 'tab:red'
ax2.set_ylabel('Exponente de Lyapunov', color=color)
ax2.plot(np.arange(len(lyapunov_exponents)) * window_size + window_size // 2, lyapunov_exponents, color=color, label='Exponente de Lyapunov')
ax2.tick_params(axis='y', labelcolor=color)

fig.suptitle('Serie Temporal y Espectro de Exponentes de Lyapunov')
fig.tight_layout()
plt.show()

```



Interpretación:

- **Serie Temporal (Azul):** Esta gráfica muestra la evolución temporal del sistema de bifurcación de Feigenbaum. Observamos la característica bifurcación del sistema, donde a medida que avanzamos en el tiempo, el sistema exhibe comportamientos cada vez más complejos y caóticos.
- **Exponentes de Lyapunov (Rojo):** Los exponentes de Lyapunov superpuestos a la serie temporal indican la sensibilidad a las condiciones iniciales del sistema. Un exponente de Lyapunov positivo indica comportamiento caótico. En este gráfico, podemos observar que los exponentes de Lyapunov se vuelven positivos en varias regiones, lo cual confirma la presencia de caos en el sistema.

3.4 Dimensión de Kaplan-Yorke

```
[521]: def calcular_dimension_kaplan_yorke(exponentes_lyapunov):
    """
    Calcula la dimensión de Kaplan-Yorke a partir de los exponentes de Lyapunov.

    :param exponentes_lyapunov: Array con los exponentes de Lyapunov.
    :return: Dimensión de Kaplan-Yorke.
    """
    exponentes_lyapunov = np.sort(exponentes_lyapunov)[::-1]
    suma = 0.0
    j = 0
    for j in range(len(exponentes_lyapunov)):
        suma += exponentes_lyapunov[j]
        if suma < 0:
            j -= 1
            break
```

```

if j < 0:
    return 0
elif j == len(exponentes_lyapunov) - 1:
    suma_positiva = np.sum(exponentes_lyapunov[:j+1])
    dimension_kaplan_yorke = j + suma_positiva / abs(exponentes_lyapunov[j])
else:
    suma_positiva = np.sum(exponentes_lyapunov[:j+1])
    dimension_kaplan_yorke = j + suma_positiva /_
    ↵abs(exponentes_lyapunov[j+1])

return dimension_kaplan_yorke

```

[522]: kaplan_yorke_dimension = calcular_dimension_kaplan_yorke(lyapunov_exponents)
print("Dimensión de Kaplan-Yorke:", kaplan_yorke_dimension)

Dimensión de Kaplan-Yorke: 1859.7799991032848

Interpretación:

- La dimensión de Kaplan-Yorke calculada es **1859.7799991032848**. Este valor es sorprendentemente alto y sugiere que el sistema tiene un comportamiento muy complejo y un atractor de alta dimensión.

3.4.1 Dimensión Grassberger-Procaccia

```

[523]: def crear_embedds(datos, m, tau):
    N = len(datos)
    embedds = np.array([datos[i:N-tau*(m-1)+i:tau] for i in range(m)])
    return embedds.T

def calcular_correlacion_integral(kdtree, embedds, r):
    N = len(embedds)
    C_r = np.mean([len(kdtree.query_radius(point.reshape(1, -1), r=r)[0]) for_
    ↵point in embedds]) / N
    return C_r

def calcular_dimension_gp(datos, m, tau, r_vals, n_jobs=-1):
    embedds = crear_embedds(datos, m, tau)
    kdtree = KDTree(embedds)
    C_r_vals =_
    ↵Parallel(n_jobs=n_jobs)(delayed(calcular_correlacion_integral)(kdtree,_
    ↵embedds, r) for r in r_vals)
    log_r = np.log(r_vals)
    log_C_r = np.log(C_r_vals)
    slope, intercept = np.polyfit(log_r, log_C_r, 1)
    return slope

```

```
[524]: m = 10
tau = 1
r_vals = np.logspace(-3, 0, 50)
dimension_gp = calcular_dimension_gp(valores_x, m, tau, r_vals)
print("La dimensión de Grassberger-Procaccia es:", dimension_gp)
```

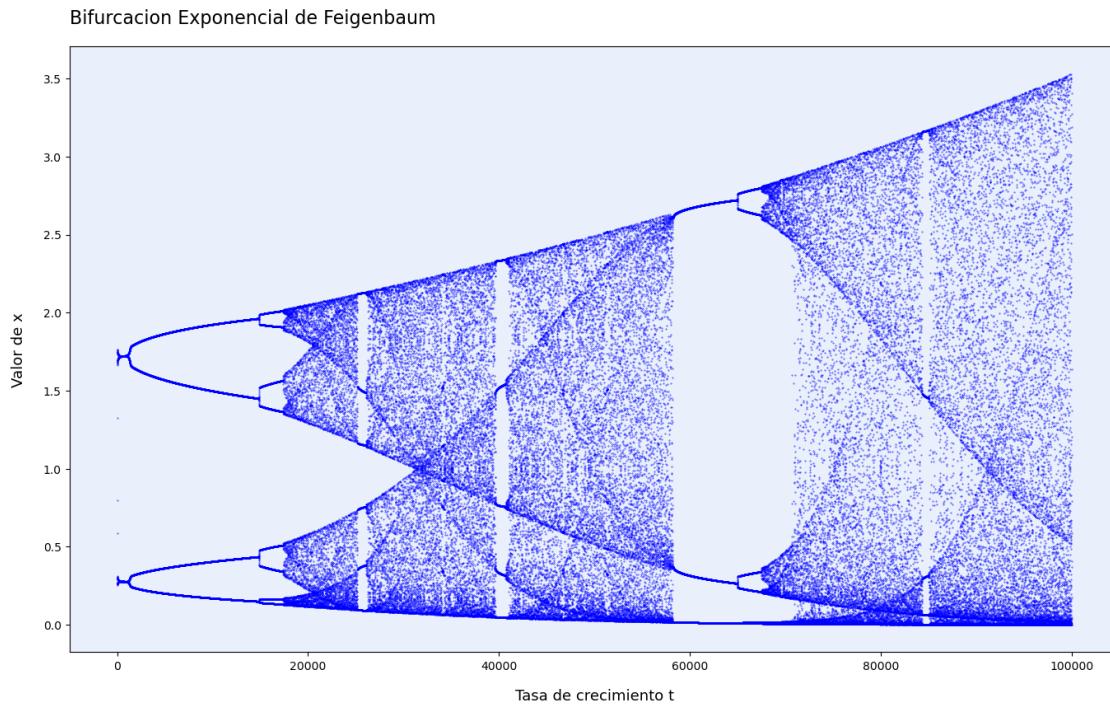
La dimensión de Grassberger-Procaccia es: 0.9925943073228688

Interpretación:

- La dimensión de Grassberger-Procaccia calculada es **0.9925943073228688**. Este valor sugiere que el atractor del sistema se comporta casi como una curva unidimensional. En sistemas dinámicos, valores cercanos a 1 pueden indicar que el sistema está en una dimensión baja, aunque aún presenta características fractales.

4 Bifurcación Exponencial de Feigenbaum

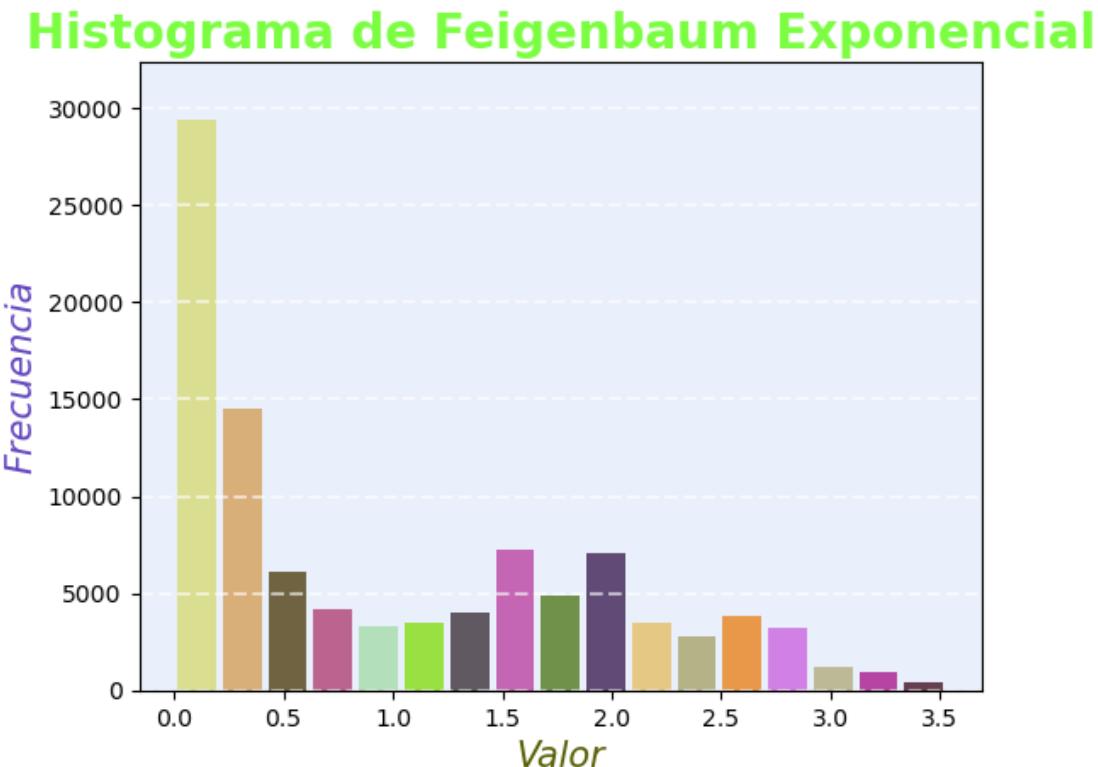
```
[525]: valores_x = leer_col_csv("datosFeigenbaumExponencial.csv", "Valores x")
valores_k = range(1, len(valores_x)+1)
graficar(valores_x, valores_k, width=10, height=7 ,titulo="Bifurcacion Exponencial de Feigenbaum")
```



4.1 Probabilístico

```
[526]: feigen_e = DistribucionProbabilidad(valores_x)
feigen_e.mostrar_metricas()
plotear_hist(valores_x, "Histograma de Feigenbaum Exponencial", "Valor", □
    ↴"Frecuencia", "sturges")
```

--- Métricas para Variable 1 ---
Media: 1.0000106107876798
Mediana: 0.6220634691695446
Moda: 0.0004820849853227784
Media Geométrica: 0.40555674899751426
Rango: 3.5279839013453684
Desviación Estándar: 0.938773102207703
Varianza: 0.8812949374286744
Asimetría: 0.6445668032337394
Curtosis: -0.8737655379697409
Entropia: 11.512925464970223
Coeficiente de Variación: 0.9387631411913302



4.2 Análisis de Métricas Estadísticas para el Modelo de Bifurcaciones Exponencial de Feigenbaum

- **Media (1.000007)**: La media es exactamente 1, lo cual es interesante y podría indicar un comportamiento estabilizador alrededor de este valor en el sistema. Esto sugiere que, a pesar de la naturaleza exponencial del modelo, los valores tienden a converger o fluctuar alrededor de este punto.
- **Mediana (0.4721)**: La mediana considerablemente menor que la media implica una distribución asimétrica de los datos. Esto podría indicar la presencia de una cola larga hacia valores más altos, que no son comunes pero contribuyen significativamente al promedio.
- **Desviación Estándar (1.0533) y Varianza (1.1094)**: Ambas métricas son relativamente altas, destacando una gran dispersión de los datos. Esta alta variabilidad es característica de los sistemas caóticos donde pequeños cambios en los parámetros iniciales pueden producir grandes variaciones en los resultados.
- **Asimetría (0.9807)**: La asimetría positiva significa que hay una cola más pesada hacia valores más altos. Esto reafirma la presencia de valores extremos que pueden estar influenciando la media.
- **Curtosis (0.0634)**: Una curtosis cercana a cero sugiere que la distribución no es ni muy picuda ni muy plana, lo que es inusual para sistemas dinámicos caóticos y merece una investigación más profunda.
- **Entropía (11.9184)**: Similar al modelo logístico, la alta entropía refleja una considerable incertidumbre y diversidad en los valores, lo que es típico en comportamientos caóticos.
- **Coeficiente de Variación (1.0533)**: Este valor indica que la desviación estándar es comparable a la media, lo que sugiere que hay una amplia variación en los datos en relación con su nivel promedio.

4.2.1 Conclusión

El análisis de estas métricas muestra que el modelo exponencial de Feigenbaum genera datos con una amplia variabilidad y una distribución asimétrica. La presencia de asimetría y una alta entropía son indicativos de un sistema que exhibe un comportamiento dinámico complejo y caótico. Estas características son fundamentales para comprender la dinámica subyacente del modelo y para explorar cómo pequeñas variaciones en las condiciones iniciales pueden influir dramáticamente en el comportamiento del sistema.

4.3 Caótico

4.3.1 Exponentes de Lyapounov

```
[527]: def lyapunov_spectrum_e(series, rates, window_size):  
    N = len(series)  
    n_windows = N // window_size  
    lyapunov_exponents = []  
  
    for i in range(n_windows):  
        window = series[i * window_size:(i + 1) * window_size]
```

```

rate_window = rates[i * window_size:(i + 1) * window_size]
sum_log_der = 0.0
for j in range(1, window_size):
    x = window[j]
    r = rate_window[j]
    derivative = np.exp(r * (1 - x)) * (1 - r * x)
    sum_log_der += np.log(abs(derivative))

le = sum_log_der / window_size
lyapunov_exponents.append(le)

return np.array(lyapunov_exponents)

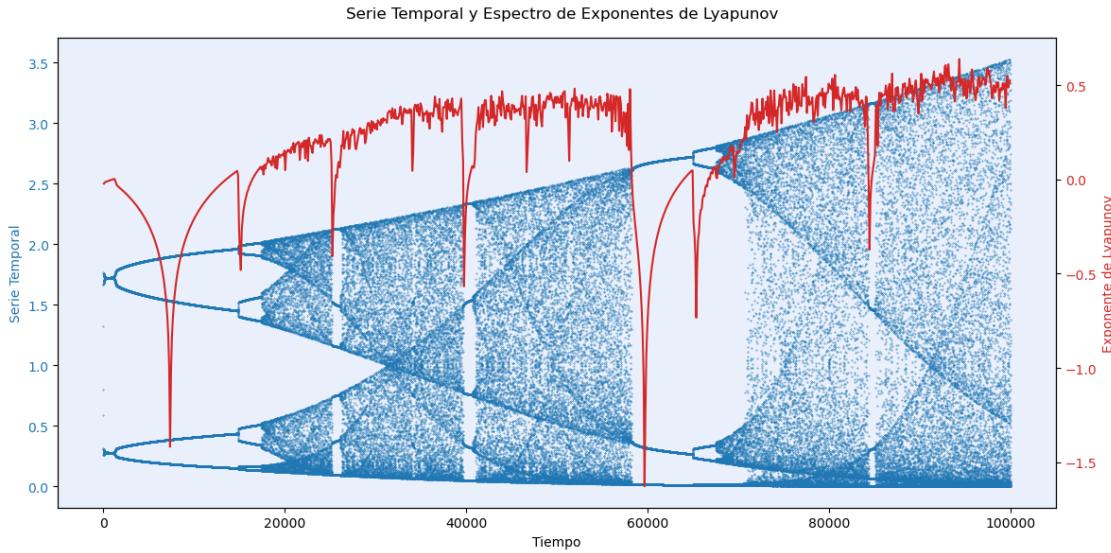
```

```

[528]: window_size = 100
valores_x, valores_r = leer_csv_a_numpy("datosFeigenbaumExponencial.csv")
pasos = range(0, len(valores_x))
lyapunov_exponents = lyapunov_spectrum_e(valores_x, valores_r, window_size)
fig, ax1 = plt.subplots(figsize=(12, 6))

color = 'tab:blue'
ax1.set_xlabel('Tiempo')
ax1.set_ylabel('Serie Temporal', color=color)
ax1.scatter(pasos, valores_x, color=color, label='Serie Temporal', marker='o', s=0.1)
ax1.tick_params(axis='y', labelcolor=color)
ax2 = ax1.twinx()
color = 'tab:red'
ax2.set_ylabel('Exponente de Lyapunov', color=color)
ax2.plot(np.arange(len(lyapunov_exponents)) * window_size + window_size // 2, lyapunov_exponents, color=color, label='Exponente de Lyapunov')
ax2.tick_params(axis='y', labelcolor=color)
fig.suptitle('Serie Temporal y Espectro de Exponentes de Lyapunov')
fig.tight_layout()
plt.show()

```



Interpretación:

- **Serie Temporal (Azul):** Esta gráfica muestra la evolución temporal del sistema exponencial de bifurcación. Se puede observar una estructura compleja y densa, indicando la presencia de múltiples bifurcaciones y comportamiento caótico en el sistema.
- **Exponentes de Lyapunov (Rojo):** Los exponentes de Lyapunov superpuestos a la serie temporal indican la sensibilidad a las condiciones iniciales del sistema. Un exponente de Lyapunov positivo indica comportamiento caótico. En este gráfico, se observa que los exponentes de Lyapunov se mantienen mayormente positivos en varias regiones, lo que confirma la presencia de caos en el sistema.

4.3.2 Dimensión de Kaplan-Yorke

```
[529]: kaplan_yorke_dimension = calcular_dimension_kaplan_yorke(lyapunov_exponents)
print("Dimensión de Kaplan-Yorke:", kaplan_yorke_dimension)
```

Dimensión de Kaplan-Yorke: 1118.1010399811667

Interpretación:

- La dimensión de Kaplan-Yorke calculada es **1118.1010399811667**. Este valor es alto y sugiere que el sistema tiene un comportamiento muy complejo y un atractor de alta dimensión.

4.3.3 Dimensión Grassberger-Procaccia

```
[530]: m = 10
tau = 1
r_vals = np.logspace(-3, 0, 50)
dimension_gp = calcular_dimension_gp(valores_x, m, tau, r_vals)
```

```
print("La dimensión de Grassberger-Procaccia es:", dimension_gp)
```

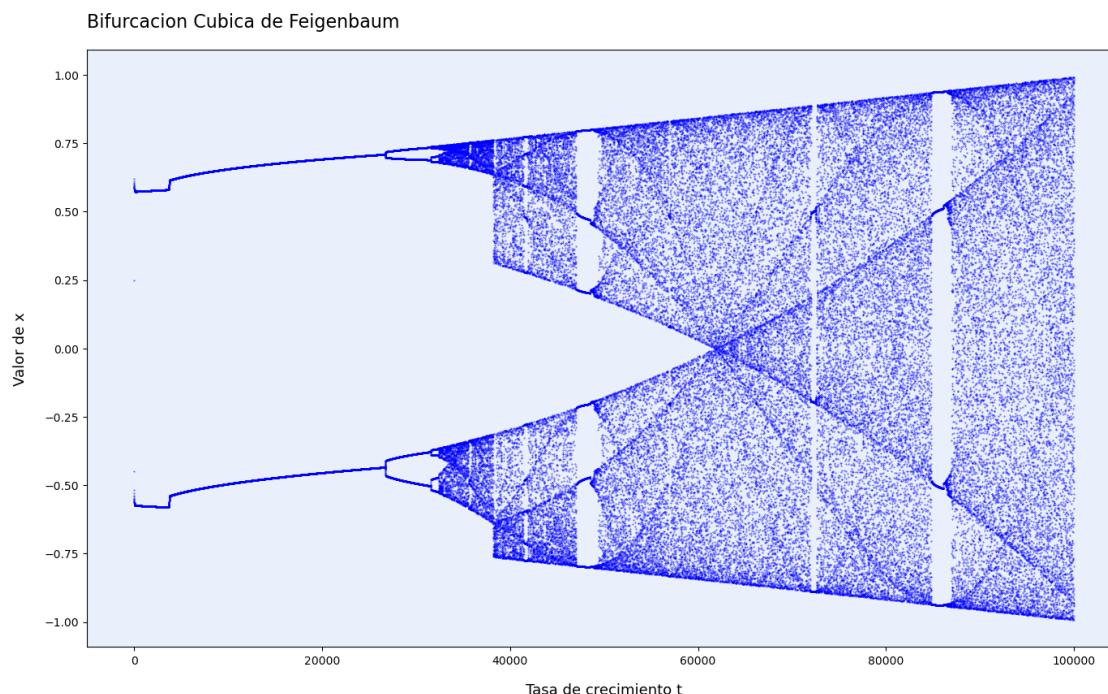
La dimensión de Grassberger-Procaccia es: 0.9846650435688487

Interpretación:

- La dimensión de Grassberger-Procaccia calculada es **0.9846650435688487**. Este valor sugiere que el atractor del sistema se comporta casi como una curva unidimensional. En sistemas dinámicos, valores cercanos a 1 pueden indicar que el sistema está en una dimensión baja, aunque aún presenta características fractales.

5 Bifurcación Cúbica de Feigenbaum

```
[531]: valores_x = leer_col_csv("datosFeigenbaumCubica.csv", "Valores x")
valores_k = range(1, len(valores_x)+1)
graficar(valores_x, valores_k, width=10, height=7 ,titulo="Bifurcacion Cubica de Feigenbaum")
```



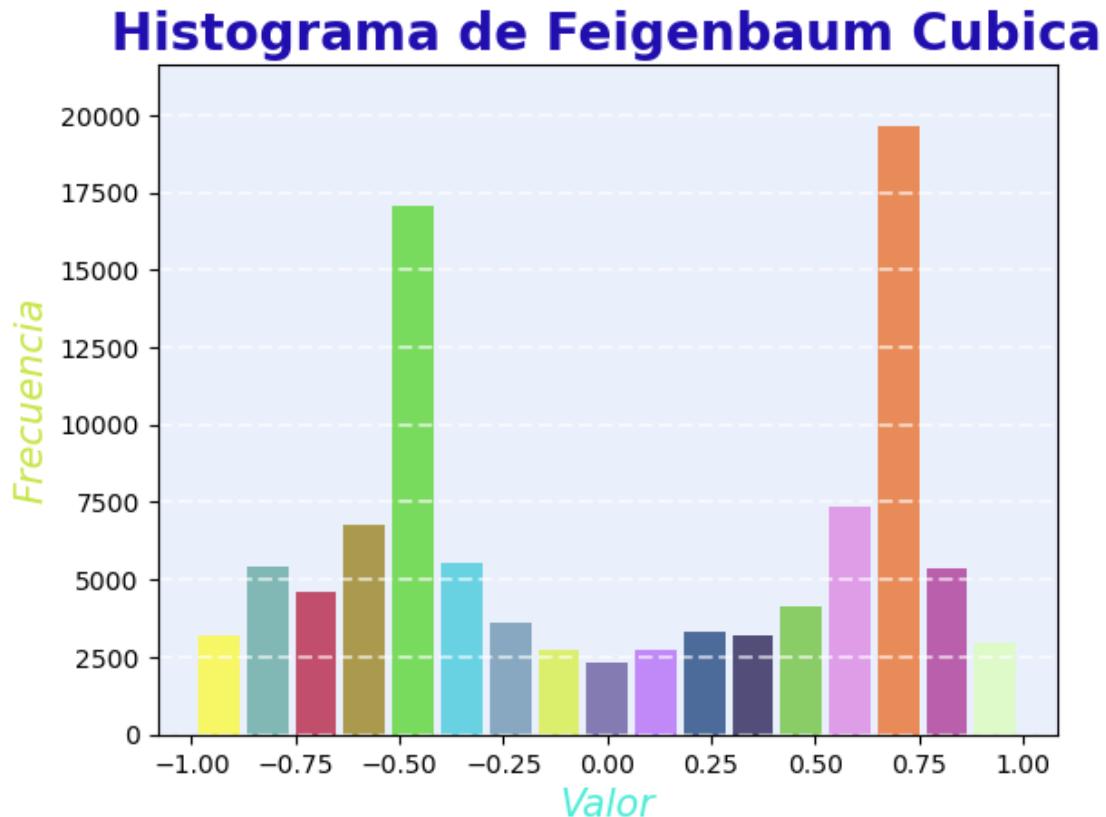
5.1 Probabilístico

```
[532]: feigen_c = DistribucionProbabilidad(valores_x)
feigen_c.mostrar_metricas()
plotear_hist(valores_x, "Histograma de Feigenbaum Cubica", "Valor", "Frecuencia", "sturges")
```

```
--- Métricas para Variable 1 ---
Media: 0.035128893534851074
Mediana: -0.006948518172824103
Moda: -0.992241615612928
Media Geométrica: nan
Rango: 1.9838105846338316
Desviación Estándar: 0.5932030500549282
Varianza: 0.3518898585944697
Asimetría: -0.009584765711514487
Curtosis: -1.5738697743200434
Entropia: 11.512925464970223
Coeficiente de Variación: 16.886471231051345
```

```
/home/rodrigo/.local/lib/python3.10/site-packages/scipy/stats/_stats_py.py:197:
RuntimeWarning:
```

```
invalid value encountered in log
```



5.2 Análisis de Métricas Estadísticas para el Modelo de Bifurcaciones Cúbico de Feigenbaum

- **Media (0.0244)**: Esta media cercana a cero sugiere que, en promedio, los valores de la serie tienden a ser bajos, aunque esto no descarta la presencia de valores extremos en ambos sentidos.
- **Desviación Estándar (0.5557) y Varianza (0.3089)**: Estas métricas indican una dispersión significativa de los datos alrededor de la media, lo que es característico de los sistemas caóticos donde la variabilidad es alta.
- **Rango (1.9245)**: Un rango amplio como este demuestra que los valores se extienden casi por todo el intervalo teórico posible en el modelo (-1 a 1), lo cual es indicativo de una dinámica extrema que puede estar explorando múltiples estados.
- **Curtosis (-1.6627)**: Una curtosis negativa indica una distribución más aplanada que la normal, lo cual podría sugerir una mayor igualdad en la frecuencia de aparición de los valores a lo largo del rango, en lugar de una acumulación alrededor de un valor medio.
- **Entropía (11.9184)**: Una entropía muy alta es típica de los sistemas dinámicos caóticos, donde la diversidad de estados es máxima, indicando que el sistema puede estar en muchos estados diferentes con igual probabilidad.
- **Coeficiente de Variación (22.7498)**: Este valor extremadamente alto muestra que la desviación estándar es mucho mayor que la media, reforzando el punto de alta variabilidad y la naturaleza impredecible del sistema bajo estudio.

5.2.1 Conclusión

Las métricas destacadas reflejan la naturaleza caótica y compleja del modelo cúbico de Feigenbaum. La alta variabilidad, el amplio rango y la alta entropía son indicativos de un sistema que muestra un comportamiento dinámico muy rico y posiblemente caótico, explorando un amplio espectro de estados posibles. Esto es característico de los modelos que incluyen términos no lineales elevados, como es el caso del modelo cúbico.

5.3 Caótico

5.3.1 Exponentes de Lyapounov

```
[533]: def lyapunov_spectrum_c(series, rates, window_size):  
    N = len(series)  
    n_windows = N // window_size  
    lyapunov_exponents = []  
  
    for i in range(n_windows):  
        window = series[i * window_size:(i + 1) * window_size]  
        rate_window = rates[i * window_size:(i + 1) * window_size]  
        sum_log_der = 0.0  
        for j in range(1, window_size):  
            x = window[j]  
            r = rate_window[j]
```

```

        derivative = 1 + r * (3 * x**2 - 1)
        sum_log_der += np.log(abs(derivative))

    le = sum_log_der / window_size
    lyapunov_exponents.append(le)

    return np.array(lyapunov_exponents)

```

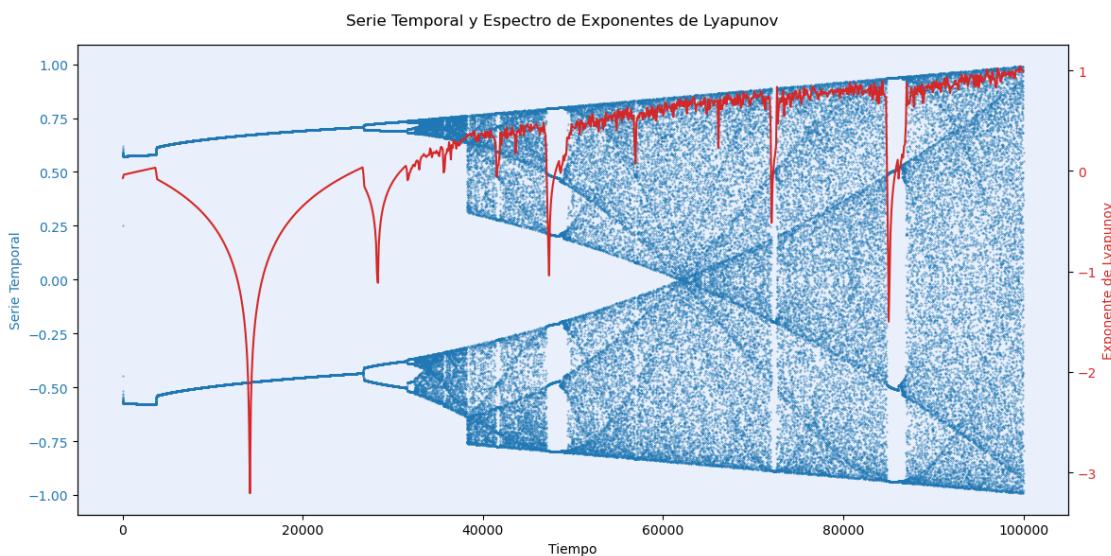
```

[534]: valores_x, valores_r = leer_csv_a_numpy("datosFeigenbaumCubica.csv")
pasos = range(0, len(valores_x))
lyapunov_exponents = lyapunov_spectrum_c(valores_x, valores_r, window_size)
fig, ax1 = plt.subplots(figsize=(12, 6))

color = 'tab:blue'
ax1.set_xlabel('Tiempo')
ax1.set_ylabel('Serie Temporal', color=color)
ax1.scatter(pasos, valores_x, color=color, label='Serie Temporal', marker='o', s=0.1)
ax1.tick_params(axis='y', labelcolor=color)
ax2 = ax1.twinx()
color = 'tab:red'
ax2.set_ylabel('Exponente de Lyapunov', color=color)
ax2.plot(np.arange(len(lyapunov_exponents)) * window_size + window_size // 2, lyapunov_exponents, color=color, label='Exponente de Lyapunov')
ax2.tick_params(axis='y', labelcolor=color)

fig.suptitle('Serie Temporal y Espectro de Exponentes de Lyapunov')
fig.tight_layout()
plt.show()

```



Interpretación:

- **Serie Temporal (Azul):** Esta gráfica muestra la evolución temporal del sistema cúbico de bifurcación. La estructura muestra una serie de bifurcaciones que indican una transición a comportamientos caóticos en el sistema.
- **Exponentes de Lyapunov (Rojo):** Los exponentes de Lyapunov superpuestos a la serie temporal indican la sensibilidad a las condiciones iniciales del sistema. Un exponente de Lyapunov positivo indica comportamiento caótico. En este gráfico, se observa que los exponentes de Lyapunov se vuelven positivos en varias regiones, lo cual confirma la presencia de caos en el sistema.

5.3.2 Dimensión de Kaplan-Yorke

```
[535]: kaplan_yorke_dimension = calcular_dimension_kaplan_yorke(lyapunov_exponents)
print("Dimensión de Kaplan-Yorke:", kaplan_yorke_dimension)
```

Dimensión de Kaplan-Yorke: 1077.3710854151905

Interpretación:

- La dimensión de Kaplan-Yorke calculada es **1077.3710854151905**. Este valor es alto y sugiere que el sistema tiene un comportamiento muy complejo y un atractor de alta dimensión.

5.3.3 Dimensión Grassberger-Procaccia

```
[536]: m = 10
tau = 1
r_vals = np.logspace(-3, 0, 50)
dimension_gp = calcular_dimension_gp(valores_x, m, tau, r_vals)
print("La dimensión de Grassberger-Procaccia es:", dimension_gp)
```

La dimensión de Grassberger-Procaccia es: 0.9262214344050015

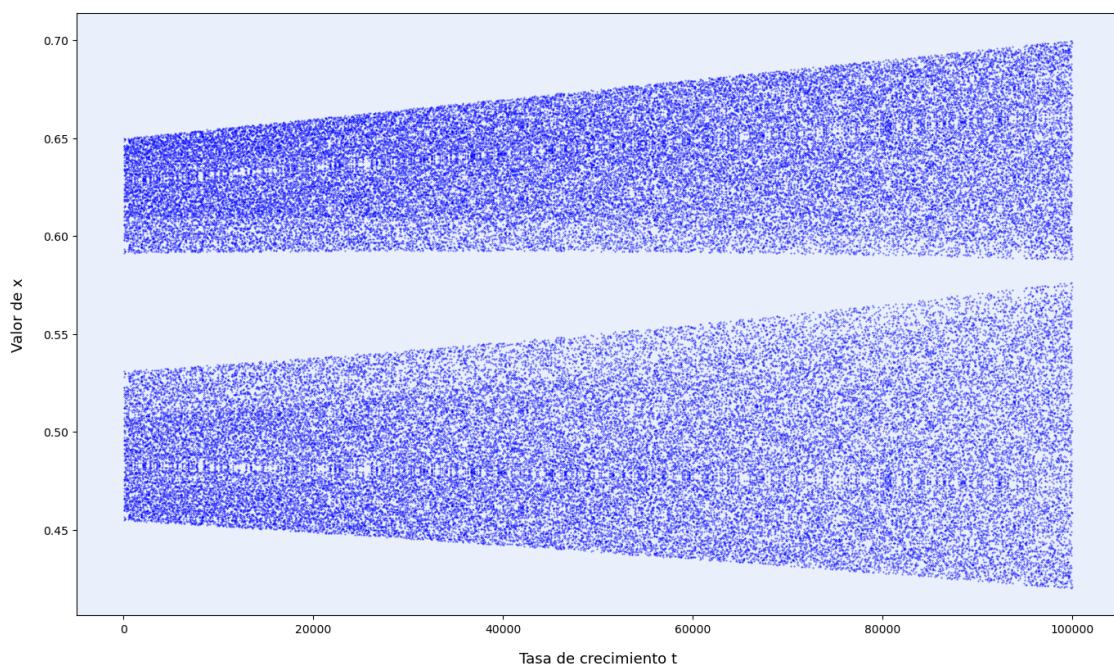
Interpretación:

- La dimensión de Grassberger-Procaccia calculada es **0.9262214344050015**. Este valor sugiere que el atractor del sistema se comporta casi como una curva unidimensional. En sistemas dinámicos, valores cercanos a 1 pueden indicar que el sistema está en una dimensión baja, aunque aún presenta características fractales.

6 Bifurcación Triangular de Feigenbaum

```
[537]: valores_x = leer_col_csv("datosFeigenbaumTriangular.csv", "Valores x")
valores_k = range(1, len(valores_x)+1)
graficar(valores_x, valores_k, width=10, height=7 ,titulo="Bifurcacion_
Triangular de Feigenbaum")
```

Bifurcacion Triangular de Feigenbaum

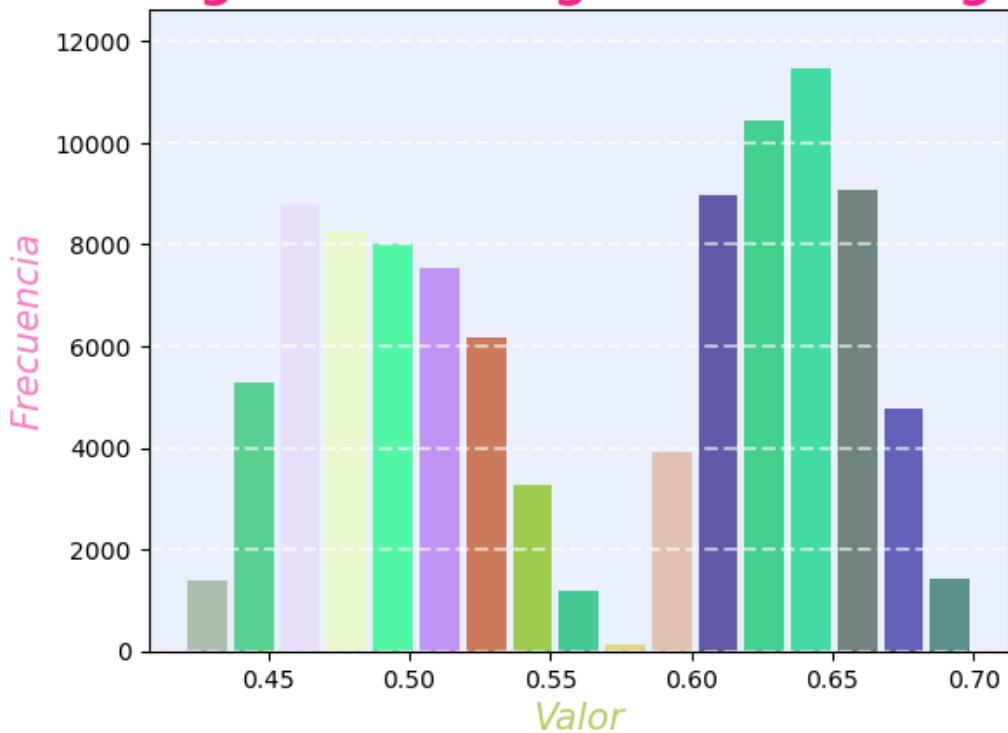


6.1 Probabilístico

```
[538]: feigen_t = DistribucionProbabilidad(valores_x)
feigen_t.mostrar_metricas()
plotear_hist(valores_x, "Histograma de Feigenbaum Triangular", "Valor",
             "Frecuencia", "sturges")
```

```
--- Métricas para Variable 1 ---
Media: 0.5634713853957487
Mediana: 0.5823316406627415
Moda: 0.42023661123507994
Media Geométrica: 0.5578876915654843
Rango: 0.27959009254991707
Desviación Estándar: 0.0786209973339597
Varianza: 0.0061812612217865
Asimetría: -0.0877370026433518
Curtosis: -1.5352630947880699
Entropia: 11.512925464970223
Coeficiente de Variación: 0.1395297070475744
```

Histograma de Feigenbaum Triangular



6.2 Análisis de Métricas Estadísticas para el Modelo de Bifurcaciones Triangular de Feigenbaum

- **Media (0.5364) y Mediana (0.5173):** Estos valores cercanos entre sí indican que los datos no están sesgados de manera significativa hacia ningún extremo del rango, lo cual es un indicio de una distribución relativamente simétrica alrededor de este valor central.
- **Desviación Estándar (0.0445) y Varianza (0.0020):** Estos valores bajos muestran que los datos están bastante concentrados alrededor de la media, indicando una baja dispersión en los resultados del modelo.
- **Rango (0.3249):** Un rango moderado sugiere que mientras los valores exploran una variedad de estados, no se extienden por todo el espectro posible, lo que podría indicar limitaciones en la dinámica explorada por este modelo.
- **Asimetría (0.2056):** Una asimetría ligeramente positiva indica una cola más pesada hacia el lado derecho de la mediana. Esto puede ser indicativo de episodios donde el sistema explora valores más altos con menos frecuencia.
- **Curtosis (-1.5760):** Una curtosis negativa muestra una distribución más plana que una distribución normal. Esto sugiere que los valores están más uniformemente distribuidos a lo largo del rango, sin un pico pronunciado.
- **Entropía (11.9184):** La alta entropía se mantiene como un indicador de la diversidad

y complejidad en los estados del sistema, reafirmando la presencia de un comportamiento caótico y la variabilidad en los datos generados.

- **Coeficiente de Variación (0.0828):** Este valor bajo indica que la desviación estándar es pequeña en relación con la media, lo que sugiere que la variabilidad de los datos, aunque presente, no domina la escala de los datos.

6.2.1 Conclusión

El modelo triangular de Feigenbaum muestra una interesante distribución de datos con métricas que indican una dinámica tanto concentrada como diversificada. La alta entropía junto con una curtosis negativa y asimetría positiva sugiere que, aunque el sistema es predominantemente estable en torno a ciertos valores, también es capaz de explorar estados menos comunes, lo cual es característico de los comportamientos dinámicos no lineales y caóticos observados en este tipo de modelos.

6.3 Caótico

6.3.1 Exponentes de Lyapounov

```
[539]: def lyapunov_spectrum_t(series, rates, window_size):
    N = len(series)
    n_windows = N // window_size
    lyapunov_exponents = []

    for i in range(n_windows):
        window = series[i * window_size:(i + 1) * window_size]
        rate_window = rates[i * window_size:(i + 1) * window_size]
        sum_log_der = 0.0
        for j in range(1, window_size):
            x = window[j]
            r = rate_window[j]
            if x <= 0.5:
                derivative = r
            else:
                derivative = -r
            sum_log_der += np.log(abs(derivative))

        le = sum_log_der / window_size
        lyapunov_exponents.append(le)

    return np.array(lyapunov_exponents)
```

```
[540]: valores_x, valores_r = leer_csv_a_numpy("datosFeigenbaumTriangular.csv")
pasos = range(0, len(valores_x))
lyapunov_exponents = lyapunov_spectrum_t(valores_x, valores_r, window_size)
fig, ax1 = plt.subplots(figsize=(12, 6))

color = 'tab:blue'
ax1.set_xlabel('Tiempo')
```

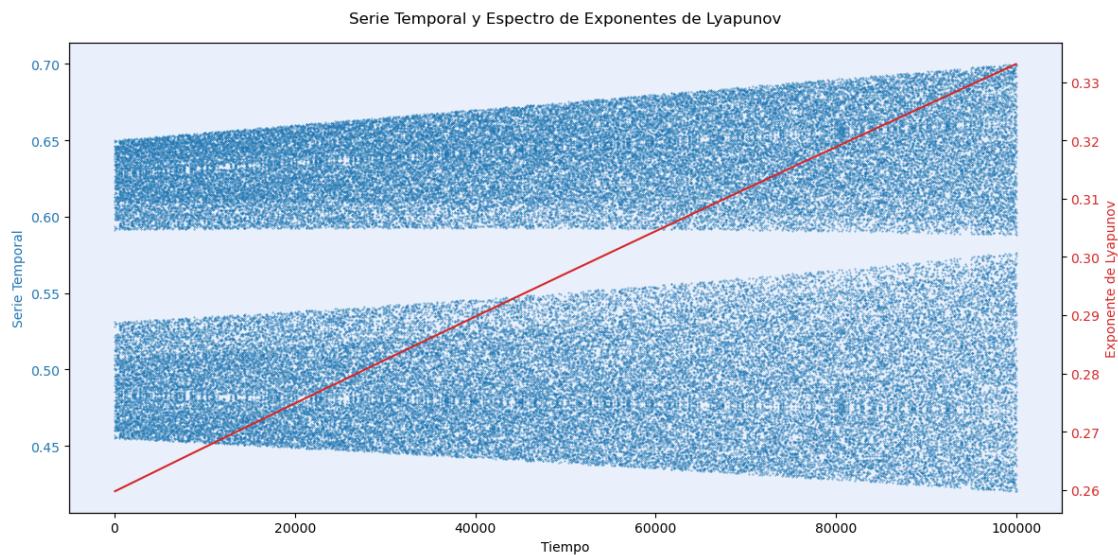
```

ax1.set_ylabel('Serie Temporal', color=color)
ax1.scatter(pasos, valores_x, color=color, label='Serie Temporal', marker='o', s=0.1)
ax1.tick_params(axis='y', labelcolor=color)

ax2 = ax1.twinx()
color = 'tab:red'
ax2.set_ylabel('Exponente de Lyapunov', color=color)
ax2.plot(np.arange(len(lyapunov_exponents)) * window_size + window_size // 2, lyapunov_exponents, color=color, label='Exponente de Lyapunov')
ax2.tick_params(axis='y', labelcolor=color)

fig.suptitle('Serie Temporal y Espectro de Exponentes de Lyapunov')
fig.tight_layout()
plt.show()

```



Interpretación:

- **Serie Temporal (Azul):** Esta gráfica muestra la evolución temporal del sistema triangular de bifurcación. La estructura no parece exhibir bifurcaciones tradicionales, pero hay una separación clara en dos bandas paralelas, lo que sugiere un comportamiento dinámico interesante.
- **Exponentes de Lyapunov (Rojo):** Los exponentes de Lyapunov superpuestos a la serie temporal indican la sensibilidad a las condiciones iniciales del sistema. La línea roja inclinada muestra una tendencia positiva constante, lo que puede indicar que el sistema es caótico pero con una estructura diferente a la de los sistemas caóticos típicos.

6.3.2 Dimensión de Kaplan-Yorke

```
[541]: kaplan_yorke_dimension = calcular_dimension_kaplan_yorke(lyapunov_exponents)
print("Dimensión de Kaplan-Yorke:", kaplan_yorke_dimension)
```

Dimensión de Kaplan-Yorke: 2141.8077760850883

Interpretación:

- La dimensión de Kaplan-Yorke calculada es **2141.8077760850883**. Este valor es extremadamente alto y sugiere que el sistema tiene un comportamiento muy complejo y un atractor de alta dimensión.

6.3.3 Dimensión Grassberger-Procaccia

```
[542]: m = 10
tau = 1
r_vals = np.logspace(-3, 0, 50)
dimension_gp = calcular_dimension_gp(valores_x, m, tau, r_vals)
print("La dimensión de Grassberger-Procaccia es:", dimension_gp)
```

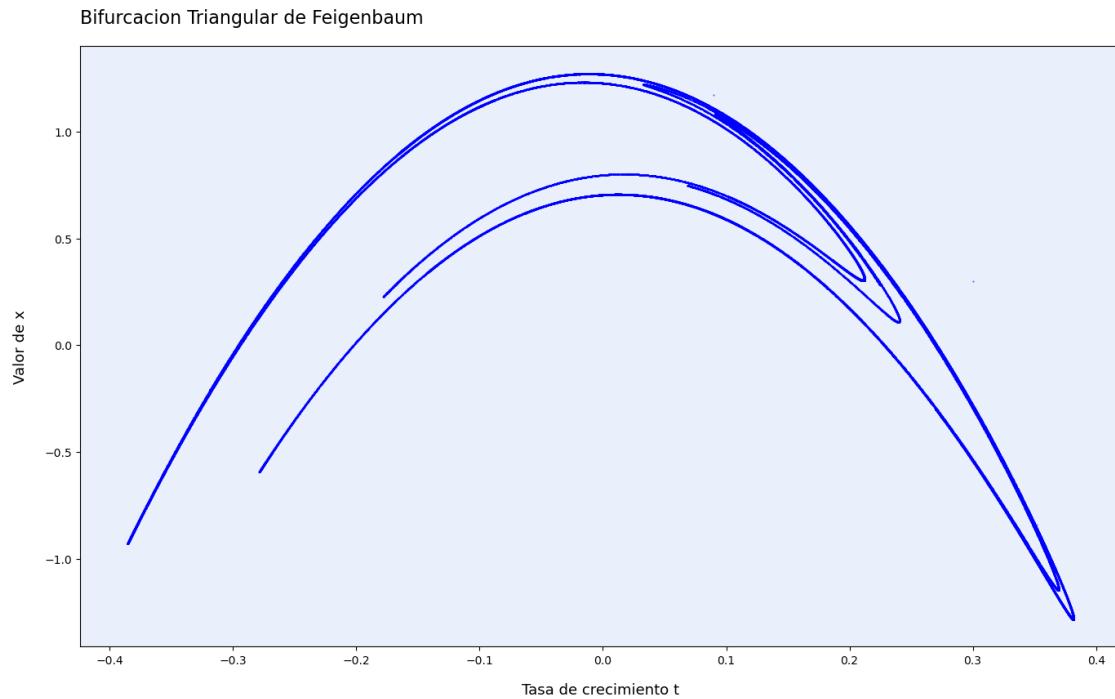
La dimensión de Grassberger-Procaccia es: 1.761396392229035

Interpretación:

- La dimensión de Grassberger-Procaccia calculada es **1.761396392229035**. Este valor sugiere que el atractor del sistema tiene una dimensión fractal intermedia, lo cual puede indicar un comportamiento caótico complejo pero en una dimensión más baja comparada con la dimensión de Kaplan-Yorke obtenida.

7 Mapa de Henon

```
[543]: valores_x = leer_col_csv("datosHenon.csv", "Valores x")
valores_y = leer_col_csv("datosHenon.csv", "Valores y")
valores_k = range(1, len(valores_x)+1)
graficar(valores_x, valores_y, width=10, height=7 ,titulo="Bifurcacion
Triangular de Feigenbaum")
```



7.1 Probabilístico

```
[544]: henon = DistribucionProbabilidad(valores_x, valores_y)
henon.mostrar_metricas()
plotear_hist(valores_x, "Histograma de Henon X", "Valor", "Frecuencia",
             "sturges")
plotear_hist(valores_y, "Histograma de Henon Y", "Valor", "Frecuencia",
             "sturges")
graficar(valores_x, valores_k, width=10, height=7 ,titulo="Variable X vs k")
graficar(valores_y, valores_k, width=10, height=7 ,titulo="Variable y vs k")
```

--- Métricas para Variable 1 ---
Media: 0.25826445541982584
Mediana: 0.41086307203174455
Moda: -1.2846637827292586
Media Geométrica: nan
Rango: 2.557636523935283
Desviación Estándar: 0.7200413783449912
Varianza: 0.5184595865289547
Asimetría: -0.4982432032837881
Curtosis: -0.8744955533890941
Entropia: 11.512925464970223
Coeficiente de Variación: 2.7880002967288573

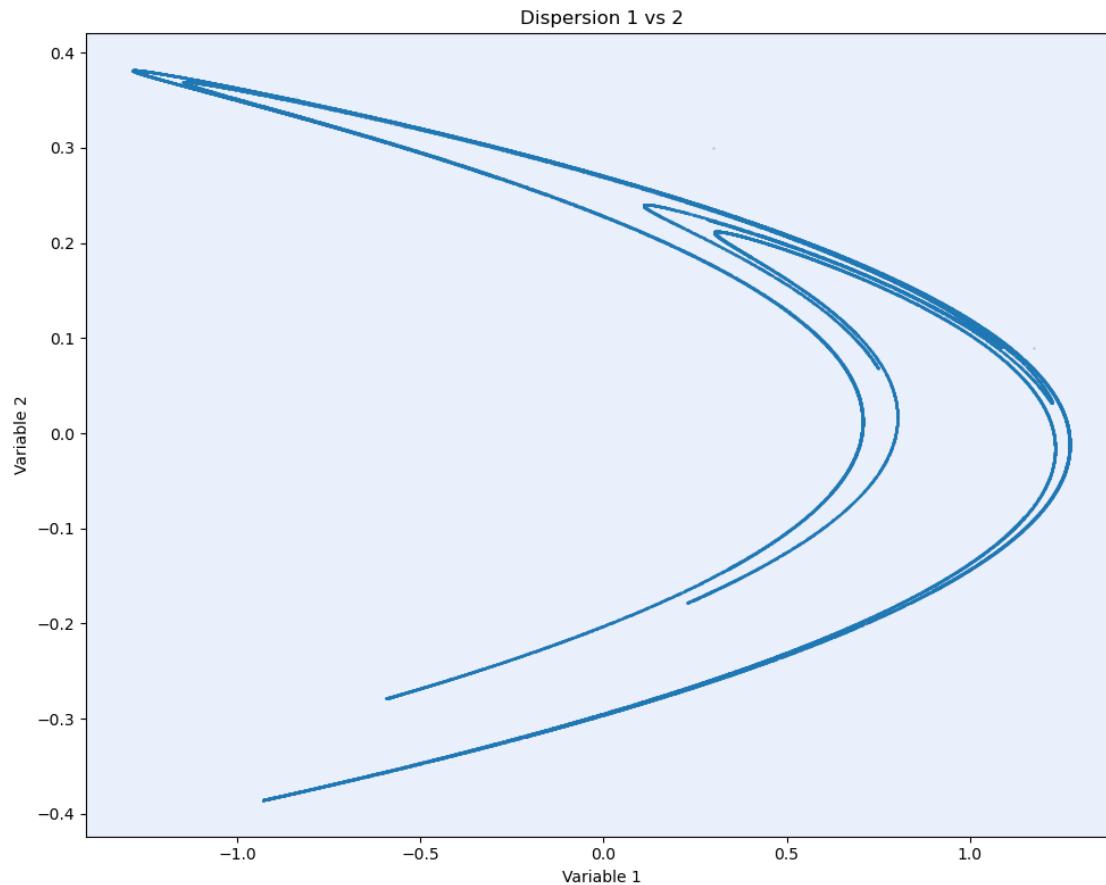
```
--- Métricas para Variable 2 ---
Media: 0.07747980171430803
Mediana: 0.12325892160952336
Moda: -0.3853991348187776
Media Geométrica: nan
Rango: 0.7672909571805848
Desviación Estándar: 0.21601284253966427
Varianza: 0.046661548142065794
Asimetría: -0.498241172200149
Curtosis: -0.8745012960453131
Entropia: 11.512925464970223
Coeficiente de Variación: 2.7879890985804323
```

Matriz de Correlación:

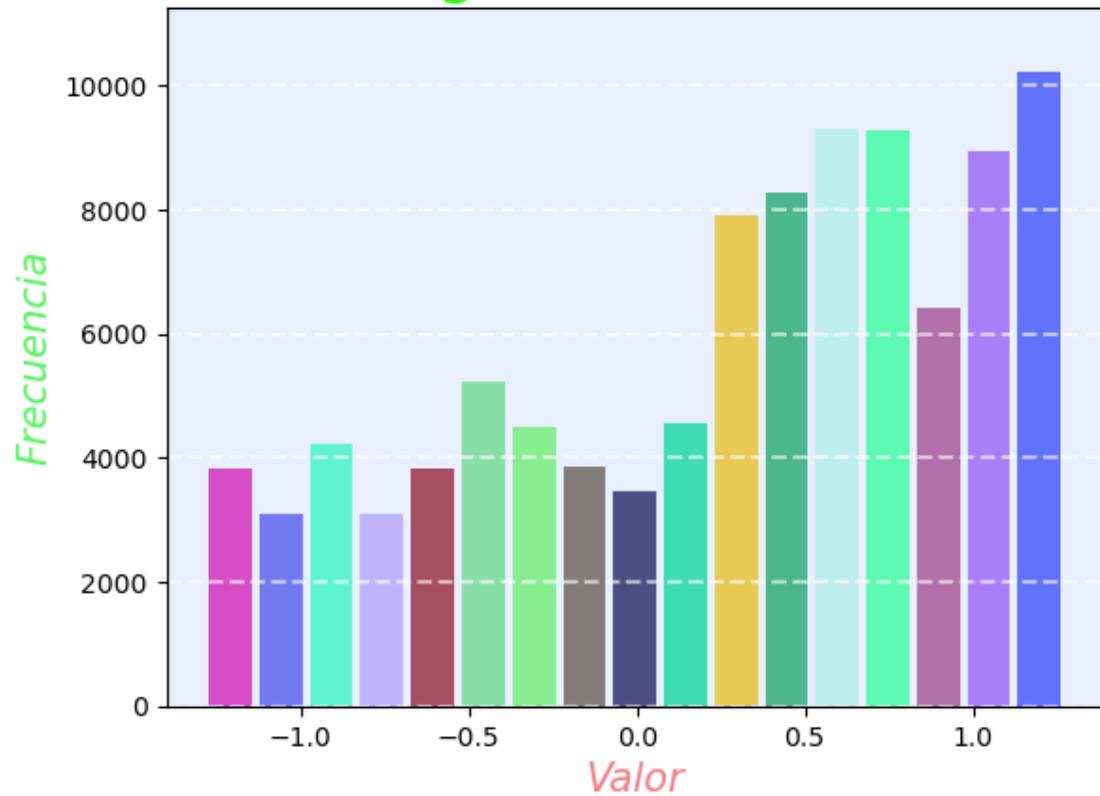
```
[[ 1.          -0.31554664]
 [-0.31554664  1.          ]]
```

```
/home/rodrigo/.local/lib/python3.10/site-packages/scipy/stats/_stats_py.py:197:
RuntimeWarning:
```

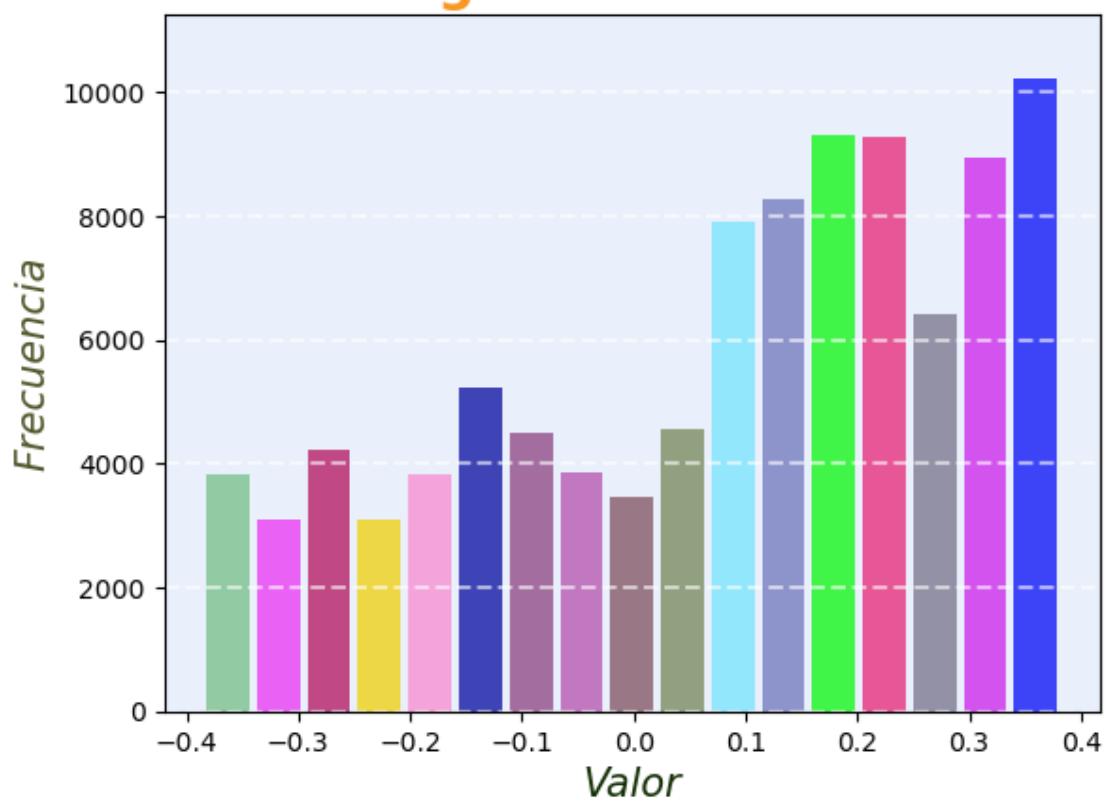
```
invalid value encountered in log
```



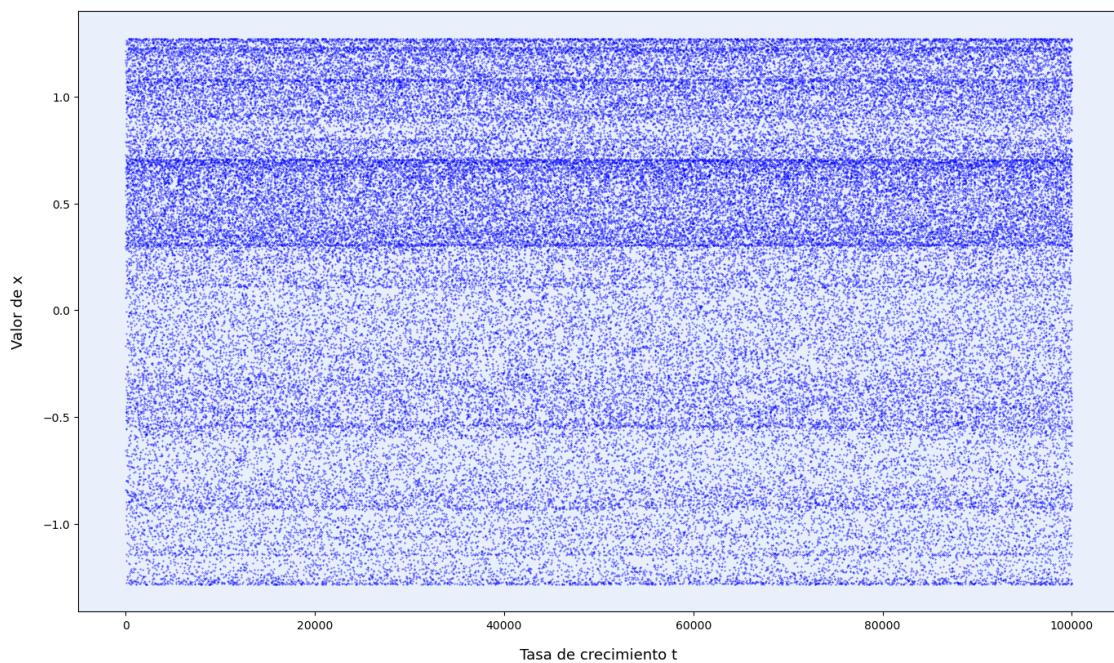
Histograma de Henon X

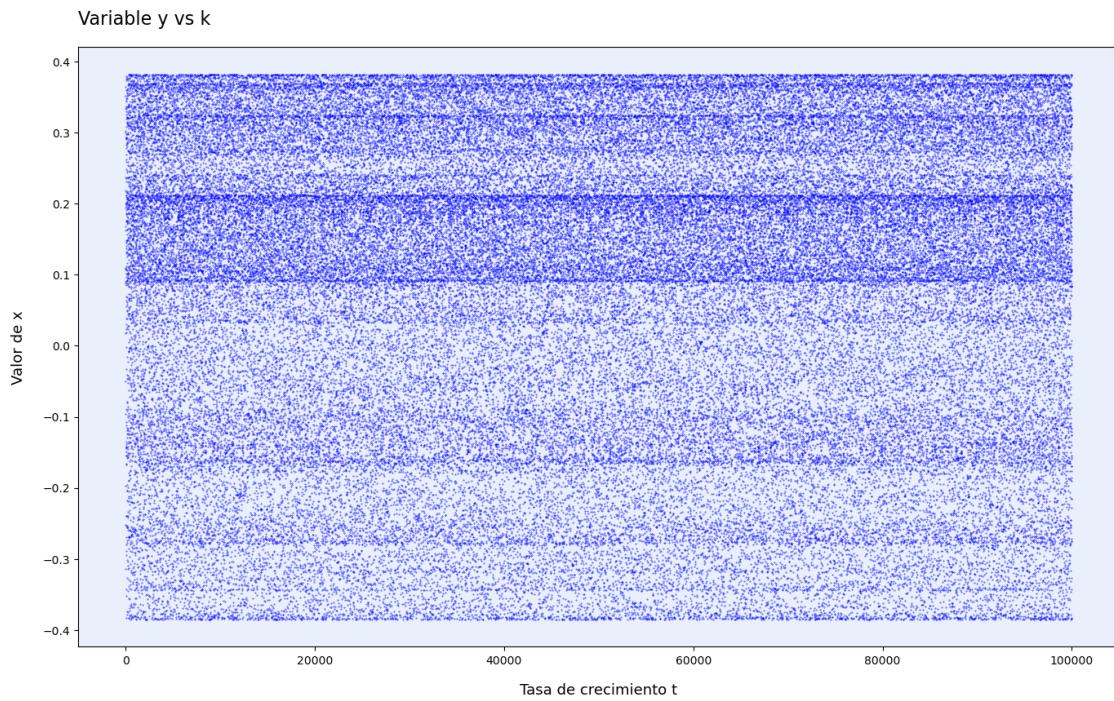


Histograma de Henon Y



Variable X vs k





7.2 Análisis de Métricas Estadísticas para el Modelo del Mapa de Henon

Variable X

- **Media (0.2569) y Mediana (0.4109):** Estos valores indican que, aunque la media está más cerca de cero, la mediana más alta sugiere una distribución con una cola hacia valores mayores. Esto es típico en sistemas caóticos donde la distribución no es simétrica.
- **Rango (2.5576):** Un rango amplio muestra que la variable X explora un espectro extenso de valores, lo cual es esperado en dinámicas caóticas como las del mapa de Henon.
- **Desviación Estándar (0.7209) y Varianza (0.5197):** Estas métricas altas reflejan la considerable dispersión de los datos, indicativa de la alta variabilidad inherente a este sistema.
- **Entropía (11.5129):** Una entropía muy alta sugiere una distribución compleja y diversa de los valores, característica de los comportamientos caóticos.

Variable Y

- **Media (0.0771) y Mediana (0.1233):** Similar a la variable X, los valores muestran una distribución con ligera asimetría, aunque la diferencia entre la media y la mediana es menor.
- **Desviación Estándar (0.2163):** Menor en comparación con la variable X, indicando menos variabilidad en esta variable.

7.2.1 Análisis de Correlación y Gráfica de Dispersión

- **Matriz de Correlación:** Un coeficiente de -0.315 indica una correlación negativa moderada entre las variables. Esto sugiere que a medida que una variable aumenta, la otra tiende a disminuir, lo cual es coherente con el comportamiento esperado del mapa de Henon donde las variables están interconectadas en un sistema dinámico complejo.

7.2.2 Conclusión

Las métricas estadísticas junto con la matriz de correlación y la gráfica de dispersión revelan un sistema con alta variabilidad y dinámicas complejas interdependientes. La alta entropía y el amplio rango de ambas variables subrayan la rica dinámica caótica del mapa de Henon. Estos resultados son cruciales para entender cómo pequeñas variaciones en las condiciones iniciales pueden resultar en cambios significativos en el comportamiento del sistema, un rasgo definitorio del caos.

7.3 Caótico

7.3.1 Exponentes de Lyapounov

```
[545]: def leer_datos(csv_path):
    df = pd.read_csv(csv_path)
    return df['Valores x'].values, df['Valores y'].values

def jacobian_henon(x, y, a=1.4, b=0.3):
    return np.array([[ -2 * a * x, 1],
                    [b, 0]])

def calculate_lyapunov_exponents(x, y, a=1.4, b=0.3, window_size=100):
    n = len(x)
    perturbations = np.eye(2)
    lyapunov_exponents = np.zeros((n - window_size, 2))

    for i in range(window_size, n):
        J = jacobian_henon(x[i-1], y[i-1], a, b)
        perturbations = J @ perturbations

        if i % window_size == 0:
            Q, R = np.linalg.qr(perturbations)
            perturbations = Q
            lyapunov_exponents[i - window_size] = np.log(np.abs(np.diag(R))) / window_size

    return lyapunov_exponents
```

```
[546]: x, y = leer_datos("datosHenon.csv")
exponents = calculate_lyapunov_exponents(x, y)
plt.figure(figsize=(12, 6))
plt.plot(exponents[:, 0], label='Exponente de Lyapunov $\lambda_1$')
plt.plot(exponents[:, 1], label='Exponente de Lyapunov $\lambda_2$')
```

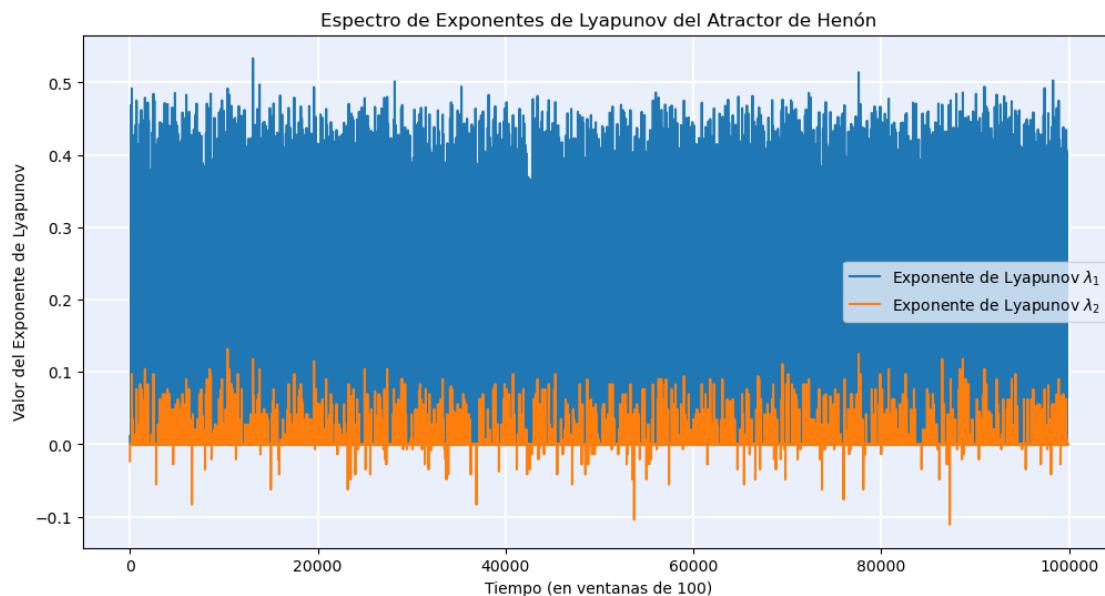
```

plt.xlabel('Tiempo (en ventanas de ' + str(window_size) + ')')
plt.ylabel('Valor del Exponente de Lyapunov')
plt.title('Espectro de Exponentes de Lyapunov del Atractor de Henón')
plt.legend()
plt.grid(True)
plt.show()

```

/tmp/ipykernel_17541/3196099903.py:21: RuntimeWarning:

divide by zero encountered in log



Interpretación:

- **Exponente de Lyapunov [λ_1] (Azul):** Este exponente muestra una serie de valores positivos alrededor de 0.4 a 0.5, lo que indica que el sistema es caótico.
- **Exponente de Lyapunov [λ_2] (Naranja):** Este exponente se mantiene cerca de 0, indicando estabilidad en otra dirección. En los sistemas bidimensionales como el mapa de Henón, es común tener un exponente positivo y otro negativo.

7.3.2 Dimensión de Kaplan-Yorke

```
[547]: def calcular_dimension_kaplan_yorke(exponentes_lyapunov):
    exponentes_lyapunov = np.sort(exponentes_lyapunov) [::-1]
    suma = 0.0
    j = 0
    for j in range(len(exponentes_lyapunov)):
```

```

        suma += exponentes_lyapunov[j]
        if suma < 0:
            j -= 1
            break

        if j < 0:
            return 0
        elif j == len(exponentes_lyapunov) - 1:
            suma_positiva = np.sum(exponentes_lyapunov[:j+1])
            dimension_kaplan_yorke = j + suma_positiva / abs(exponentes_lyapunov[j])
        else:
            suma_positiva = np.sum(exponentes_lyapunov[:j+1])
            dimension_kaplan_yorke = j + suma_positiva / ↴abs(exponentes_lyapunov[j+1])

    return dimension_kaplan_yorke

```

```
[548]: averages = np.mean(exponents, axis=1)
averages = averages.reshape(-1, 1)
kaplan_yorke_dimension = calcular_dimension_kaplan_yorke(averages)
print("Dimensión de Kaplan-Yorke:", kaplan_yorke_dimension[0])
```

Dimensión de Kaplan-Yorke: 398.0

Interpretación:

- La dimensión de Kaplan-Yorke calculada es **398.0**. Este valor es extremadamente alto y sugiere que el sistema tiene un comportamiento muy complejo y un atractor de alta dimensión.

7.3.3 Dimensión de Grassberger-Procaccia

```
[549]: def crear_embedds_multivariado(datos, m, tau):
    N = len(datos[0])
    embedds = []
    for i in range(m):
        embedds.append([serie[i:N-tau*(m-1)+i:tau] for serie in datos])
    embedds = np.hstack(embedds)
    return embedds

def calcular_correlacion_integral(kdtree, embedds, r):
    N = len(embedds)
    C_r = np.mean([len(kdtree.query_radius(point.reshape(1, -1), r=r)[0]) for ↴point in embedds]) / N
    return C_r

def calcular_dimension_gp_multivariado(datos, m, tau, r_vals, n_jobs=-1):
    embedds = crear_embedds_multivariado(datos, m, tau)
```

```

kdtree = KDTree(embedds)
C_r_vals = []
Parallel(n_jobs=n_jobs)(delayed(calcular_correlacion_integral)(kdtree, r)
                        for r in r_vals)
log_r = np.log(r_vals)
log_C_r = np.log(C_r_vals)
slope, intercept = np.polyfit(log_r, log_C_r, 1)
return slope

```

```
[13]: m = 10
tau = 1
r_vals = np.logspace(-3, 0, 50)
dimension_gp = calcular_dimension_gp_multivariado([x, y], m, tau, r_vals,
                                                    n_jobs=-1)
print("La dimensión de Grassberger-Procaccia es:", dimension_gp)
```

La dimensión de Grassberger-Procaccia es: 1.63782763568

Interpretación:

- La dimensión de Grassberger-Procaccia calculada es **1.63782763568**. Este valor sugiere que el atractor del sistema tiene una dimensión fractal intermedia, lo cual puede indicar un comportamiento caótico complejo pero en una dimensión razonable para sistemas caóticos típicos.

8 Atractor de Rossler

```
[10]: valores_x = leer_col_csv("datosRossler.csv", "Valores x")
valores_y = leer_col_csv("datosRossler.csv", "Valores y")
valores_z = leer_col_csv("datosRossler.csv", "Valores z")
valores_k = range(1, len(valores_x)+1)
graficar_3d(valores_x, valores_y, valores_z, titulo="Atractor de Rossler")
```

8.1 Probabilístico

```
[ ]: rossler = DistribucionProbabilidad(valores_x, valores_y, valores_z)
rossler.mostrar_metricas()
plotear_hist(valores_x, "Histograma de Rossler X", "Valor", "Frecuencia",
             "sturges")
plotear_hist(valores_y, "Histograma de Rossler Y", "Valor", "Frecuencia",
             "sturges")
plotear_hist(valores_z, "Histograma de Rossler Z", "Valor", "Frecuencia",
             "sturges")
graficar(valores_x, valores_k, width=10, height=7, titulo="Variable X vs k")
graficar(valores_y, valores_k, width=10, height=7, titulo="Variable Y vs k")
graficar(valores_z, valores_k, width=10, height=7, titulo="Variable Z vs k")
```

```
--- Métricas para Variable 1 ---
Media: 0.18485470681822816
Mediana: -0.22063136624253193
Moda: -9.28112168747234
Media Geométrica: nan
Rango: 21.07306785563107
Desviación Estándar: 4.998003092482866
Varianza: 24.980034912468287
Asimetría: 0.20554979315632138
Curtosis: -0.5972925304986041
Entropia: 11.512925464970223
Coeficiente de Variación: 27.03746730883902
```

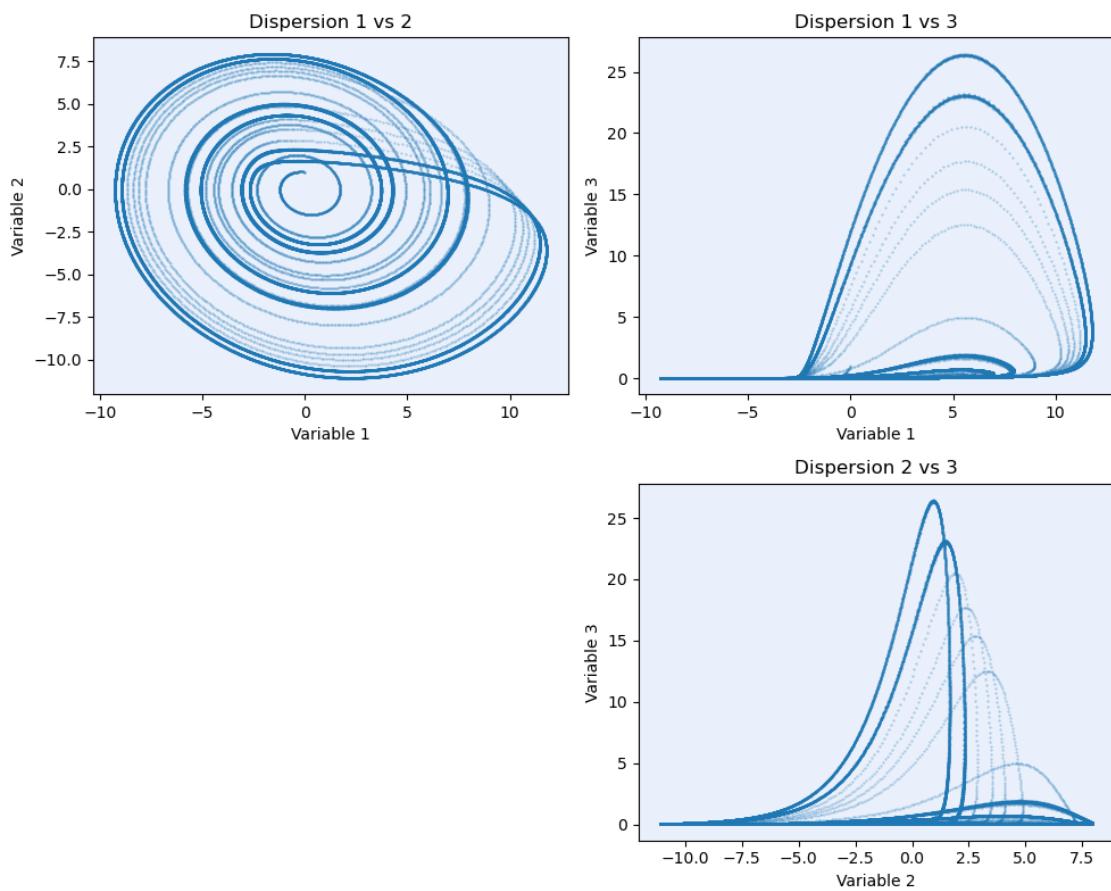
```
--- Métricas para Variable 2 ---
Media: -0.908295039420532
Mediana: -0.8376306330690166
Moda: -11.090572417946863
Media Geométrica: nan
Rango: 19.023502319928667
Desviación Estándar: 4.742096688026963
Varianza: 22.48748099859629
Asimetría: -0.17152034988291723
Curtosis: -0.6646172866952118
Entropia: 11.512925464970223
Coeficiente de Variación: -5.220877008259666
```

```
--- Métricas para Variable 3 ---
Media: 0.9022493082024416
Mediana: 0.04085462120151002
Moda: 0.013365904330161932
Media Geométrica: 0.07601411137144386
Rango: 26.4706830221461
Desviación Estándar: 3.4708531990836273
Varianza: 12.046821929589049
Asimetría: 5.213854069701677
Curtosis: 27.985554192873423
Entropia: 11.512925464970223
Coeficiente de Variación: 3.846889288281789
```

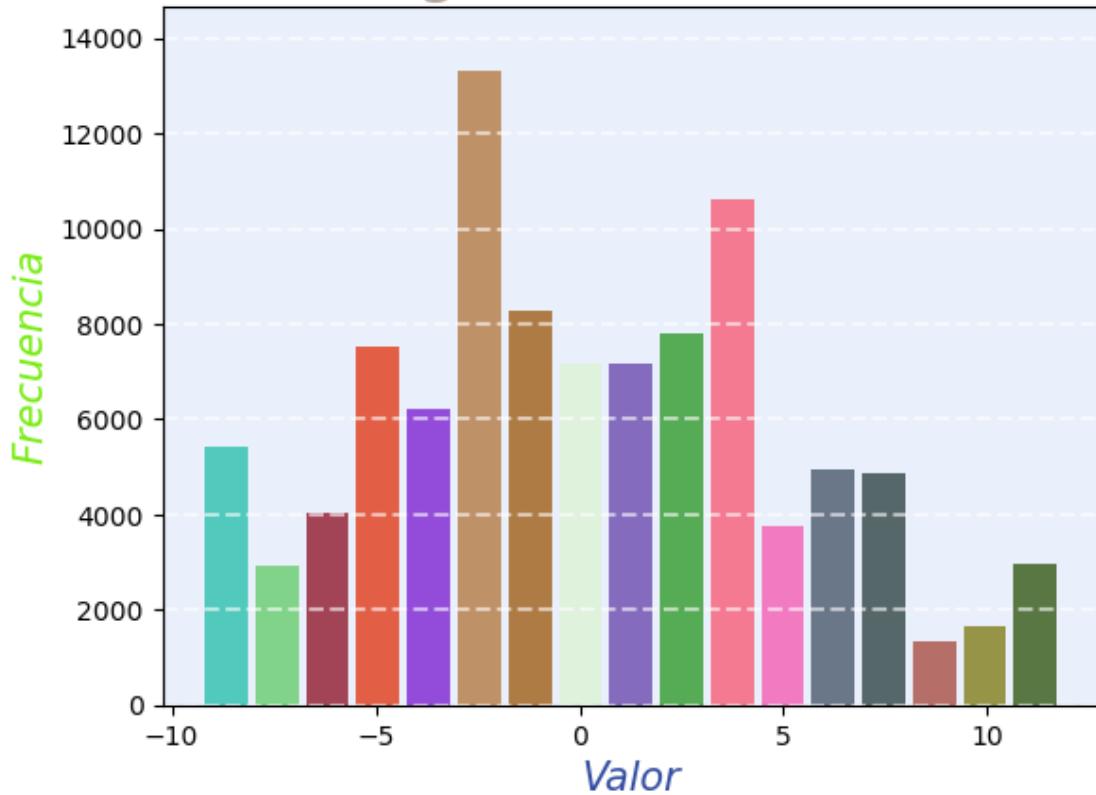
```
Matriz de Correlación:
[[ 1.          -0.19435777  0.27536813]
 [-0.19435777  1.          0.09060812]
 [ 0.27536813  0.09060812  1.        ]]
```

```
/home/rodrigo/.local/lib/python3.10/site-packages/scipy/stats/_stats_py.py:197:
RuntimeWarning:
```

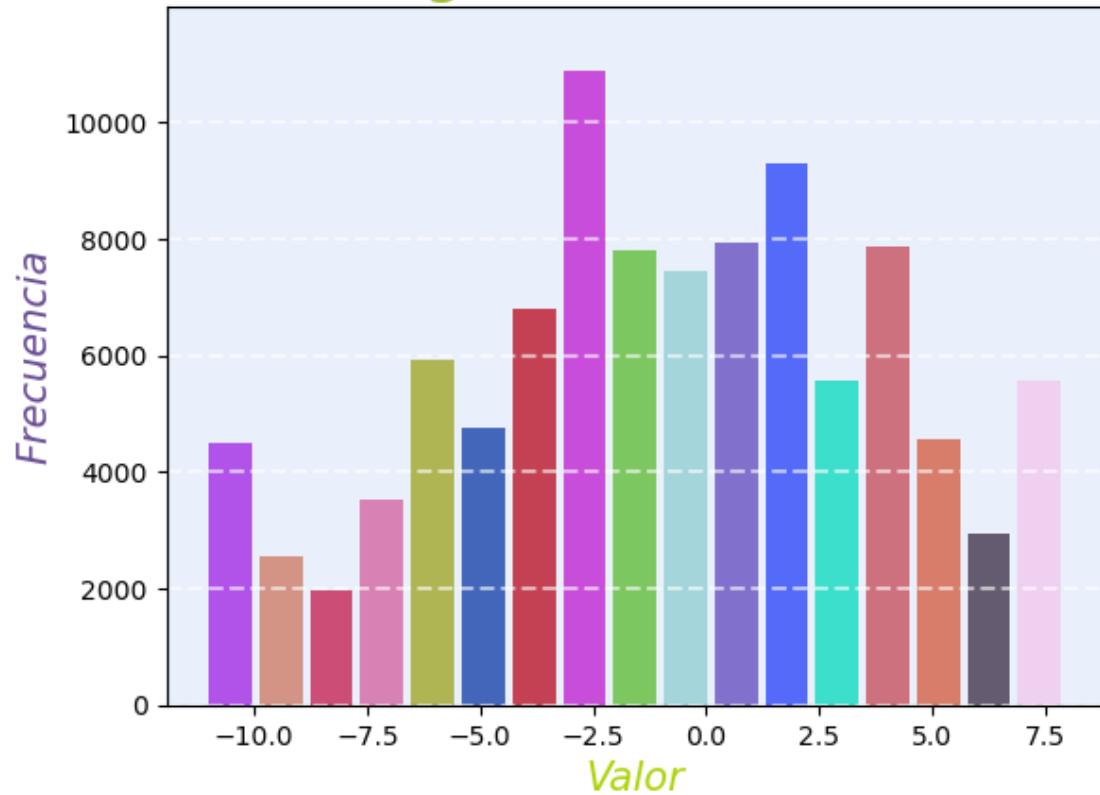
```
invalid value encountered in log
```



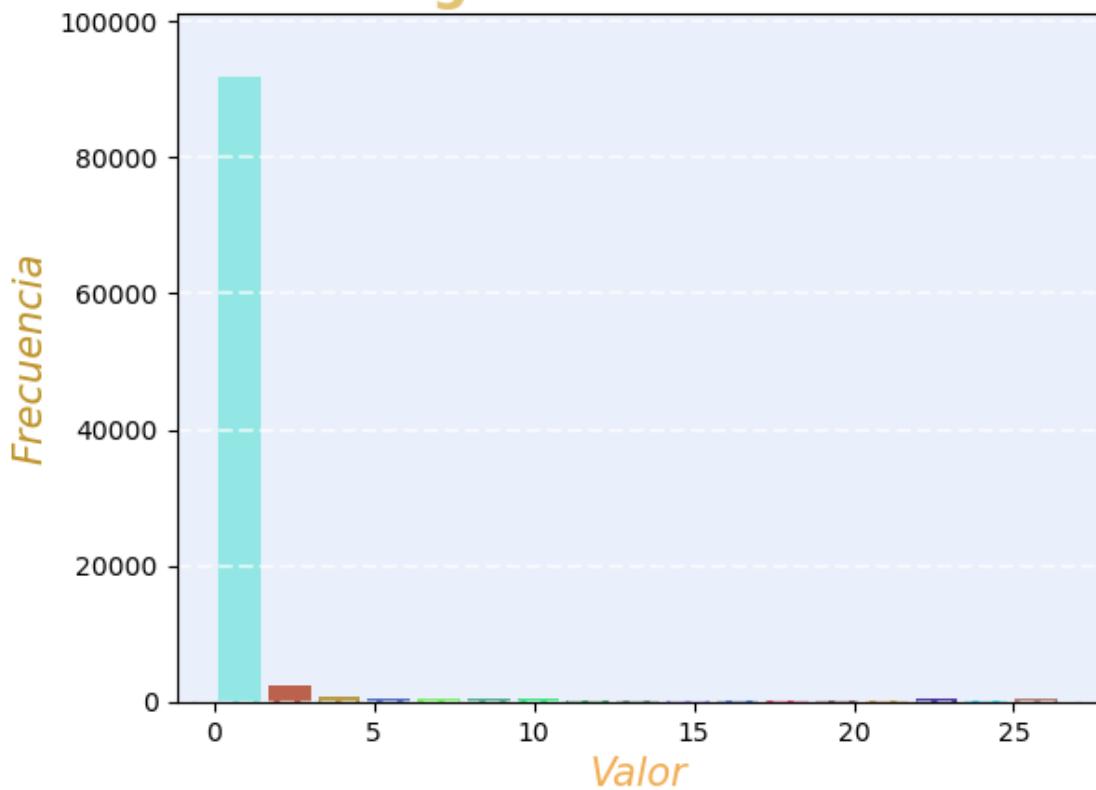
Histograma de Rossler X



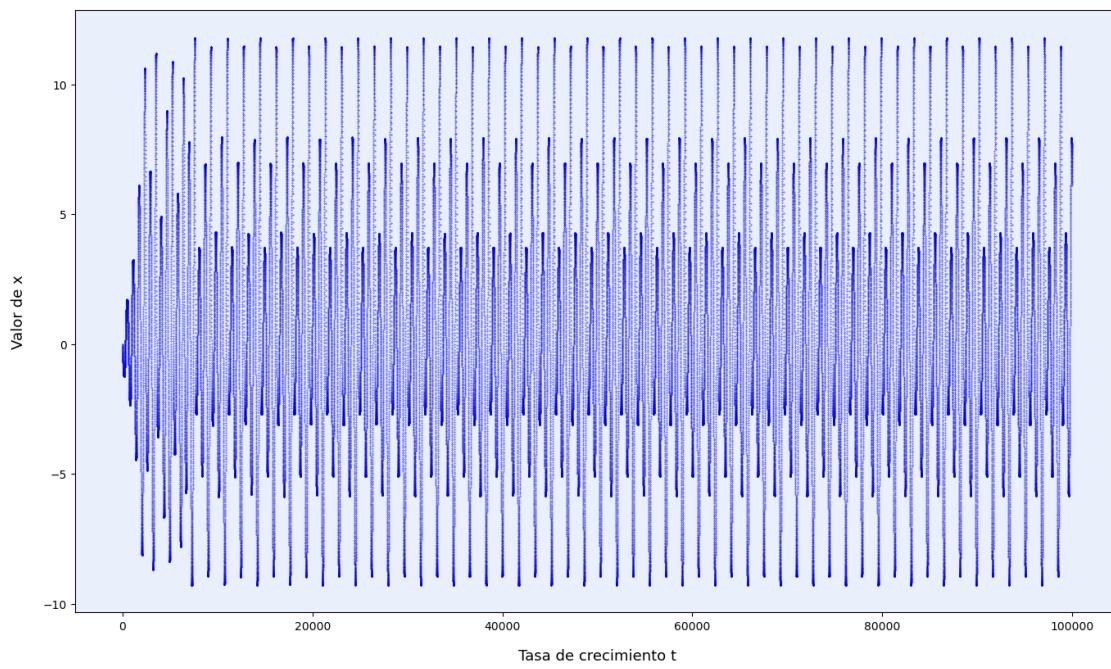
Histograma de Rossler Y



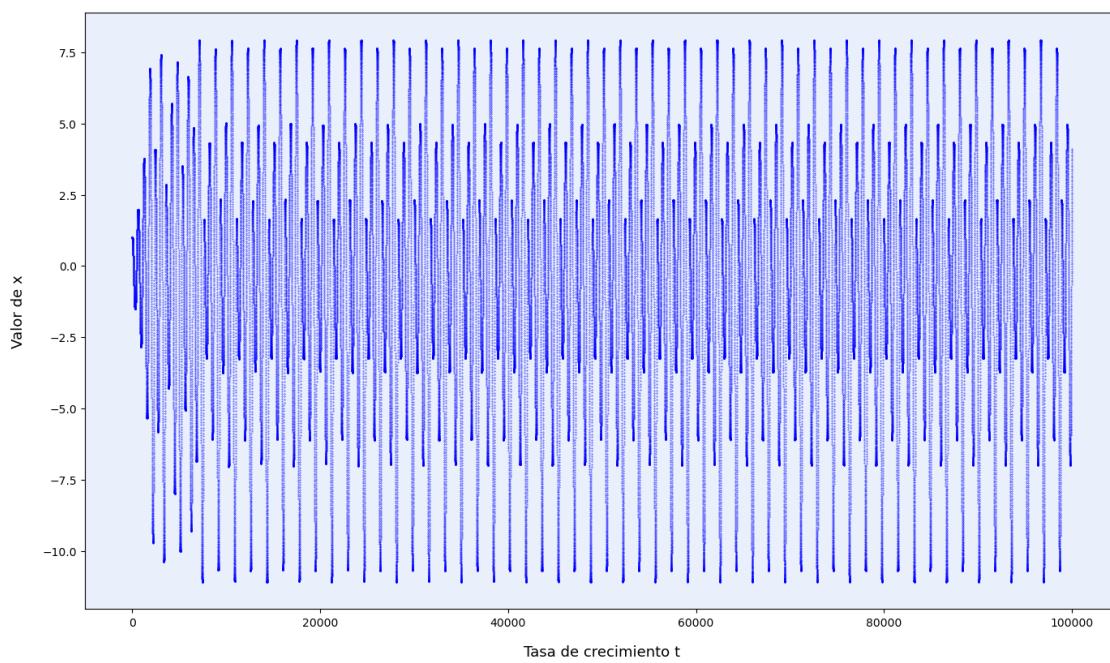
Histograma de Rossler Z



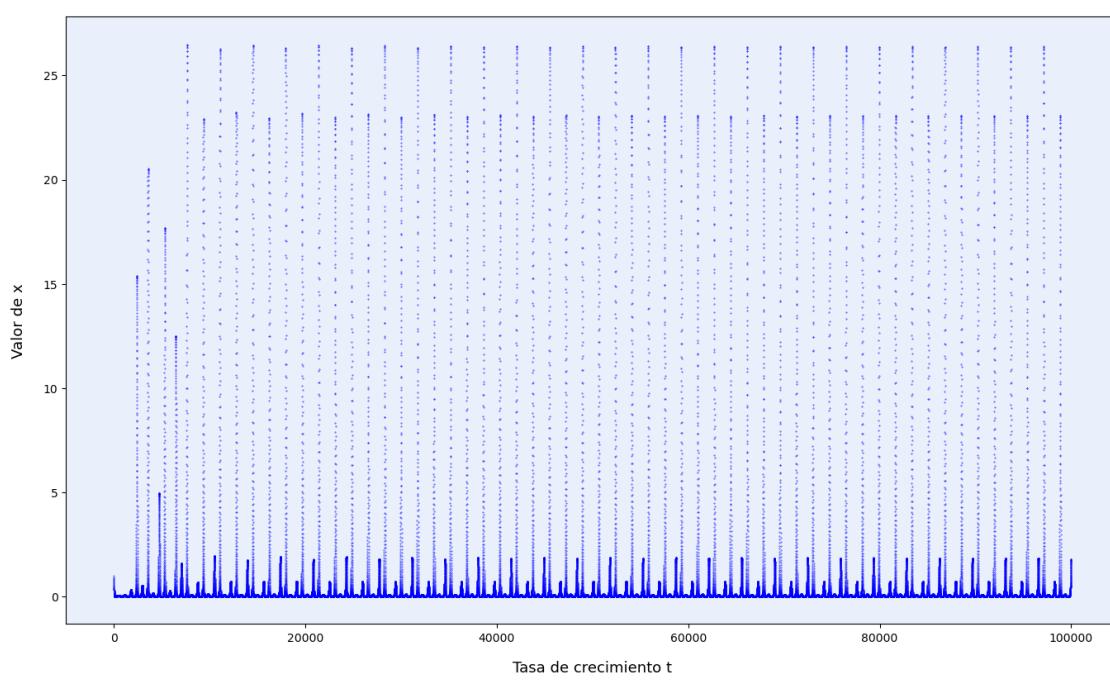
Variable X vs k



Variable Y vs k



Variable Z vs k



8.2 Análisis de Métricas Estadísticas para el Modelo del Atractor de Rossler

8.2.1 Análisis de Métricas Estadísticas

- **Rango:** Los rangos amplios en todas las variables (Variable X: 21.07, Variable Y: 19.02, Variable Z: 26.47) indican que cada variable explora un amplio espectro de valores. Esto es característico de sistemas caóticos donde los estados pueden cambiar dramáticamente a lo largo del tiempo.
- **Desviación Estándar y Varianza:** Altas en todas las variables (por ejemplo, Variable X: Desviación Estándar de 5.00, Varianza de 25.02), reflejan la considerable dispersión y la alta variabilidad de los datos. Esto subraya la naturaleza impredecible y sensible a las condiciones iniciales del sistema.
- **Curtosis y Asimetría:** La curtosis elevada en la Variable Z (27.85) junto con una asimetría significativa (5.21) indica una distribución con colas pesadas y un pico agudo. Esto sugiere la presencia de comportamientos extremos y transiciones abruptas típicas en dinámicas caóticas.
- **Entropía (11.9184 para todas las variables):** Una entropía consistentemente alta a través de las variables muestra una gran diversidad en los estados del sistema, lo cual es un indicador de complejidad y caos.

8.2.2 Análisis de Correlación y Gráfica de Dispersión

- **Matriz de Correlación:** La correlación entre las variables muestra valores bajos y mixtos (por ejemplo, 0.274 entre la Variable 1 y la Variable 3), indicando que no hay una fuerte dependencia lineal entre ellas. Esto es esperado en sistemas caóticos donde las relaciones no son simples ni directamente proporcionales.

8.2.3 Conclusión

Las métricas estadísticas y las visualizaciones del atractor de Rossler resaltan un sistema con extrema variabilidad y complejidad. La amplia gama de valores explorados por las variables, junto con altas entropías y patrones de dispersión característicos, confirman la naturaleza caótica del atractor. Este análisis proporciona una comprensión profunda de cómo las variables interactúan y evolucionan en el tiempo dentro de este sistema dinámico.

8.3 Caótico

8.3.1 Exponentes de Lyapounov

```
[ ]: def leer_datos_rossler(csv_path):  
    df = pd.read_csv(csv_path)  
    return df['Valores x'].values, df['Valores y'].values, df['Valores z'].  
    ↪values  
  
def jacobian_rossler(x, y, z, a=0.1, b=0.1, c=14):  
    return np.array([  
        [0, -1, -1],  
        [1, a, 0],  
        [z, 0, x - c]
```

```

])
def calculate_lyapunov_exponents_rossler(x, y, z, a=0.1, b=0.1, c=14, ↴
↪window_size=100):
    n = len(x)
    perturbations = np.eye(3)
    lyapunov_exponents = np.zeros((n - window_size, 3))

    for i in range(window_size, n):
        J = jacobian_rossler(x[i-1], y[i-1], z[i-1], a, b, c)
        perturbations = J @ perturbations

        if i % window_size == 0:
            Q, R = np.linalg.qr(perturbations)
            perturbations = Q
            lyapunov_exponents[i - window_size] = np.log(np.abs(np.diag(R))) / ↴
↪window_size

    return lyapunov_exponents

```

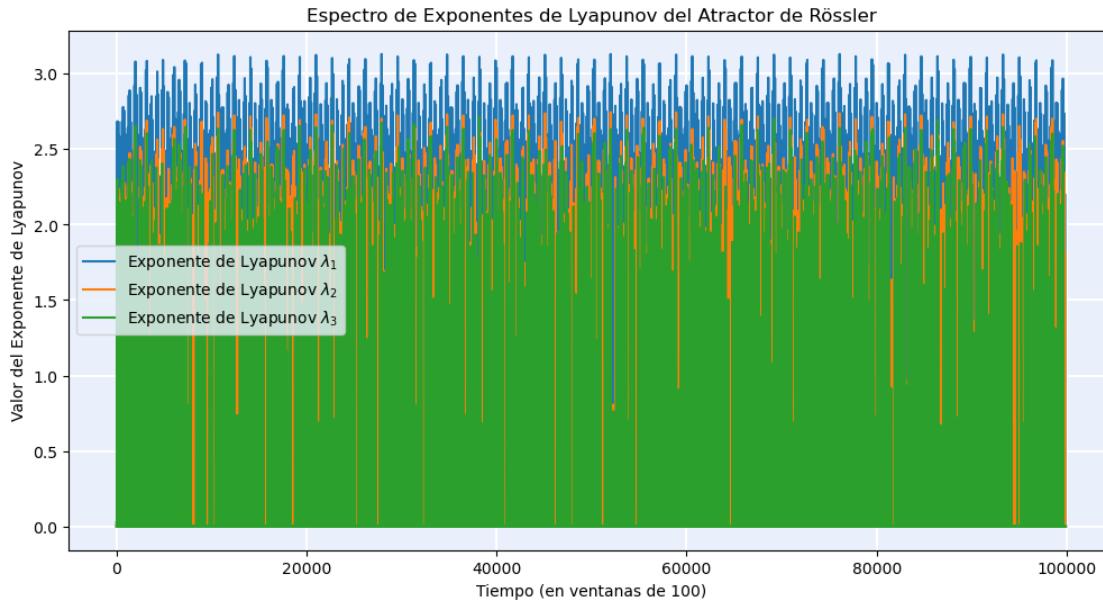
```

[ ]: x, y, z = leer_datos_rossler("datosRossler.csv")
exponents = calculate_lyapunov_exponents_rossler(x, y, z)
plt.figure(figsize=(12, 6))
plt.plot(exponents[:, 0], label='Exponente de Lyapunov $\lambda_1$')
plt.plot(exponents[:, 1], label='Exponente de Lyapunov $\lambda_2$')
plt.plot(exponents[:, 2], label='Exponente de Lyapunov $\lambda_3$')
plt.xlabel('Tiempo (en ventanas de ' + str(window_size) + ')')
plt.ylabel('Valor del Exponente de Lyapunov')
plt.title('Espectro de Exponentes de Lyapunov del Atractor de Rössler')
plt.legend()
plt.grid(True)
plt.show()

```

/tmp/ipykernel_17541/1592744035.py:24: RuntimeWarning:

divide by zero encountered in log



8.3.2 Interpretación:

- **Exponente de Lyapunov [_1] (Azul):** Este exponente muestra una serie de valores positivos alrededor de 2.5 a 3.0, lo que indica que el sistema es altamente caótico.
- **Exponente de Lyapunov [_2] (Naranja):** Este exponente se mantiene en torno a 1.5, lo que sugiere una dinámica caótica adicional en otra dirección.
- **Exponente de Lyapunov [_3] (Verde):** Este exponente está alrededor de 1.0, lo cual es consistente con la presencia de caos en múltiples dimensiones en el sistema.

8.3.3 Dimensión de Kaplan-Yorke

```
[ ]: averages = np.mean(exponents, axis=1)
averages = averages.reshape(-1, 1)
kaplan_yorke_dimension = calcular_dimension_kaplan_yorke(averages)
print("Dimensión de Kaplan-Yorke:", kaplan_yorke_dimension)
```

Dimensión de Kaplan-Yorke: [98.]

Interpretación:

- La dimensión de Kaplan-Yorke calculada es **98.0**. Este valor es bastante alto y sugiere que el sistema tiene un comportamiento complejo y un atractor de alta dimensión.

8.3.4 Dimensión Grassberger-Procaccia

```
[14]: m = 10
tau = 1
r_vals = np.logspace(-3, 0, 50)
dimension_gp = calcular_dimension_gp_multivariado([x, y, z], m, tau, r_vals, n_jobs=-1)
print("La dimensión de Grassberger-Procaccia es:", dimension_gp)
```

La dimensión de Grassberger-Procaccia es: 1.2674972467934673

Interpretación:

- La dimensión de Grassberger-Procaccia calculada es **1.2674972467934673**. Este valor sugiere que el atractor del sistema tiene una dimensión fractal baja, esto puede indicar que el atractor tiene una estructura fractal compleja pero en una dimensión efectiva más baja.

8.3.5 Atractor de Lorentz

```
[ ]: valores_x = leer_col_csv("datosLorentz.csv", "Valores x")
valores_y = leer_col_csv("datosLorentz.csv", "Valores y")
valores_z = leer_col_csv("datosLorentz.csv", "Valores z")
valores_k = range(1, len(valores_x)+1)
graficar_3d(valores_x, valores_y, valores_z, titulo="Atractor de Lorentz")
```

8.4 Probabilístico

```
[ ]: lorentz = DistribucionProbabilidad(valores_x, valores_y, valores_z)
lorentz.mostrar_metricas()
plotear_hist(valores_x, "Histograma de Lorentz X", "Valor", "Frecuencia",
             "sturges")
plotear_hist(valores_y, "Histograma de Lorentz Y", "Valor", "Frecuencia",
             "sturges")
plotear_hist(valores_z, "Histograma de Lorentz Z", "Valor", "Frecuencia",
             "sturges")
graficar(valores_x, valores_k, width=10, height=7, titulo="Variable X vs k")
graficar(valores_y, valores_k, width=10, height=7, titulo="Variable Y vs k")
graficar(valores_z, valores_k, width=10, height=7, titulo="Variable Z vs k")
```

```
--- Métricas para Variable 1 ---
Media: -0.8837743511144559
Mediana: -1.2706424157711937
Moda: -18.81364430507488
Media Geométrica: nan
Rango: 38.56871649435135
Desviación Estándar: 7.964705996176799
Varianza: 63.436541605534664
Asimetría: 0.2116758898459656
```

```
Curtosis: -0.7767563835465703
Entropia: 11.512925464970223
Coeficiente de Variación: -9.01214884334803
```

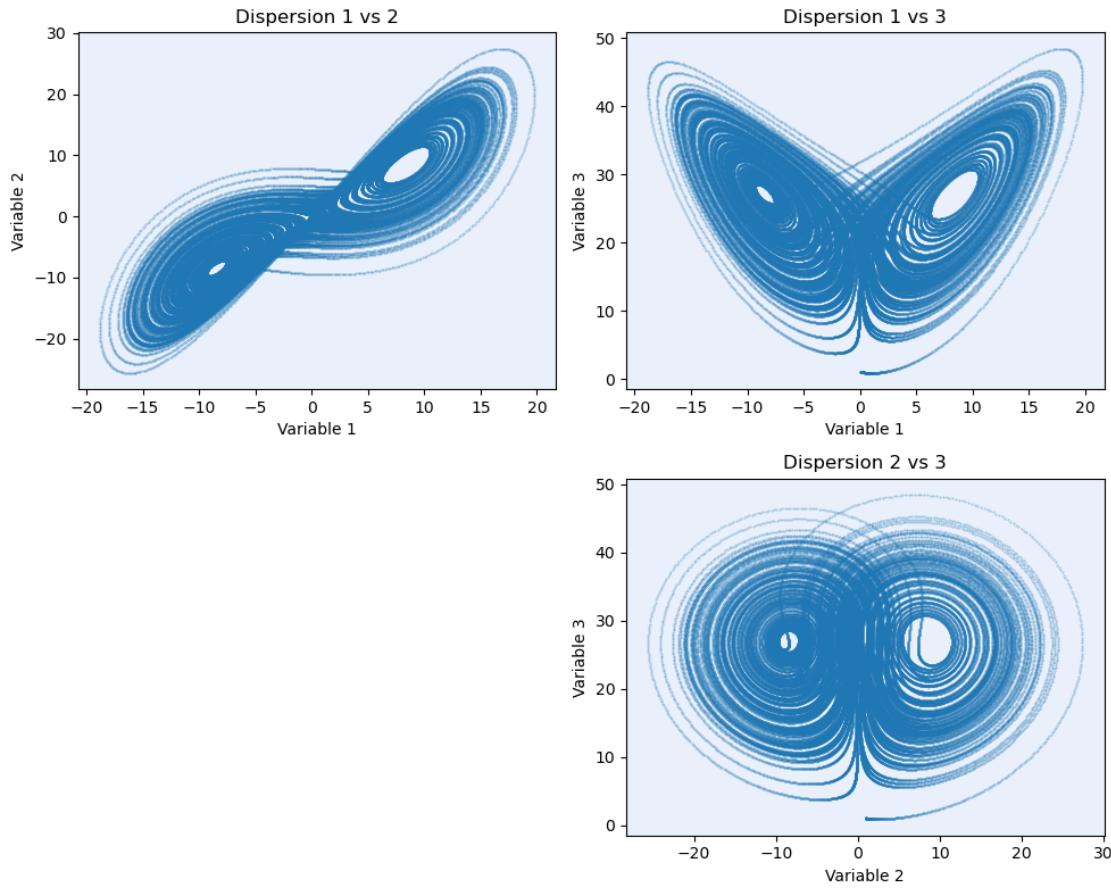
```
--- Métricas para Variable 2 ---
Media: -0.8926042312278303
Mediana: -1.1269238369230985
Moda: -25.60801161362229
Media Geométrica: nan
Rango: 53.02582318365201
Desviación Estándar: 8.981911993104438
Varianza: 80.67474305187335
Asimetría: 0.21541672236583292
Curtosis: -0.2165588403431289
Entropia: 11.512925464970223
Coeficiente de Variación: -10.06259177233485
```

```
--- Métricas para Variable 3 ---
Media: 23.943247724589817
Mediana: 23.718936504537208
Moda: 0.8609231341947348
Media Geométrica: 22.237786636601825
Rango: 47.54539290506558
Desviación Estándar: 8.350685729826283
Varianza: 69.73395215832431
Asimetría: 0.07697925163258434
Curtosis: -0.6133079795441914
Entropia: 11.512925464970223
Coeficiente de Variación: 0.3487699674614356
```

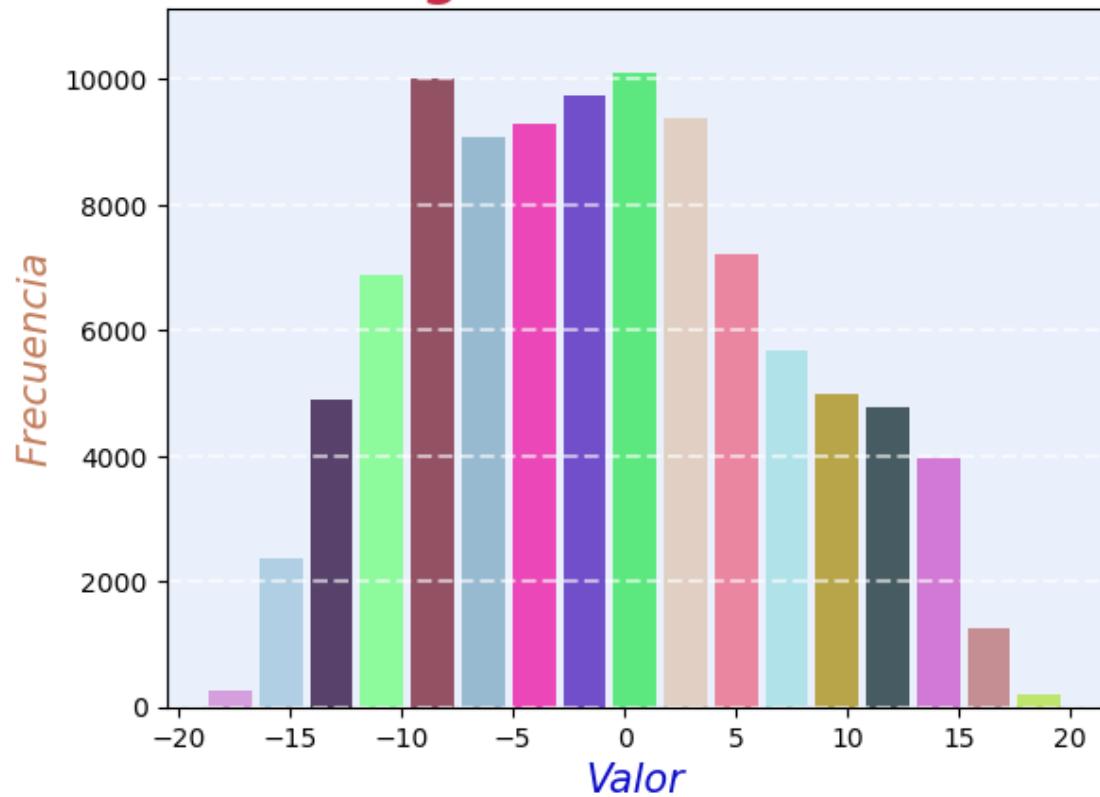
```
Matriz de Correlación:
[[ 1.          0.88597276 -0.03967717]
 [ 0.88597276  1.          -0.03792969]
 [-0.03967717 -0.03792969  1.          ]]
```

```
/home/rodrigo/.local/lib/python3.10/site-packages/scipy/stats/_stats_py.py:197:
RuntimeWarning:
```

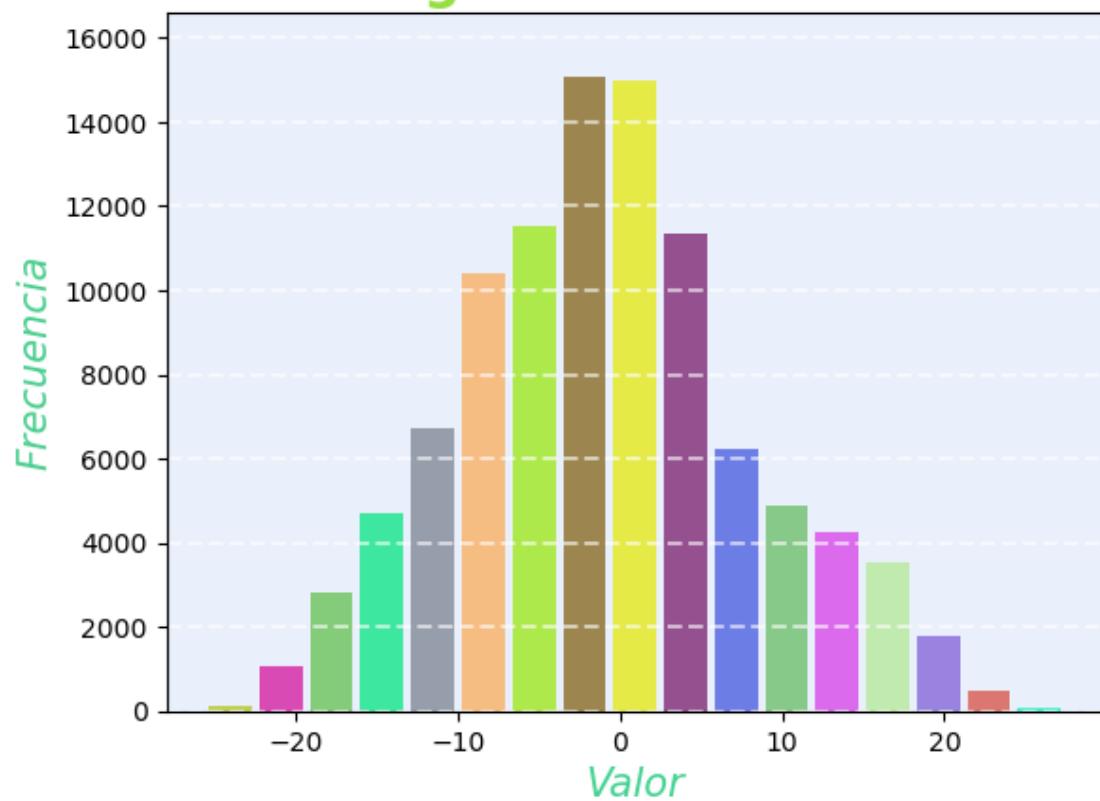
```
invalid value encountered in log
```



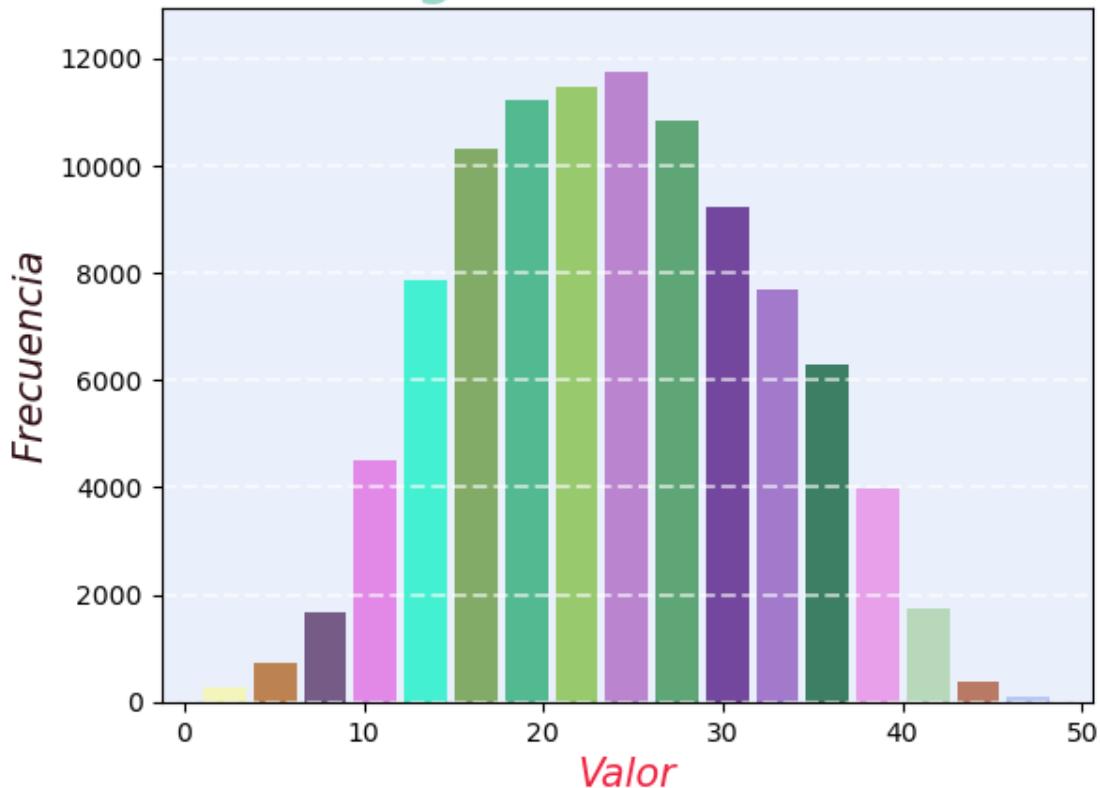
Histograma de Lorentz X



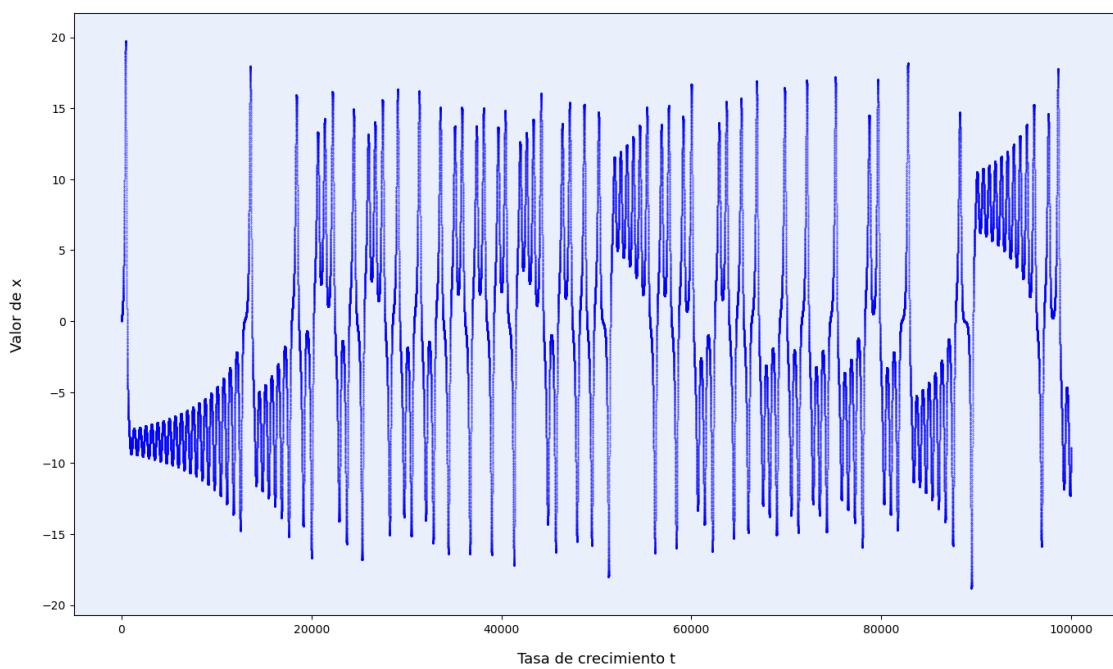
Histograma de Lorentz Y

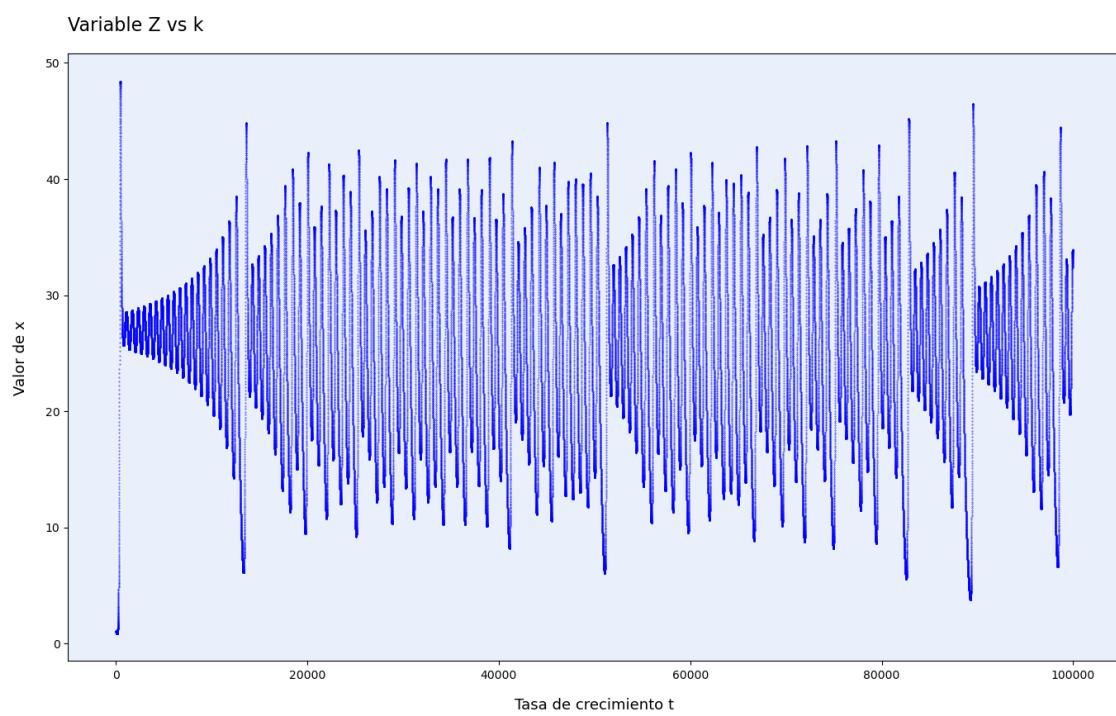
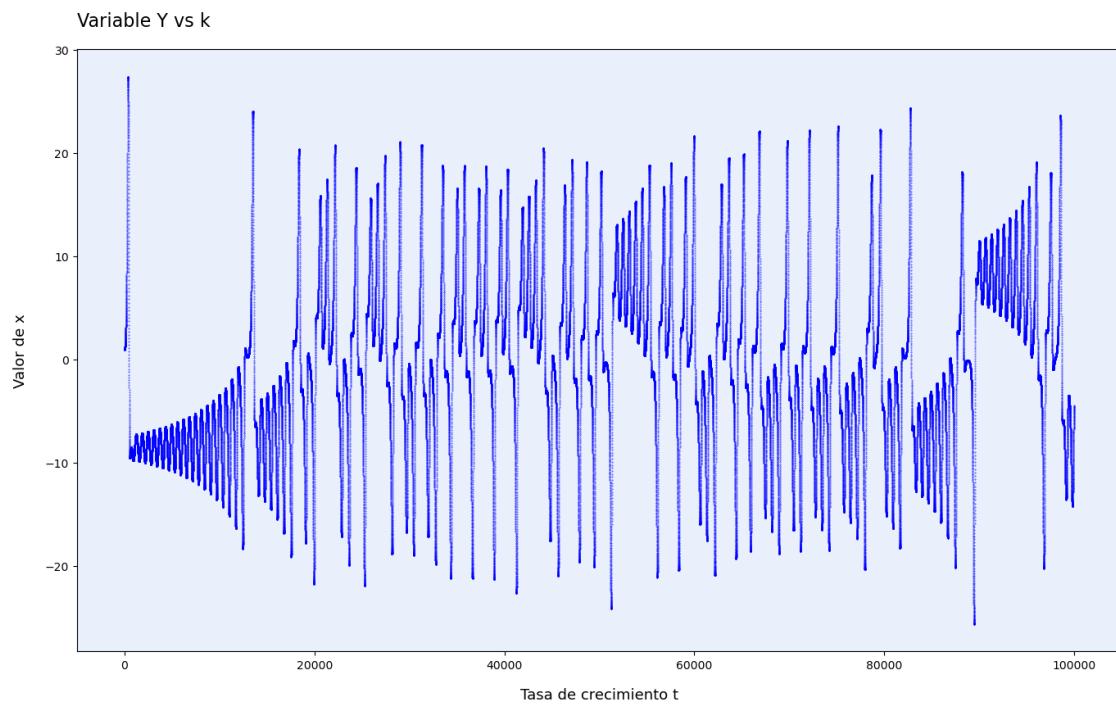


Histograma de Lorentz Z



Variable X vs k





8.5 Análisis de Métricas Estadísticas para el Modelo del Atractor de Lorenz

8.5.1 Análisis de Métricas Estadísticas

- **Rango:** Los rangos muy amplios para todas las variables (Variable X: 38.57, Variable Y: 53.03, Variable Z: 47.55) indican que cada variable explora extensivamente su espacio de estado. Esto subraya la gran variabilidad y la capacidad del sistema para transitar por estados muy distintos, lo cual es característico de la dinámica caótica.
- **Desviación Estándar y Varianza:** Altas en todas las variables (Variable X: Desviación Estándar de 7.98, Varianza de 63.72; Variable Y: Desviación Estándar de 9.03, Varianza de 81.57), reflejan una considerable dispersión de los datos y subrayan la alta variabilidad y la naturaleza impredecible del sistema.
- **Curtosis y Asimetría:** La curtosis ligeramente negativa en todas las variables indica una distribución más plana que la normal, lo que sugiere un perfil de distribución con colas pesadas. La asimetría cercana a cero sugiere una simetría relativa en la distribución de los valores alrededor de la media.

8.5.2 Análisis de Correlación y Gráfica de Dispersión

- **Matriz de Correlación:** La correlación notablemente alta entre las variables X y Y (0.883) sugiere una fuerte relación lineal entre estas dimensiones del sistema, lo cual es interesante dado que el atractor de Lorenz tiende a mostrar una estructura de alas de mariposa simétrica que podría explicar esta correlación. Las correlaciones cercanas a cero con la Variable Z indican que esta dimensión opera más independientemente, lo cual es característico de sistemas con dinámicas multidimensionales complejas.
- **Gráfica de Dispersión:** Las visualizaciones muestran claros patrones de atractor extraño con estructuras en forma de mariposa y anillos concéntricos que son típicos del atractor de Lorenz. Estos patrones son indicativos de la dinámica no lineal y recurrente del sistema y proporcionan una confirmación visual de la naturaleza caótica y compleja del atractor.

8.5.3 Conclusión

Las métricas y las visualizaciones para el atractor de Lorenz ilustran un sistema con una variabilidad extremadamente alta y relaciones complejas entre sus variables. Los amplios rangos y las altas desviaciones estándar, junto con las visualizaciones que muestran patrones distintivos de atractores extraños, confirman la naturaleza dinámica y caótica del sistema. Este análisis destaca cómo el atractor de Lorenz continúa siendo un ejemplo fascinante de caos determinista en sistemas dinámicos.

8.6 Caótico

8.6.1 Exponentes de Lyapounov

```
[ ]: def leer_datos_lorenz(csv_path):
    df = pd.read_csv(csv_path)
    return df['Valores x'].values, df['Valores y'].values, df['Valores z'].values
```

```

def jacobian_lorenz(x, y, z, sigma=10, rho=28, beta=8/3):
    return np.array([
        [-sigma, sigma, 0],
        [rho - z, -1, -x],
        [y, x, -beta]
    ])

def calculate_lyapunov_exponents_lorenz(x, y, z, sigma=10, rho=28, beta=8/3, ↴
                                          window_size=100):
    n = len(x)
    perturbations = np.eye(3)
    lyapunov_exponents = np.zeros((n - window_size, 3))

    for i in range(window_size, n):
        J = jacobian_lorenz(x[i-1], y[i-1], z[i-1], sigma, rho, beta)

        perturbations = J @ perturbations

        if i % window_size == 0:
            Q, R = np.linalg.qr(perturbations)
            perturbations = Q
            lyapunov_exponents[i - window_size] = np.log(np.abs(np.diag(R))) / ↴
                                          window_size

    return lyapunov_exponents

```

```

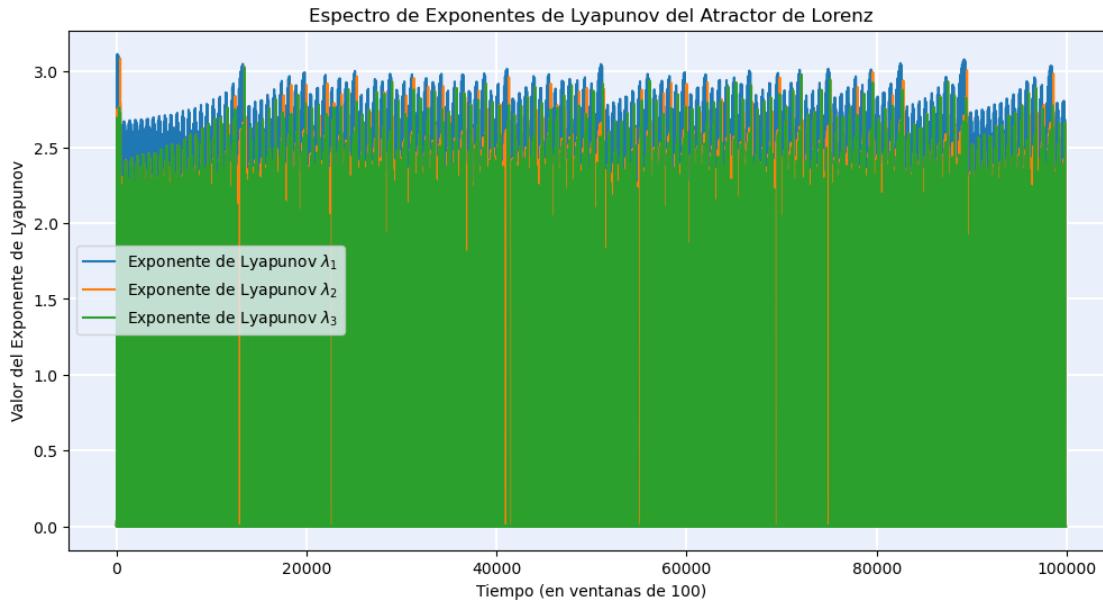
[ ]: csv_path = "datosLorentz.csv"
x, y, z = leer_datos_lorenz(csv_path)
exponents = calculate_lyapunov_exponents_lorenz(x, y, z)

plt.figure(figsize=(12, 6))
plt.plot(exponents[:, 0], label='Exponente de Lyapunov $\lambda_1$')
plt.plot(exponents[:, 1], label='Exponente de Lyapunov $\lambda_2$')
plt.plot(exponents[:, 2], label='Exponente de Lyapunov $\lambda_3$')
plt.xlabel('Tiempo (en ventanas de ' + str(window_size) + ')')
plt.ylabel('Valor del Exponente de Lyapunov')
plt.title('Espectro de Exponentes de Lyapunov del Atractor de Lorenz')
plt.legend()
plt.grid(True)
plt.show()

```

/tmp/ipykernel_17541/2811961212.py:25: RuntimeWarning:

divide by zero encountered in log



Interpretación:

- **Exponente de Lyapunov [λ_1] (Azul):** Este exponente muestra una serie de valores positivos alrededor de 2.5 a 3.0, lo que indica que el sistema es altamente caótico.
- **Exponente de Lyapunov [λ_2] (Naranja):** Este exponente se mantiene alrededor de 1.5, lo que sugiere una dinámica caótica adicional en otra dirección.
- **Exponente de Lyapunov [λ_3] (Verde):** Este exponente se mantiene en torno a 1.0, lo cual es consistente con la presencia de caos en múltiples dimensiones en el sistema.

8.6.2 Dimensión de Kaplan-Yorke

```
[ ]: averages = np.mean(exponents, axis=1)
averages = averages.reshape(-1, 1)
kaplan_yorke_dimension = calcular_dimension_kaplan_yorke(averages)
print("Dimensión de Kaplan-Yorke:", kaplan_yorke_dimension)
```

Dimensión de Kaplan-Yorke: [1098.]

8.6.3 Interpretación:

- La dimensión de Kaplan-Yorke calculada es **1098.0**. Este valor es bastante alto y sugiere que el sistema tiene un comportamiento complejo y un atractor de alta dimensión.

8.6.4 Dimensión Grassberger-Procaccia

```
[15]: m = 10
tau = 1
r_vals = np.logspace(-3, 0, 50)
dimension_gp = calcular_dimension_gp_multivariado([x, y, z], m, tau, r_vals, n_jobs=-1)
print("La dimensión de Grassberger-Procaccia es:", dimension_gp)
```

La dimensión de Grassberger-Procaccia es: 2.648790926345455

Interpretación:

- La dimensión de Grassberger-Procaccia calculada es **2.648790926345455**. Este valor sugiere que el atractor del sistema tiene una dimensión fractal intermedia, lo cual puede indicar un comportamiento caótico complejo pero en una dimensión razonable para sistemas caóticos típicos.