analisisCaos

April 30, 2024

Análisis Estadístico de Series de Tiempo Caóticas

Alumno: Rodrigo Gerardo Trejo Arriaga

Título de la Práctica: Estadísticas descriptivas de atractores caóticos

Este segmento de la práctica explora las propiedades estadísticas de los conjuntos de datos ge

Fecha de Entrega: 30 de Abril, 2024

[118]:
```
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
```

[119]:
```
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:])
```

0.1 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.

0.1.1 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.

0.1.2 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.

[120]:
```python
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),
        'Entropia': 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()
```

[121]:
```
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.
    """
    plt.style.use('dark_background')
```

```
    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()
```

[122]:
```
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.style.use('seaborn-darkgrid')
    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()
```

[123]:
```
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)
    )
```

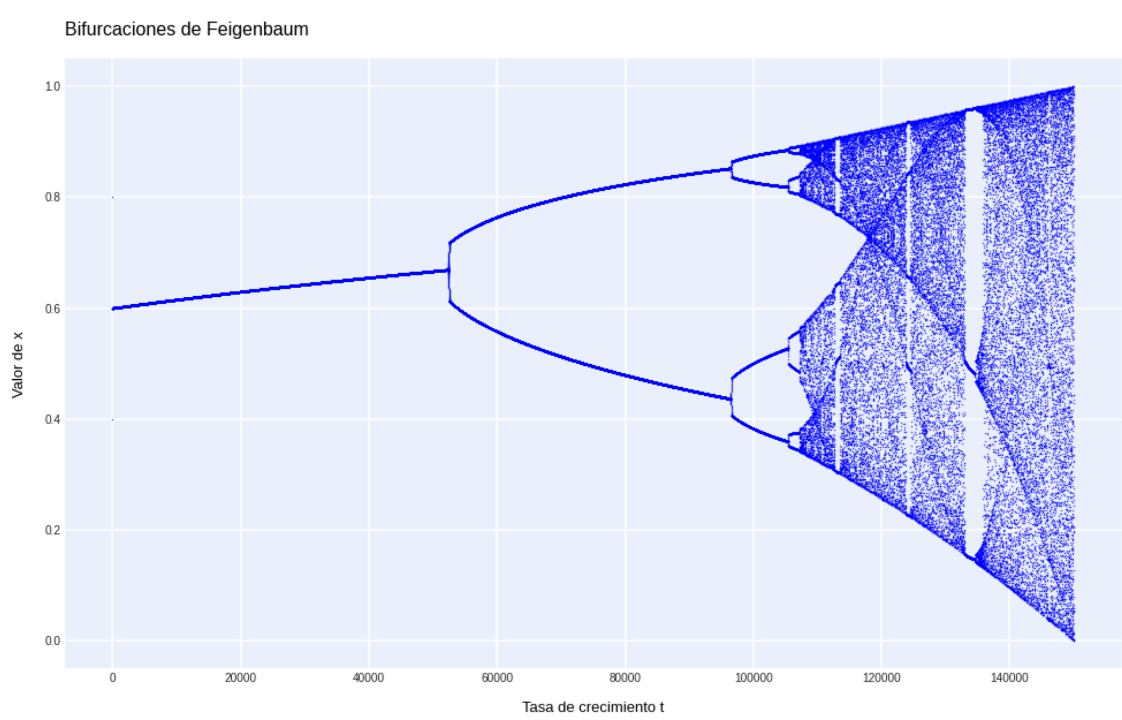
```
    pio.write_image(figura, file_name, format='pdf')
    figura.show()
```

[124]:
```
def leer_col_csv(file_path, column_name):
    data = pd.read_csv(file_path)
    column_data = data[column_name]
    return np.array(column_data)
```

1 Bifurcación de Feigenbaum

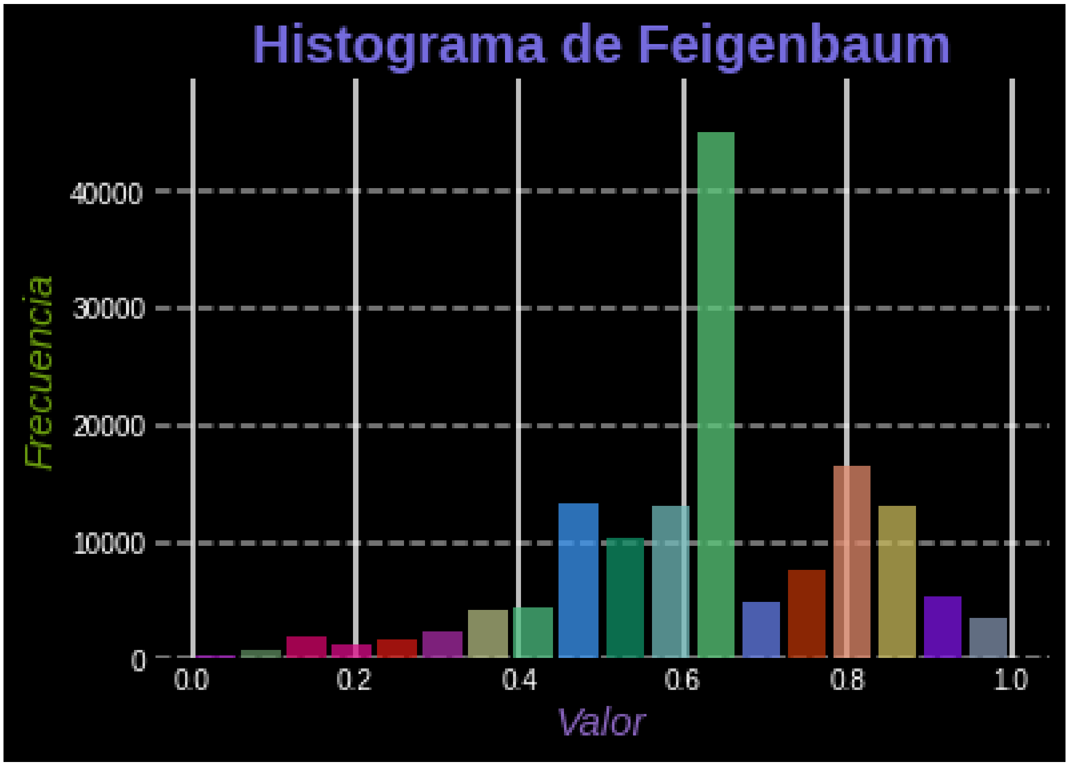
[125]:
```
valores_x = leer_col_csv("datosFeigenbaum.csv", "Valores x")
valores_k = range(1, len(valores_x)+1)
graficar(valores_x, valores_k, width=10, height=7 ,titulo="Bifurcaciones de␣
 ↪Feigenbaum")
```

```
/tmp/ipykernel_14152/3848508717.py:5: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no
longer correspond to the styles shipped by seaborn. However, they will remain
available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API
instead.
```



[126]:
```
feigen = DistribucionProbabilidad(valores_x)
feigen.mostrar_metricas()
plotear_hist(valores_x, "Histograma de Feigenbaum", "Valor", "Frecuencia",
  "sturges")
```

```
--- Métricas para Variable 1 ---
Media: 0.6372494639749813
Mediana: 0.6393204627595699
Moda: 0.00021750936713698025
Media Geométrica: 0.6021687887398679
Rango: 0.9997281076145631
Desviación Estándar: 0.17690664403524417
Varianza: 0.03129596070381259
Asimetría: -0.5660847621652911
Curtosis: 0.5445698625294444
Entropia: 11.918390573078392
Coeficiente de Variación: 0.27760971807139817
```



1.1 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.

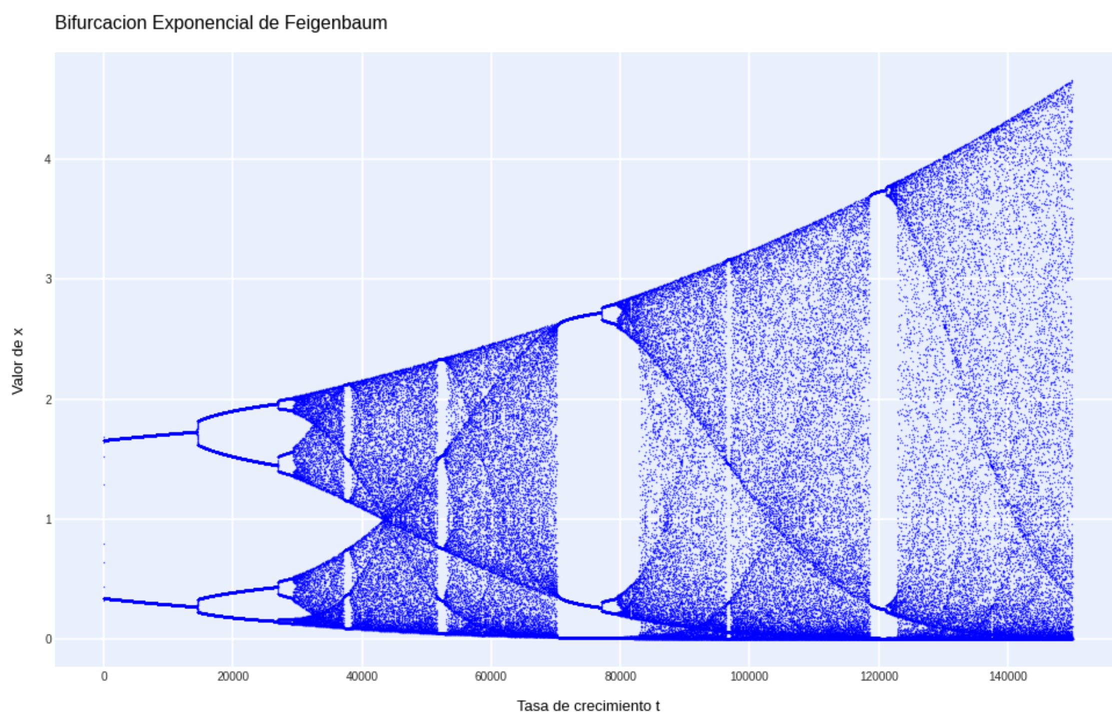
1.1.1 Conclusión

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.

2 Bifurcación Exponencial de Feigenbaum

[127]:
```
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")
```

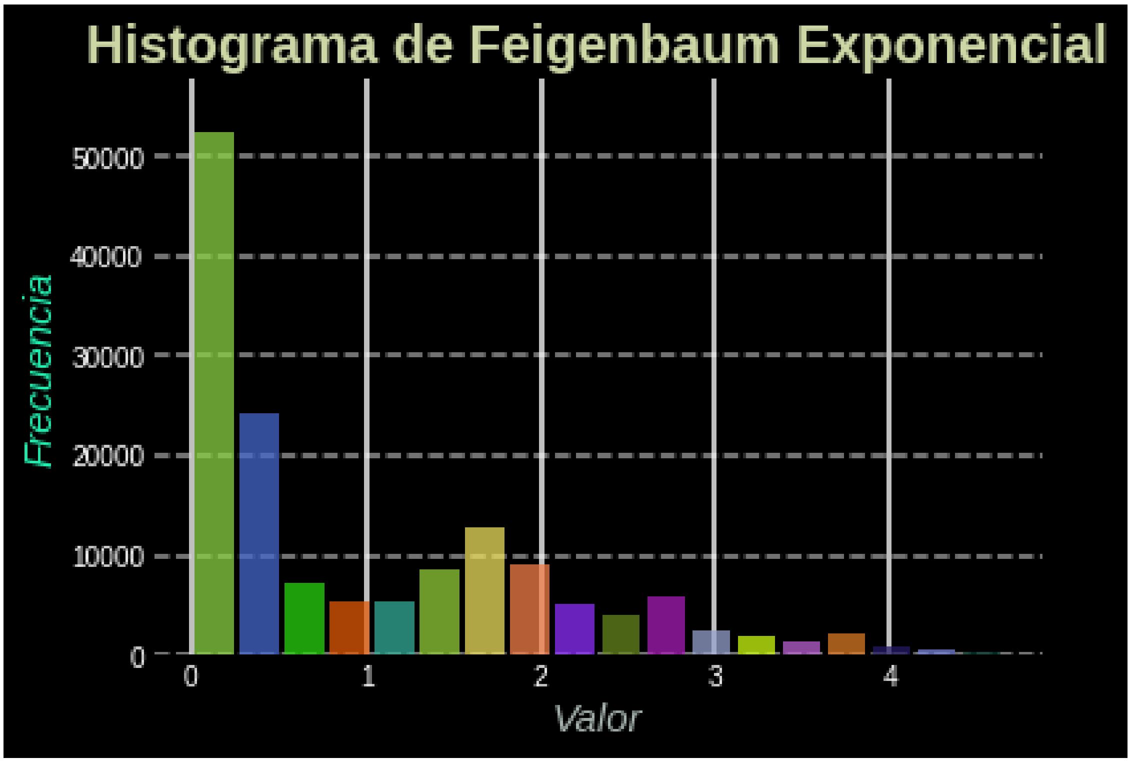
```
/tmp/ipykernel_14152/3848508717.py:5: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no
longer correspond to the styles shipped by seaborn. However, they will remain
available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API
instead.
```



[128]:
```
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: 1.0000072604870167
Mediana: 0.4721920834067132
Moda: 2.985814642355891e-06
Media Geométrica: 0.27303533280453596
```

```
Rango: 4.65706017749897
Desviación Estándar: 1.0532967093824084
Varianza: 1.1094339579958097
Asimetría: 0.9807630688401133
Curtosis: 0.06335793515428323
Entropia: 11.918390573078392
Coeficiente de Variación: 1.053289061990849
```



2.1 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.

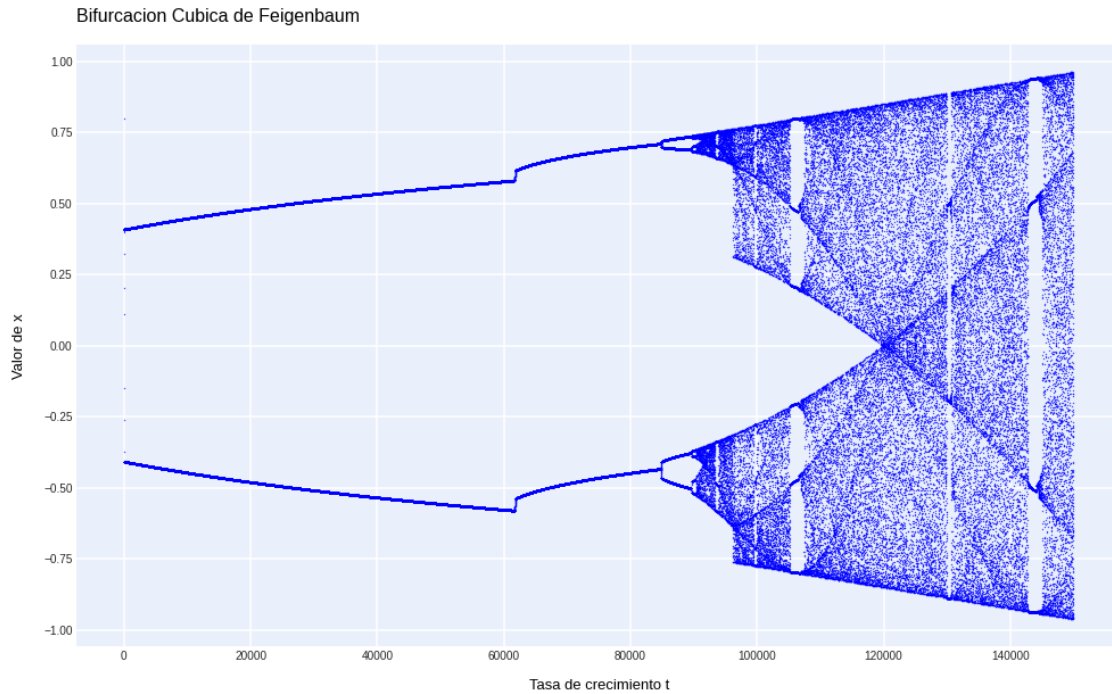
2.1.1 Conclusión

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.

3 Bifurcación Cúbica de Feigenbaum

[129]:
```
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")
```

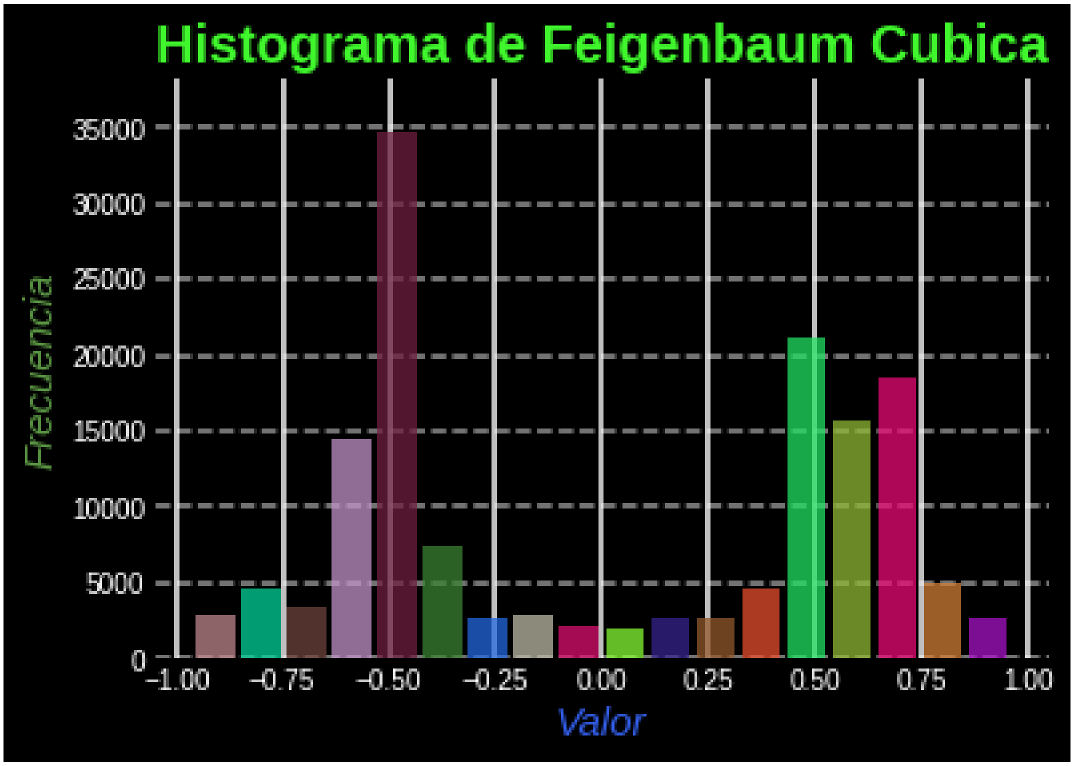
```
/tmp/ipykernel_14152/3848508717.py:5: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no
longer correspond to the styles shipped by seaborn. However, they will remain
available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API
instead.
```



[130]:
```python
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: 0.024430261042410088
Mediana: -0.006403832324115333
Moda: -0.9623880646432766
Media Geométrica: nan
Rango: 1.924159514072628
Desviación Estándar: 0.5557835643122274
Varianza: 0.30889537035960385
Asimetría: 0.01034405617948531
Curtosis: -1.662704225845407
Entropia: 11.918390573078392
Coeficiente de Variación: 22.74980047685149
```

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

invalid value encountered in log
```



3.1 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.

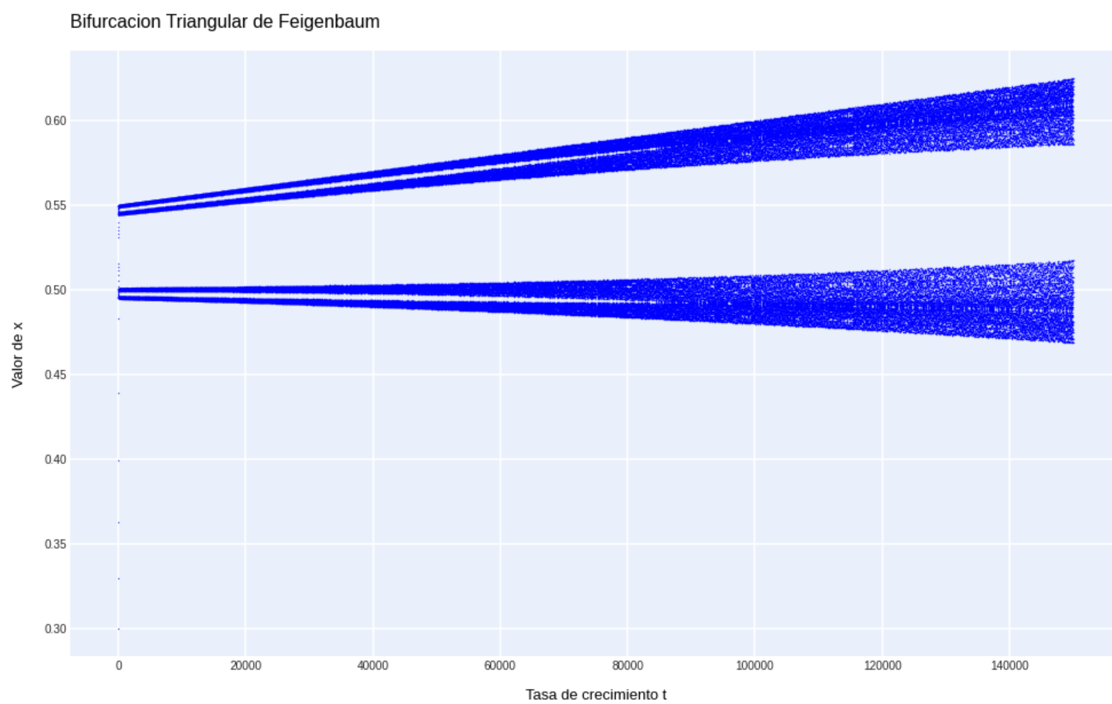
3.1.1 Conclusión

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.

4 Bifurcación Triangular de Feigenbaum

[131]:
```
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")
```

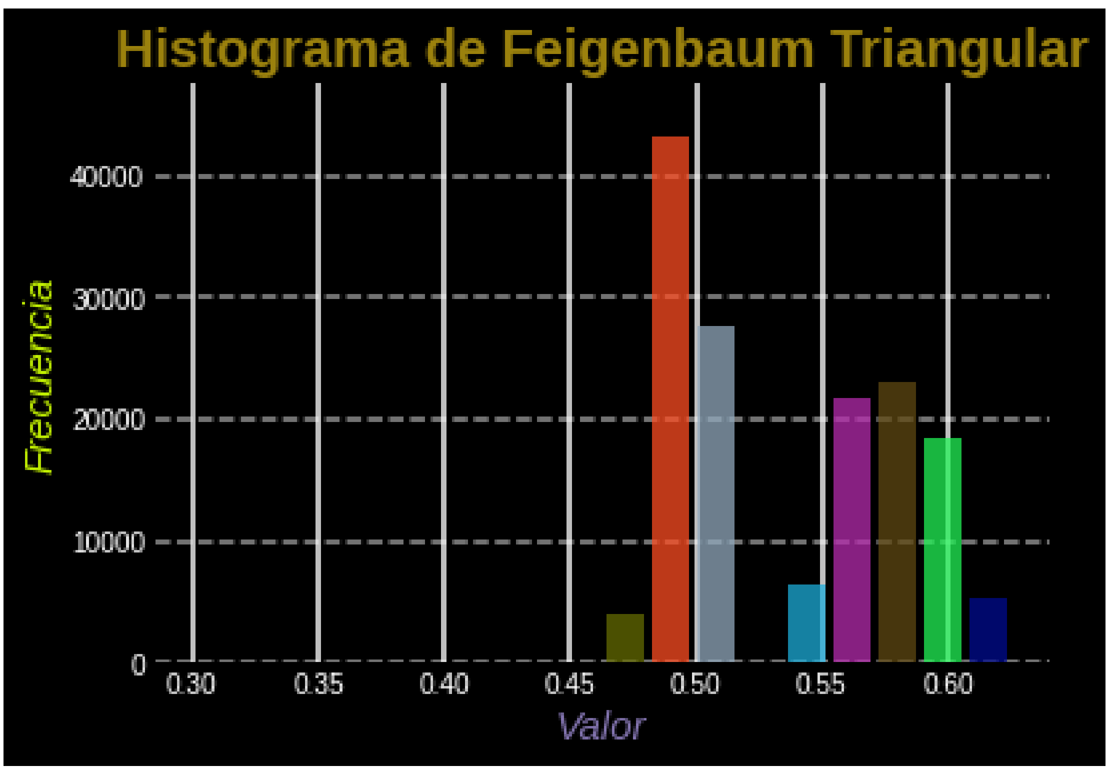
```
/tmp/ipykernel_14152/3848508717.py:5: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no
longer correspond to the styles shipped by seaborn. However, they will remain
available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API
instead.
```



[132]:

```
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: 0.5364884148803621
Mediana: 0.5173285010385174
Moda: 0.3
Media Geométrica: 0.5346609303242754
Rango: 0.3249136266819744
Desviación Estándar: 0.044466168428566745
Varianza: 0.001977240134717666
Asimetría: 0.20551589089020408
Curtosis: -1.5760468857707368
Entropia: 11.918390573078392
Coeficiente de Variación: 0.08288374398258494
```



4.1 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.

4.1.1 Conclusión

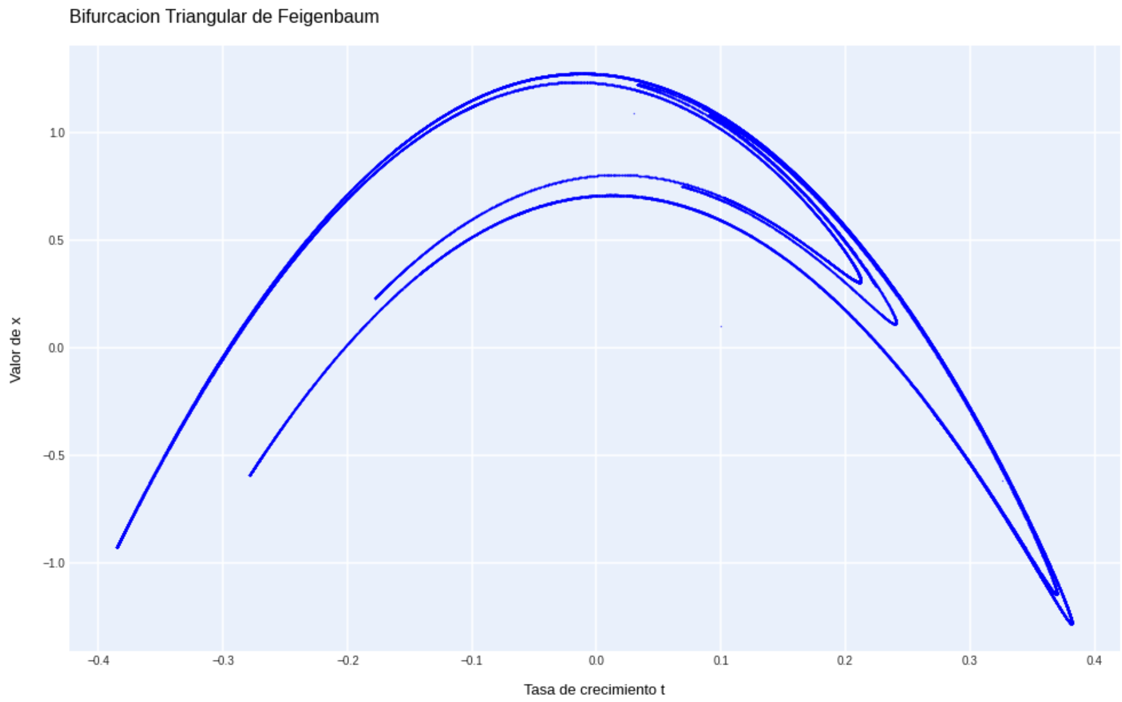
El modelo triangular de Feigenbaum muestra una interesante distribución de datos con métricas que indican una dinámica tanto concentrada como diversificada. La alta entropía junto con una curtosis negativa y asimetría positiva sugiere que, aunque el sistema es predominantemente estable en torno a ciertos 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.

5 Mapa de Henon

[133]:
```
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")
```

```
/tmp/ipykernel_14152/3848508717.py:5: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no
longer correspond to the styles shipped by seaborn. However, they will remain
```

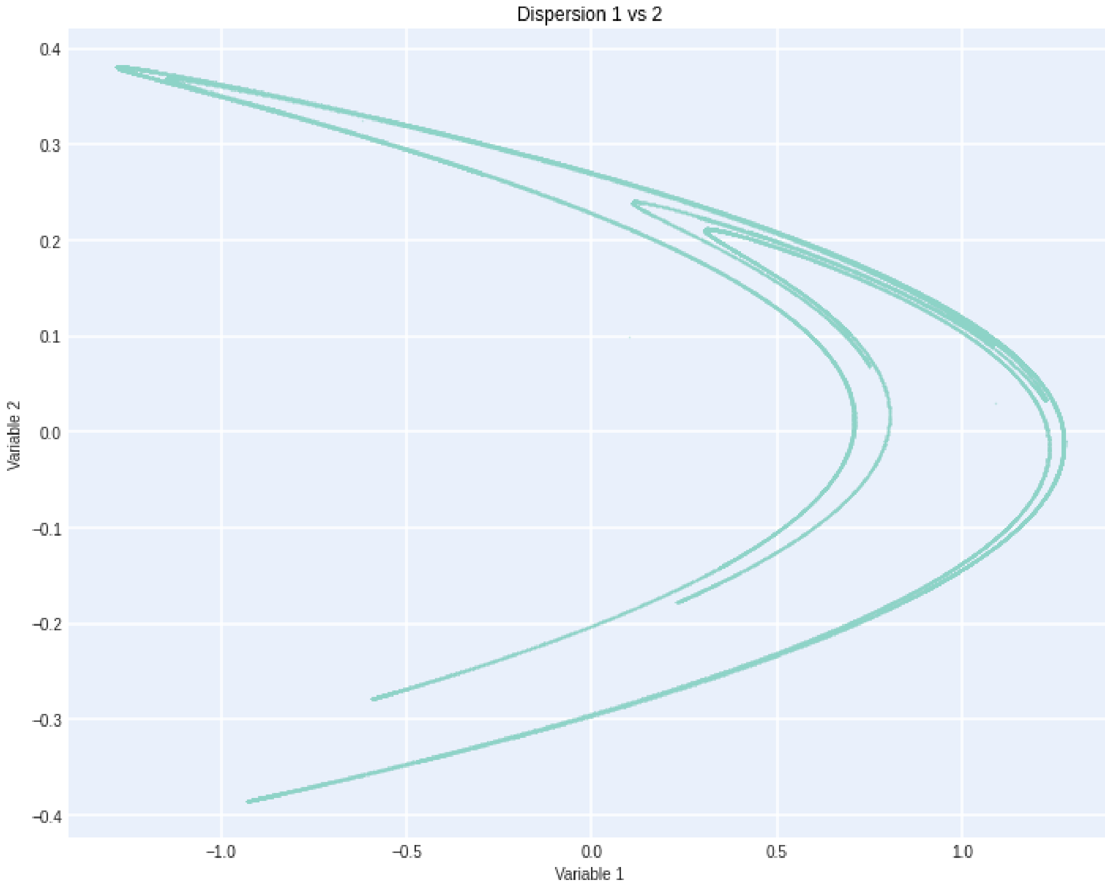
```
available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API
instead.
```

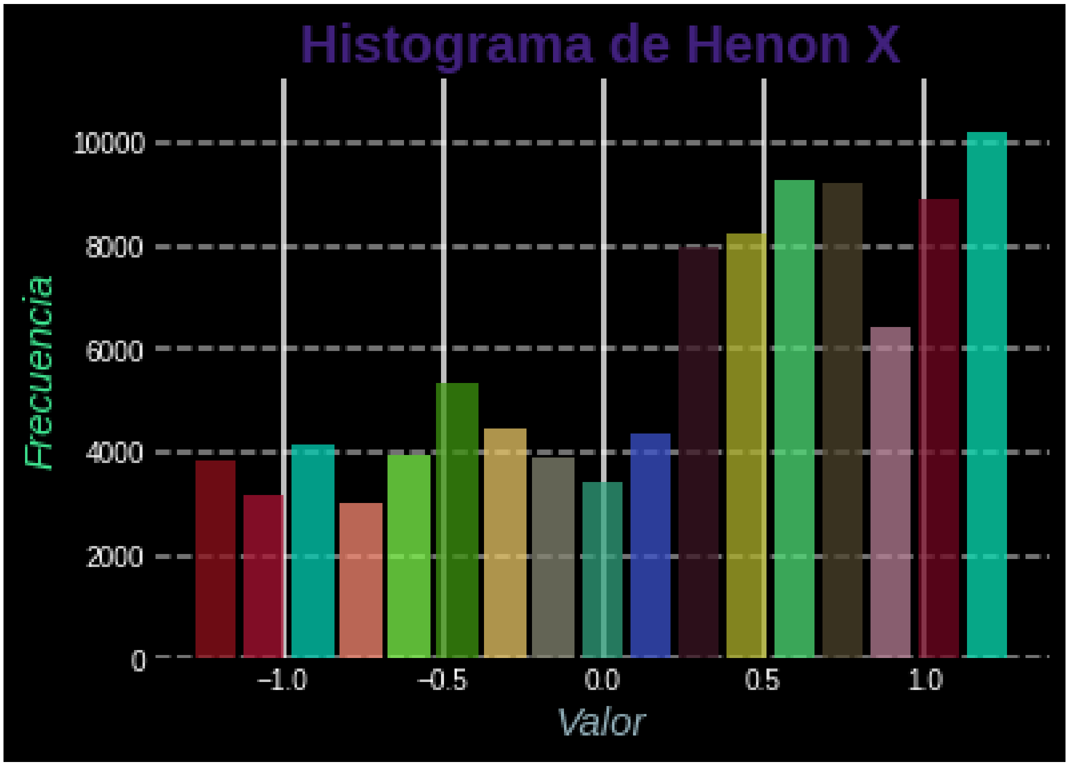


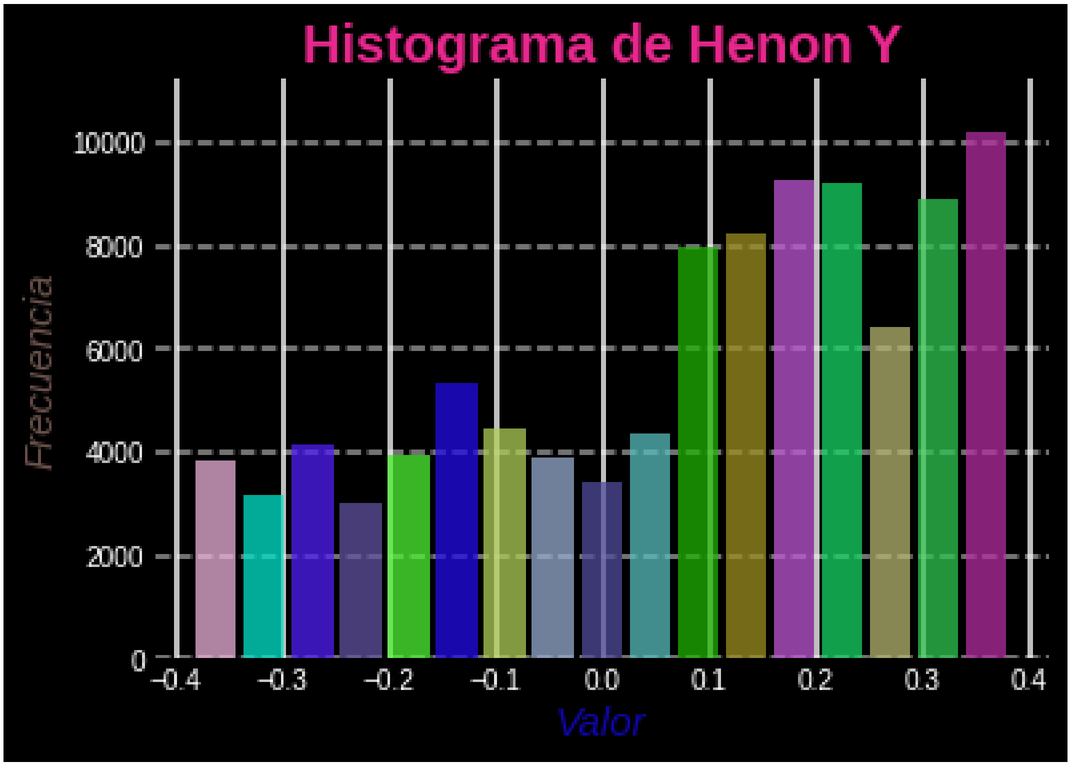
[134]:
```python
henon = DistribucionProbabilidad(valores_x, valores_y)
henon.mostrar_metricas()
plotear_hist(valores_x, "Histograma de Henon X", "Valor", "Frecuencia",␣
↪"sturges")
plotear_hist(valores_y, "Histograma de Henon Y", "Valor", "Frecuencia",␣
↪"sturges")
graficar(valores_x, valores_k, width=10, height=7 ,titulo="Variable X vs k")
graficar(valores_y, valores_k, width=10, height=7 ,titulo="Variable y vs k")
```

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

invalid value encountered in log


--- Métricas para Variable 1 ---
Media: 0.2569448362628383
Mediana: 0.41098282457162805
Moda: -1.284642106388292
Media Geométrica: nan
Rango: 2.557614886272998
```

```
Desviación Estándar: 0.7209690694596401
Varianza: 0.5197963991174993
Asimetría: -0.4942953556145772
Curtosis: -0.8810956783886894
Entropia: 11.512925464970223
Coeficiente de Variación: 2.805929396931466

--- Métricas para Variable 2 ---
Media: 0.07708251422357397
Mediana: 0.12328727549522578
Moda: -0.3853926319164876
Media Geométrica: nan
Rango: 0.7672844658818994
Desviación Estándar: 0.21629041878055238
Varianza: 0.046781545256266724
Asimetría: -0.4942859889221436
Curtosis: -0.8810932469508241
Entropia: 11.512925464970223
Coeficiente de Variación: 2.8059595740898238

Matriz de Correlación:
[[ 1.         -0.31503984]
 [-0.31503984  1.        ]]
```

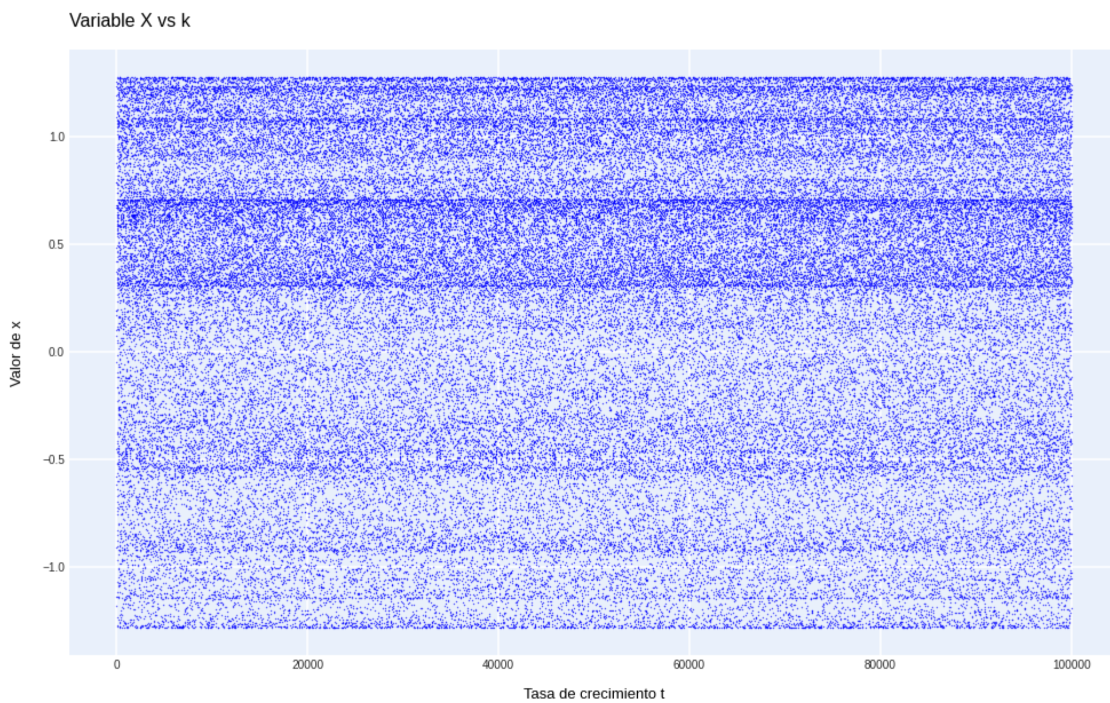

Dispersion 1 vs 2
0.4
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.4
Variable 2
−1.0
−0.5
0.0
0.5
1.0
Variable 1


Histograma de Henon X
Frecuencia
0
2000
4000
6000
8000
10000
−1.0
−0.5
0.0
0.5
1.0
Valor


Histograma de Henon Y
Frecuencia
0
2000
4000
6000
8000
10000
−0.4
−0.3
−0.2
−0.1
0.0
0.1
0.2
0.3
0.4
Valor

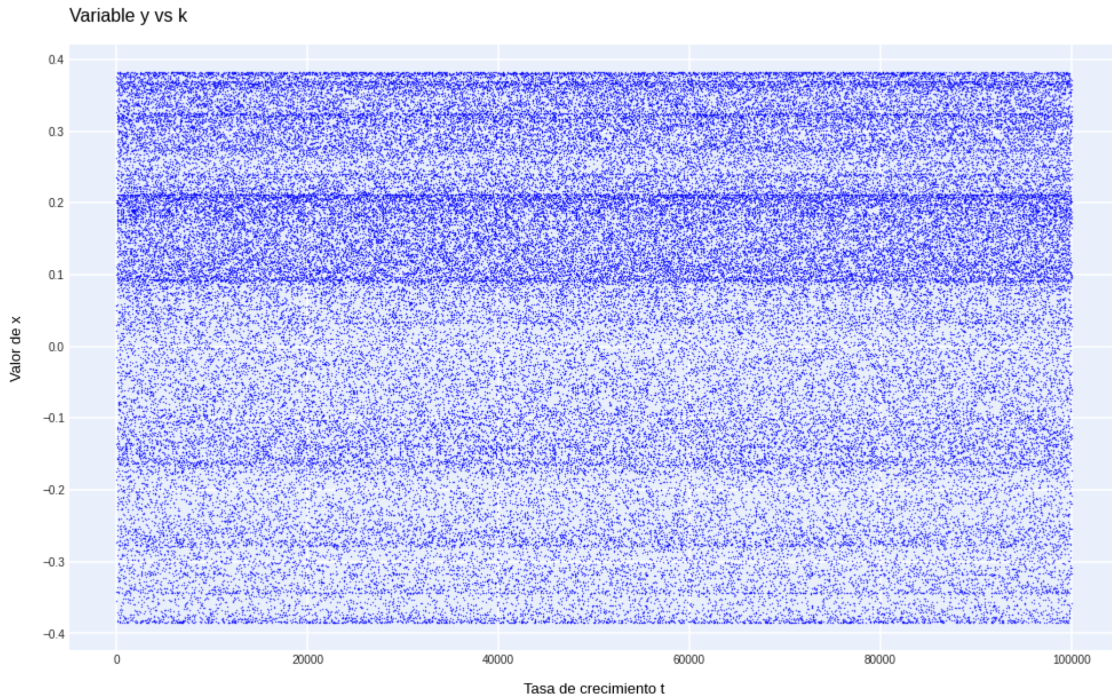
/tmp/ipykernel_14152/3848508717.py:5: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no longer correspond to the styles shipped by seaborn. However, they will remain available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API instead.



/tmp/ipykernel_14152/3848508717.py:5: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no longer correspond to the styles shipped by seaborn. However, they will remain available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API instead.



5.1 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.

5.1.1 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.

5.1.2 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.

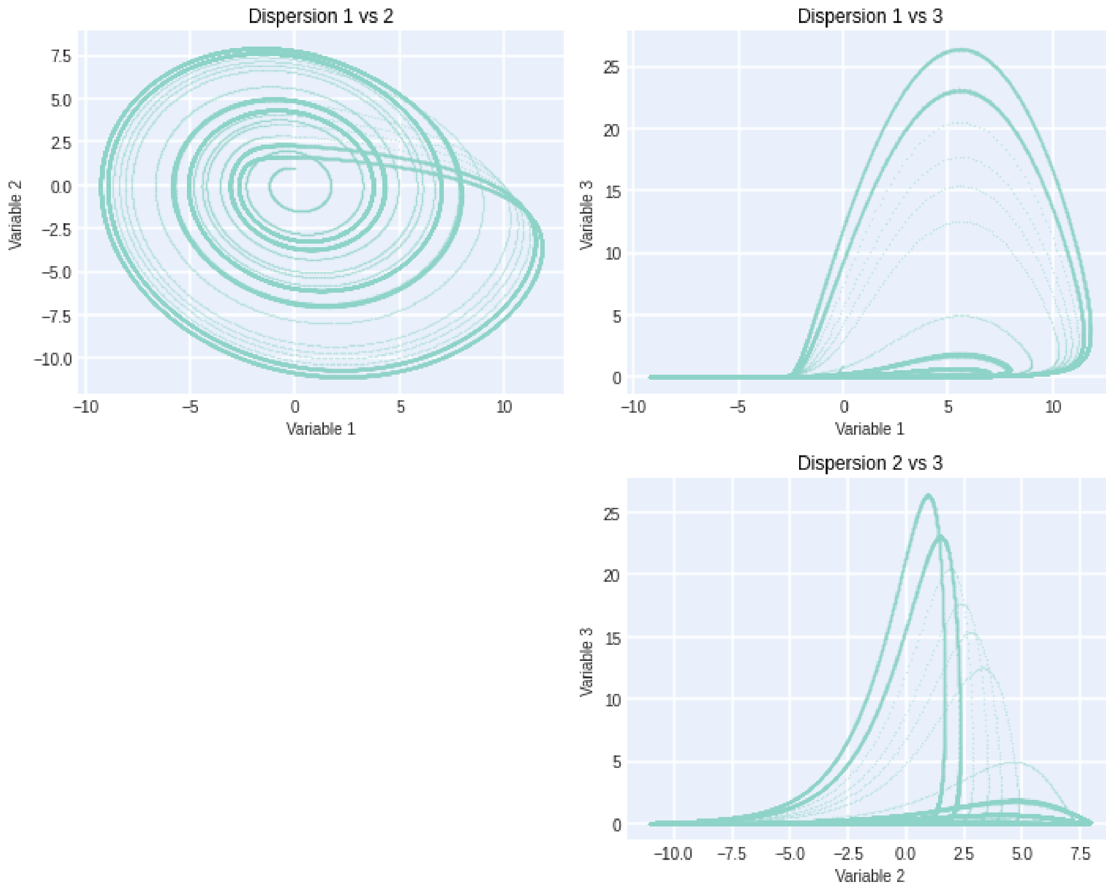
6 Atractor de Rossler

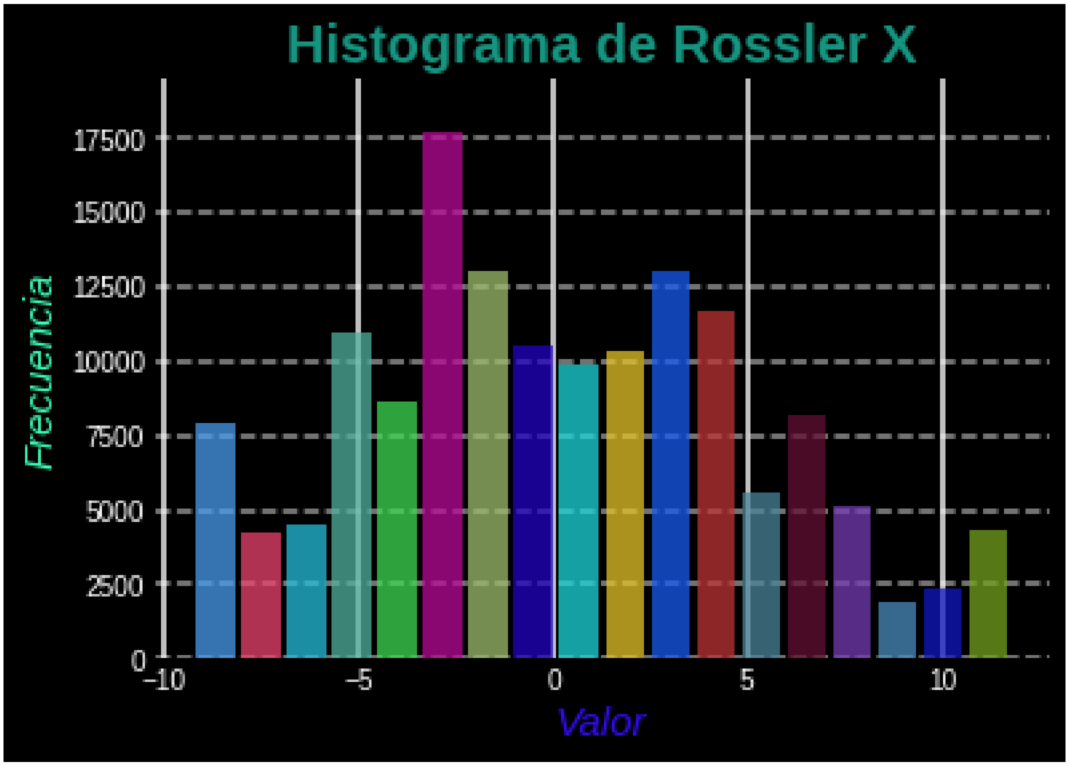
[135]:
```
valores_x = leer_col_csv("datosRossler.csv", "Valores x")
valores_y = leer_col_csv("datosRossler.csv", "Valores y")
valores_z = leer_col_csv("datosRossler.csv", "Valores z")
valores_k = range(1, len(valores_x)+1)
graficar_3d(valores_x, valores_y, valores_z, titulo="Atractor de Rossler")
```

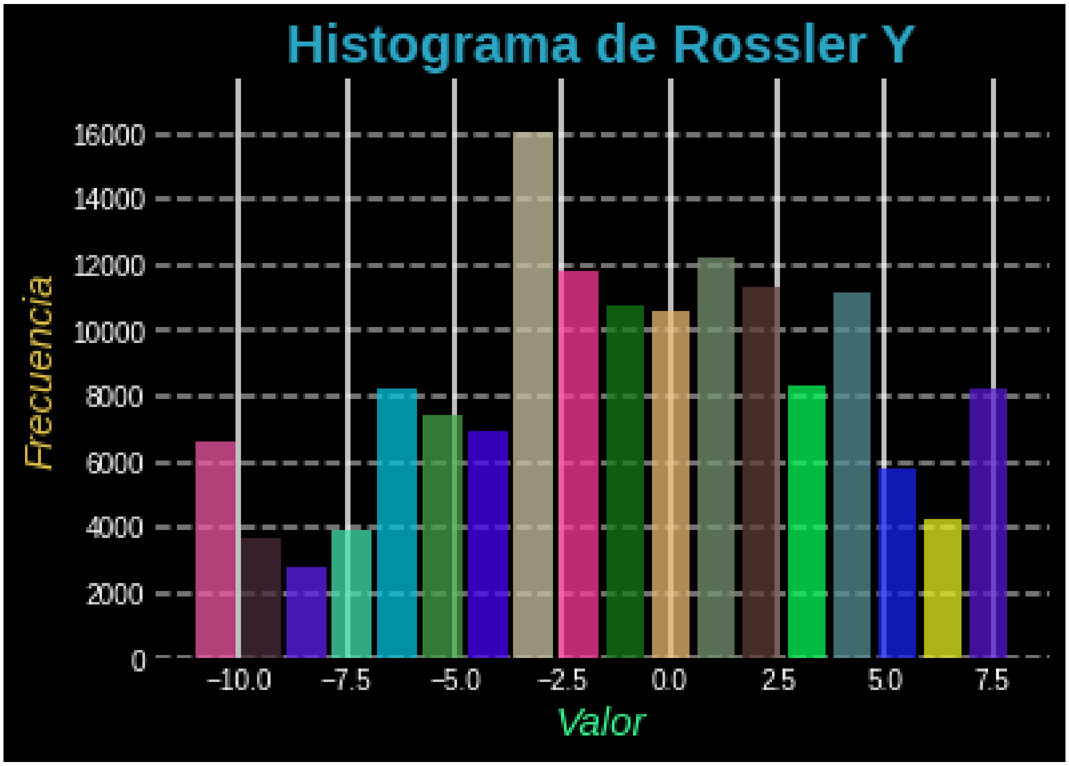
[136]:
```
rossler = DistribucionProbabilidad(valores_x, valores_y, valores_z)
rossler.mostrar_metricas()
plotear_hist(valores_x, "Histograma de Rossler X", "Valor", "Frecuencia",␣
 ↪"sturges")
plotear_hist(valores_y, "Histograma de Rossler Y", "Valor", "Frecuencia",␣
 ↪"sturges")
plotear_hist(valores_z, "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")
```

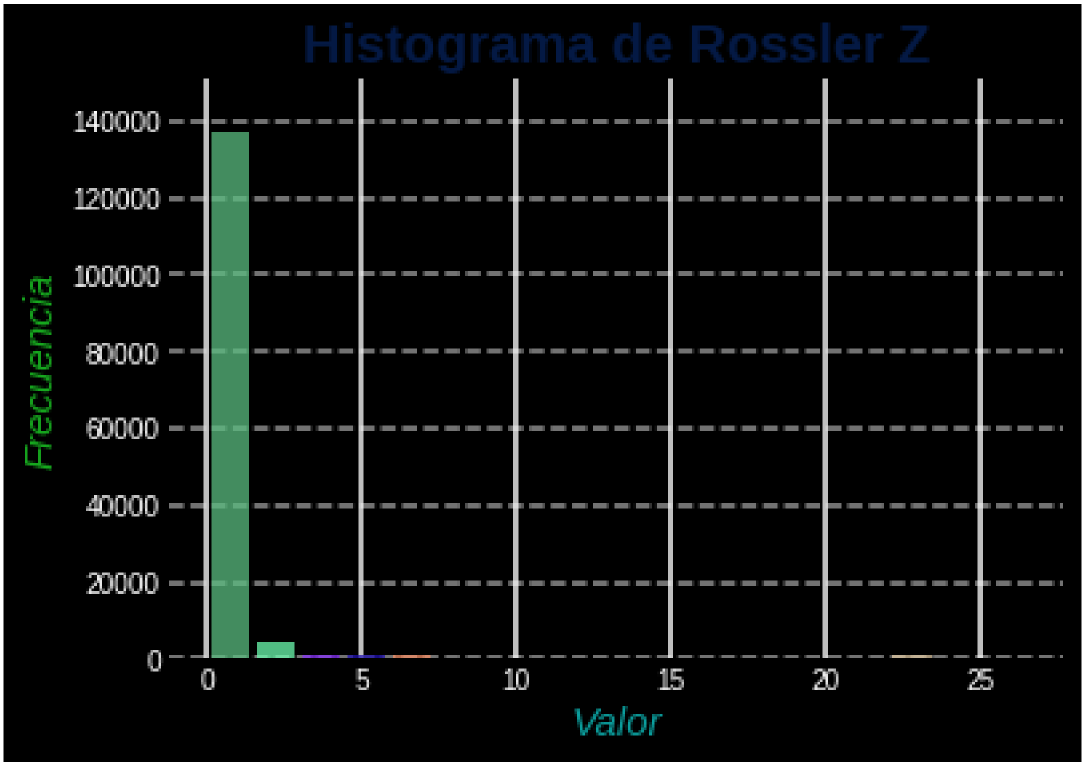
```
--- Métricas para Variable 1 ---
Media: 0.18563944846053237
Mediana: -0.21249560937774703
Moda: -9.28112168747234
Media Geométrica: nan
Rango: 21.07306785563107
Desviación Estándar: 5.001686224312171
Varianza: 25.016865086474144
Asimetría: 0.20337137757016072
Curtosis: -0.5941548063540214
Entropia: 11.918390573078392
Coeficiente de Variación: 26.94301381408999
```

```
--- Métricas para Variable 2 ---
Media: -0.9060905477200308
Mediana: -0.8399406284596411
Moda: -11.090572417946863
Media Geométrica: nan
Rango: 19.023502319928667
Desviación Estándar: 4.751307088682195
Varianza: 22.574919050961675
Asimetría: -0.171230127674978
Curtosis: -0.6627299121069101
Entropia: 11.918390573078392
Coeficiente de Variación: -5.243744237965808

--- Métricas para Variable 3 ---
Media: 0.9068813180616333
Mediana: 0.04095521477796735
Moda: 0.013365904330161932
Media Geométrica: 0.07616044427889065
Rango: 26.4706830221461
Desviación Estándar: 3.49222187173087
Varianza: 12.195613601395461
Asimetría: 5.2065651642010815
Curtosis: 27.850758581336255
Entropia: 11.918390573078392
Coeficiente de Variación: 3.8508036301763715

Matriz de Correlación:
[[ 1.         -0.19408196  0.27481984]
 [-0.19408196  1.          0.08902033]
 [ 0.27481984  0.08902033  1.        ]]

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

invalid value encountered in log
```

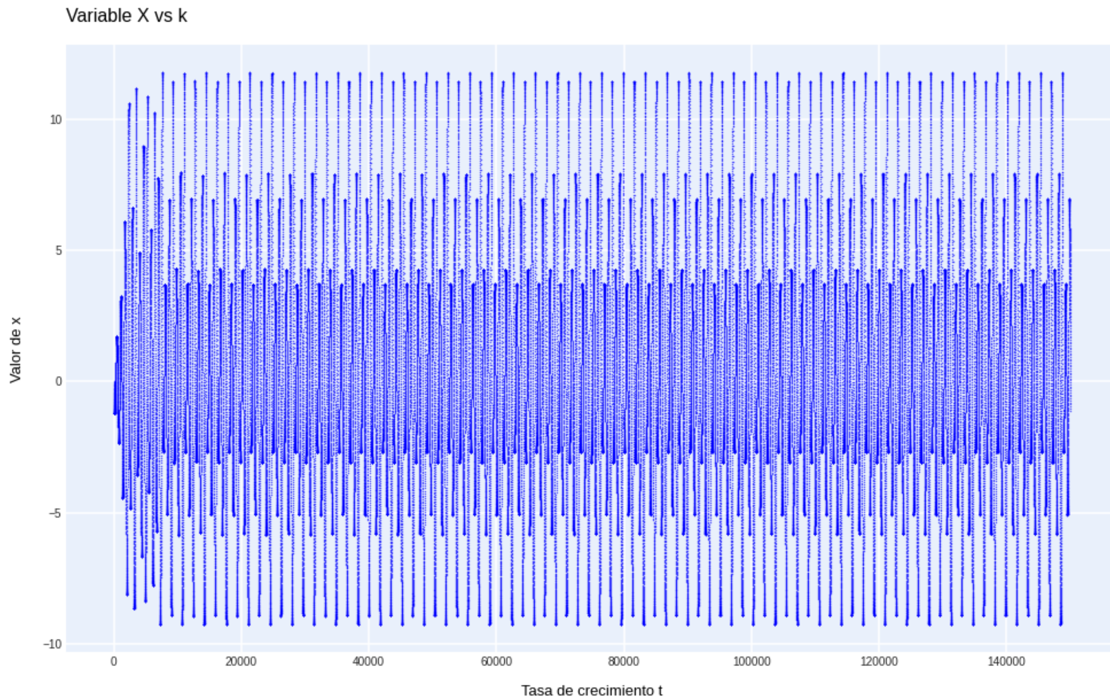

Dispersion 1 vs 2
Variable 1
Variable 2
Dispersion 1 vs 3
Variable 1
Variable 3
Dispersion 2 vs 3
Variable 2
Variable 3


Histograma de Rossler X
Frecuencia
Valor
17500
15000
12500
10000
7500
5000
2500
0
−10
−5
0
5
10

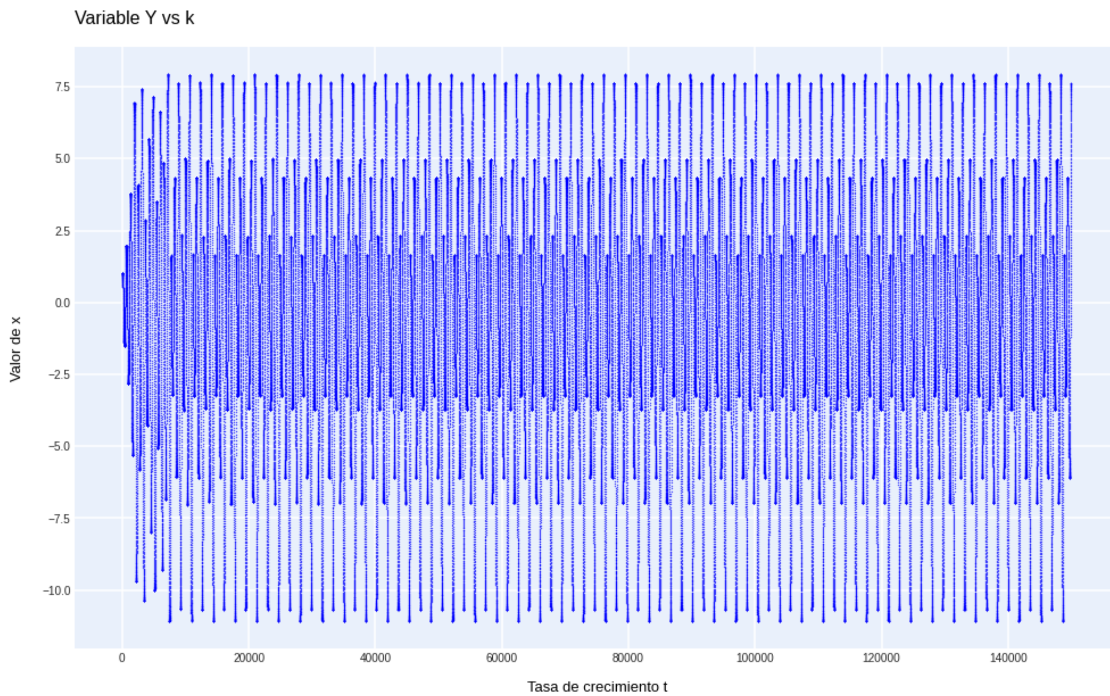

Histograma de Rossler Y
Frecuencia
Valor
16000
14000
12000
10000
8000
6000
4000
2000
0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5



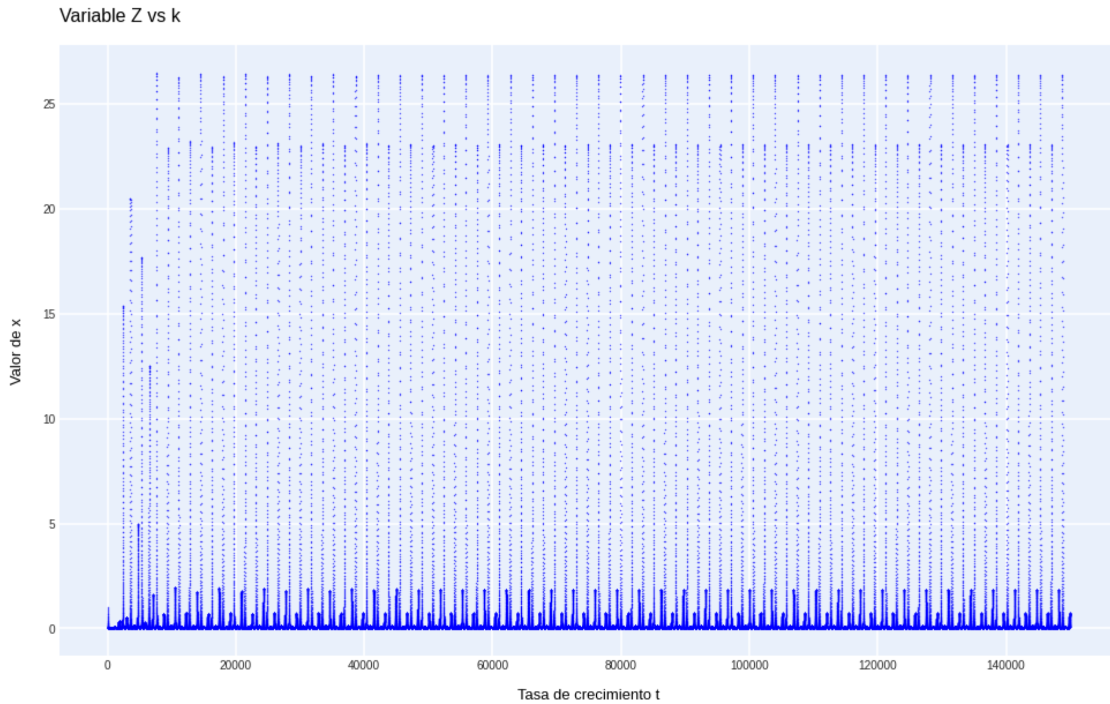

```
/tmp/ipykernel_14152/3848508717.py:5: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no
longer correspond to the styles shipped by seaborn. However, they will remain
available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API
instead.
```




```
/tmp/ipykernel_14152/3848508717.py:5: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no
longer correspond to the styles shipped by seaborn. However, they will remain
available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API
instead.
```




```
/tmp/ipykernel_14152/3848508717.py:5: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no
longer correspond to the styles shipped by seaborn. However, they will remain
available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API
instead.
```



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

6.1.1 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.

6.1.2 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.

6.1.3 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.

6.1.4 Atractor de Lorentz

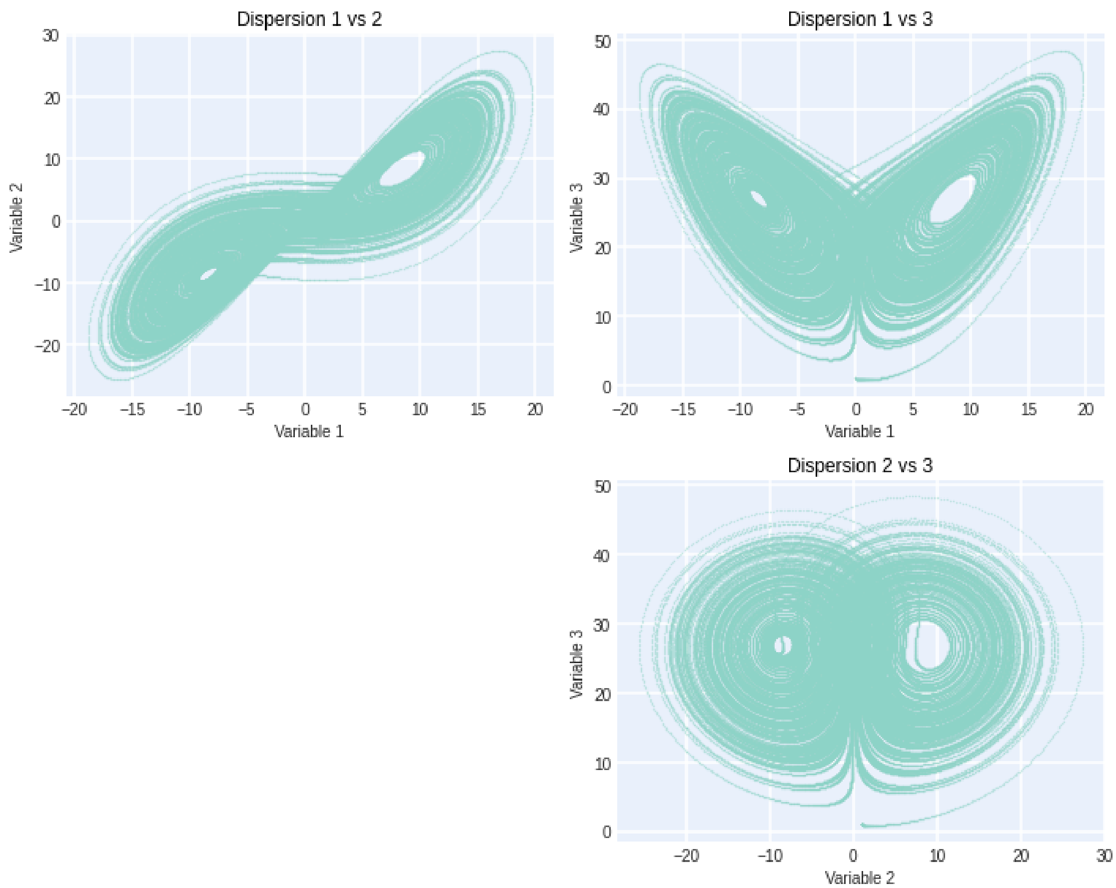
[137]:
```python
valores_x = leer_col_csv("datosLorentz.csv", "Valores x")
valores_y = leer_col_csv("datosLorentz.csv", "Valores y")
valores_z = leer_col_csv("datosLorentz.csv", "Valores z")
valores_k = range(1, len(valores_x)+1)
graficar_3d(valores_x, valores_y, valores_z, titulo="Atractor de Lorentz")
```

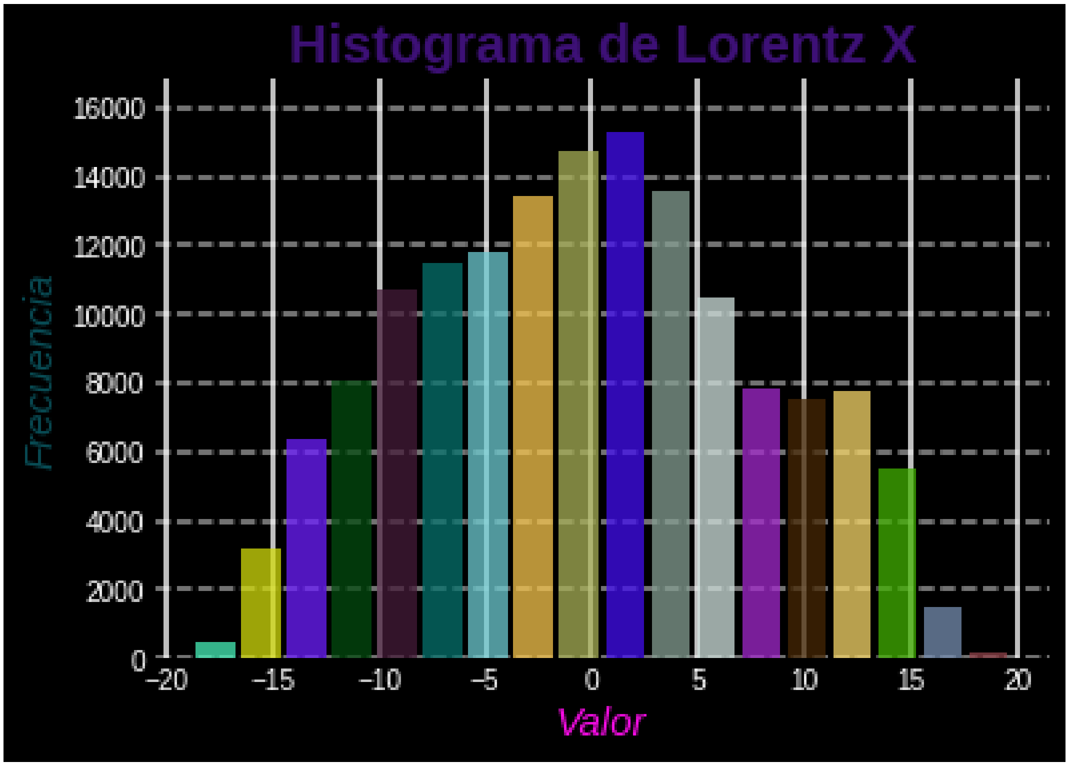
[138]:
```python
lorentz = DistribucionProbabilidad(valores_x, valores_y, valores_z)
lorentz.mostrar_metricas()
plotear_hist(valores_x, "Histograma de Lorentz X", "Valor", "Frecuencia",␣
 ↪"sturges")
plotear_hist(valores_y, "Histograma de Lorentz Y", "Valor", "Frecuencia",␣
 ↪"sturges")
plotear_hist(valores_z, "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")
```

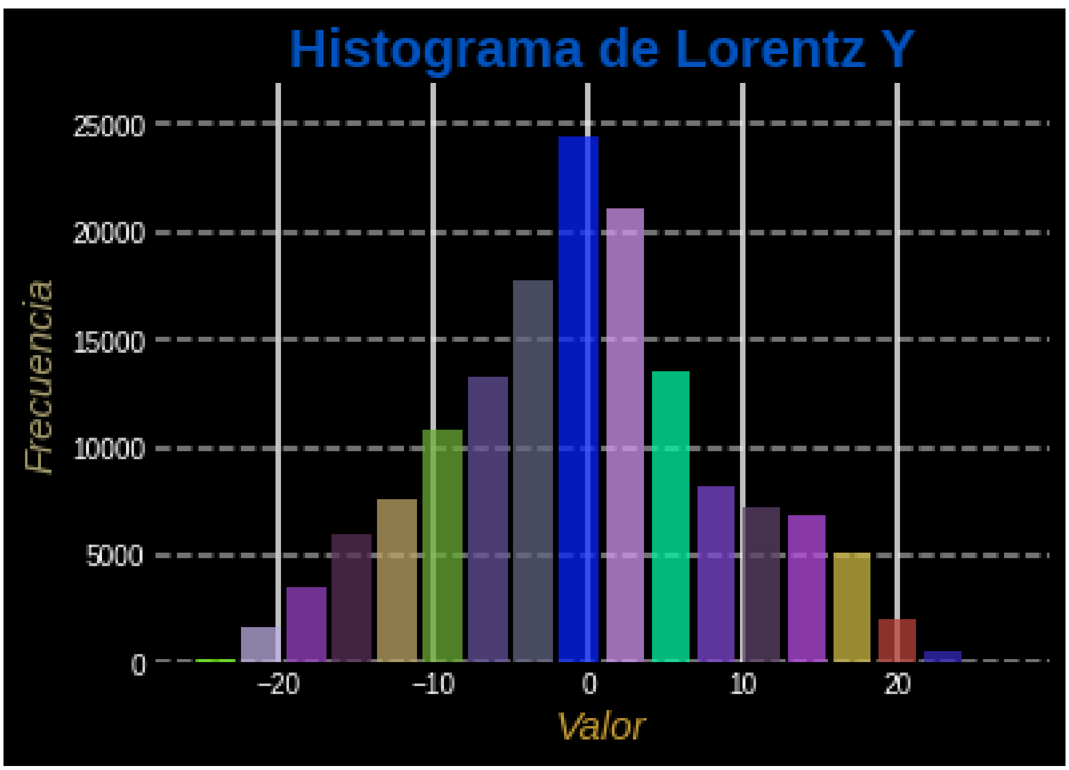
```
--- Métricas para Variable 1 ---
Media: -0.2671314021976278
Mediana: -0.27769278778693524
Moda: -18.81364430507488
Media Geométrica: nan
Rango: 38.56871649435135
Desviación Estándar: 7.982480817842013
Varianza: 63.72000000721568
Asimetría: 0.06974236570151332
Curtosis: -0.7870238232600086
Entropia: 11.918390573078392
Coeficiente de Variación: -29.882225572029355

--- Métricas para Variable 2 ---
Media: -0.27509389023814057
Mediana: -0.3267720075949264
Moda: -25.60801161362229
```

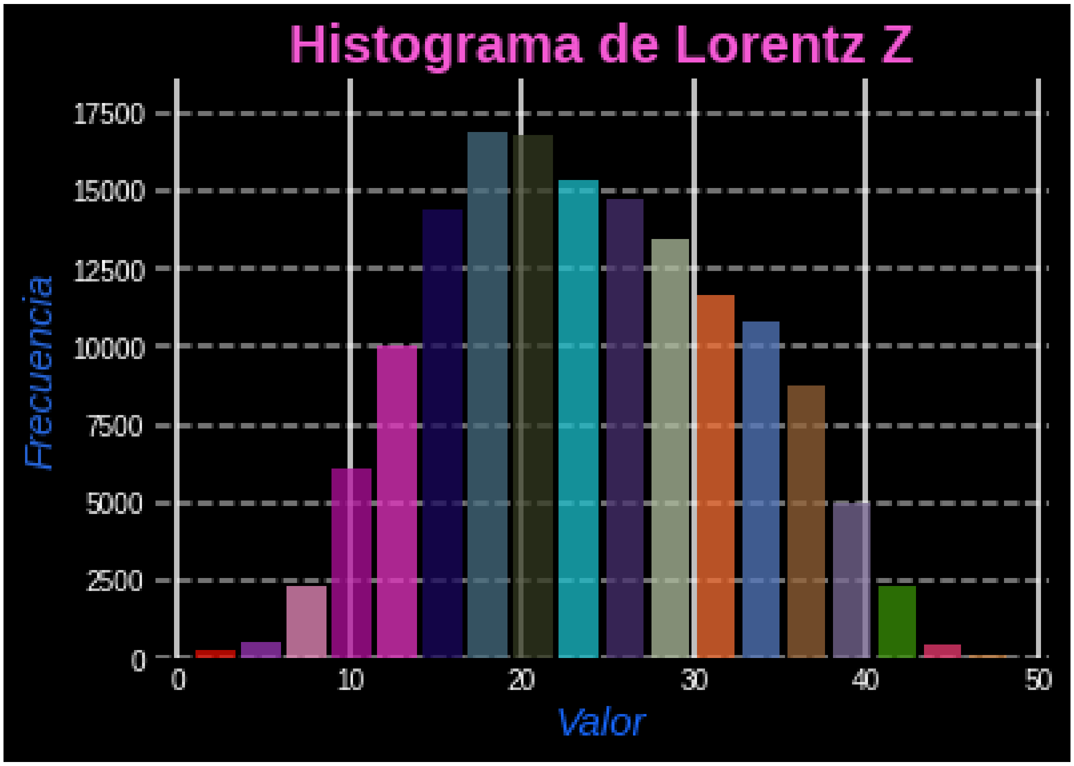
```
Media Geométrica: nan
Rango: 53.02582318365201
Desviación Estándar: 9.031835142896675
Varianza: 81.5740460484634
Asimetría: 0.07708891915776829
Curtosis: -0.24075481925855646
Entropia: 11.918390573078392
Coeficiente de Variación: -32.83182747198815

--- Métricas para Variable 3 ---
Media: 23.82982911699884
Mediana: 23.291729887165992
Moda: 0.8609231341947348
Media Geométrica: 22.133467229861726
Rango: 47.54539290506558
Desviación Estándar: 8.447977085534136
Varianza: 71.36831683770983
Asimetría: 0.1350439513551708
Curtosis: -0.7111785493898726
Entropia: 11.918390573078392
Coeficiente de Variación: 0.354512701037702

Matriz de Correlación:
[[ 1.          0.88320174 -0.01105357]
 [ 0.88320174  1.         -0.01435357]
 [-0.01105357 -0.01435357  1.        ]]
```

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

invalid value encountered in log
```

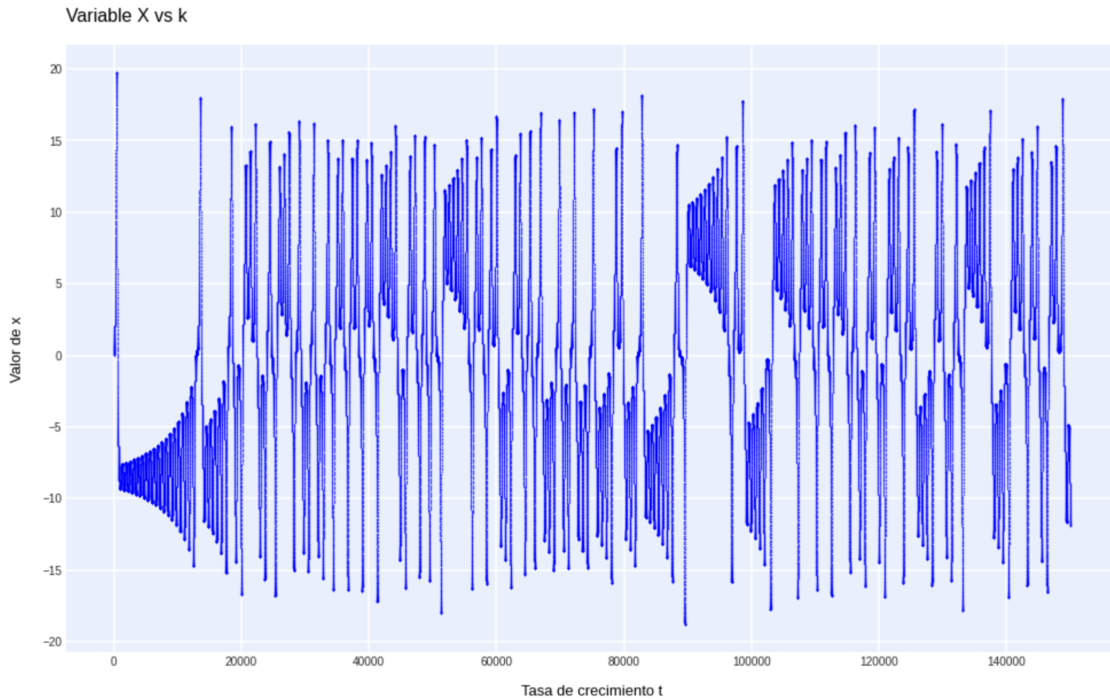

Dispersion 1 vs 2
Variable 1
Variable 2
Dispersion 1 vs 3
Variable 1
Variable 3
Dispersion 2 vs 3
Variable 2
Variable 3


Histograma de Lorentz X
Frecuencia
Valor
16000
14000
12000
10000
8000
6000
4000
2000
0
−20
−15
−10
−5
0
5
10
15
20

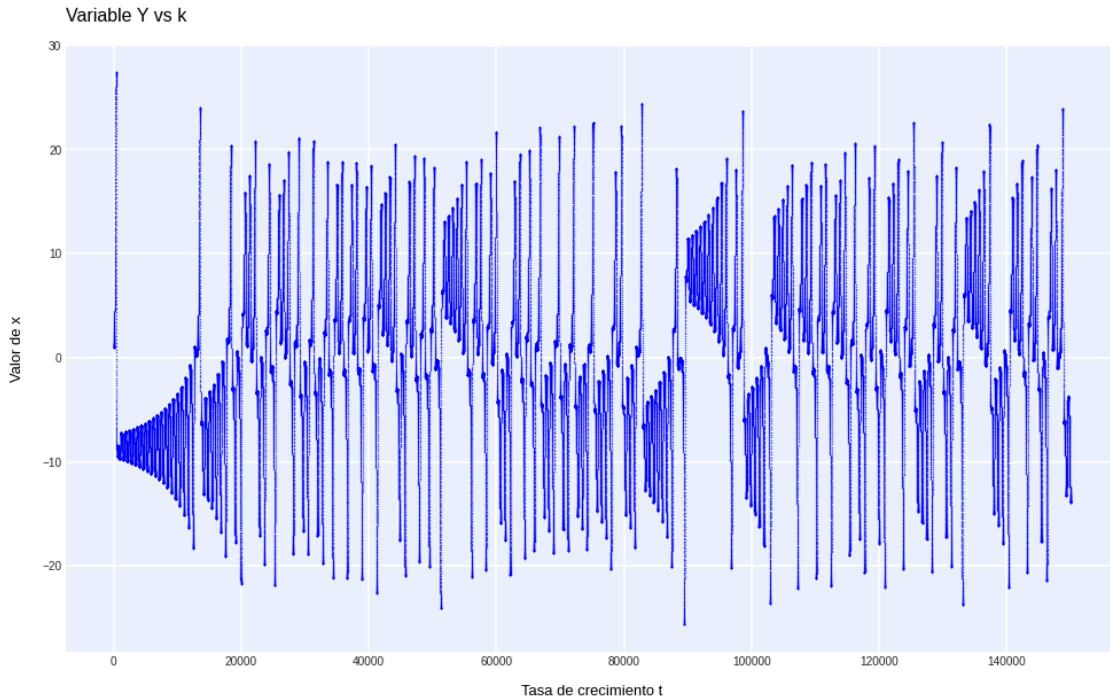

Histograma de Lorentz Y
Frecuencia
Valor
25000
20000
15000
10000
5000
0
−20
−10
0
10
20




```
/tmp/ipykernel_14152/3848508717.py:5: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no
longer correspond to the styles shipped by seaborn. However, they will remain
available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API
instead.
```




```
/tmp/ipykernel_14152/3848508717.py:5: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no
longer correspond to the styles shipped by seaborn. However, they will remain
available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API
instead.
```



/tmp/ipykernel_14152/3848508717.py:5: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no longer correspond to the styles shipped by seaborn. However, they will remain available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API instead.

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

6.2.1 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.

6.2.2 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.

6.2.3 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.