Prácticas de Algorítmica.
3º de Grado en Ingeniería Informática.
Curso 2022-2023.
Práctica 3.

Objetivos.

Con esta práctica se pretende que el alumno implemente dos algoritmos de segmentación de series temporales basados en la técnica de los algoritmos voraces.

Definiciones.

Una serie temporal es una sucesión de observaciones de una variable realizadas a intervalos regulares de tiempo.

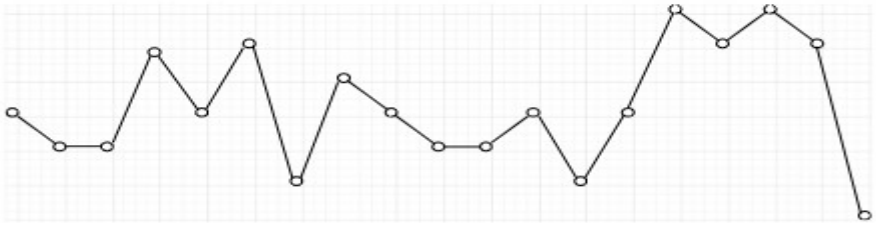

Figura: *Serie temporal*

Una segmentación de la serie temporal es un subconjunto de puntos (puntos dominantes o de corte) que conforman una representación más simple de la misma, conservando la información más relevante.

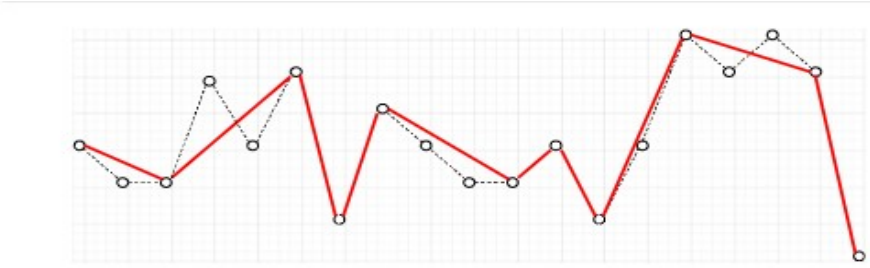

Figura: *Segmentación de una serie temporal*

Como se puede apreciar en la figura, la serie segmentada es una simplificación de la serie original, con lo que supone una compresión de la misma.

Evidentemente, hay diferencias entre la serie segmentada y la original. Existen dos formas de cuantificar esta diferencia de cara a ver la bondad de la segmentación:

- ***Error máximo (eMax)**: Es la máxima distancia, medida en vertical, entre los puntos puntos de la serie y los segmentos rectos que componen la segmentación. En la figura anterior sería la distancia desde el cuarto punto al segmento que queda por debajo (aproximadamente 1.9).*
- ***Suma de los errores al cuadrado (ISE)**: Es la suma de los cuadrados de las distancias en vertical de todos los puntos de la serie a los segmentos que la aproximan.*

Enunciado:

Se han de implementar dos algoritmos de segmentación basados en el método greedy (algoritmos voraces). El programa principal contendrá un menú con dos opciones y se

implementará de forma similar a como se hizo en la primera práctica, es decir, cada opción del menú llamará a una función de medio nivel sin parámetros y en esas funciones de medio nivel se implementarán cada uno de los métodos.

Para ello se suministra la clase SerieTemporal, y las clase Punto y Recta, que son clases auxiliares a la clase SerieTemporal.

La clase SerieTemporal contiene las siguientes funciones miembro:

- *Constructor de la serie a partir de un fichero de puntos. Crea la serie, carga los puntos desde el fichero y los marca como no dominantes.*
- *Constructor de copia a partir de otra serie temporal.*
- *Observador y modificador para devolver o modificar el número de puntos de la serie.*
- *Observador y modificador para devolver o modificar un punto de la serie.*
- *Observador y modificador para ver o asignar un punto de la serie como dominante.*
- *Método para guardar los puntos de la serie temporal en un fichero.*
- *Método para guardar los puntos dominantes de la serie temporal en un fichero.*
- *Método para contar los puntos dominantes de la serie temporal.*
- *Método para calcular la suma de errores cuadráticos (ISE), el error máximo y su ubicación, de la segmentación realizada.*
- *Método para calcular la suma de errores cuadráticos entre dos puntos, suponiendo que estos están unidos mediante un segmento.*
- *Método para calcular el error máximo entre dos puntos y su ubicación, suponiendo que estos están unidos mediante un segmento.*

Método para mostrar por pantalla los puntos dominantes obtenidos en la segmentación.

Algunos de estos métodos a su vez, hacen uso de los métodos de las clases Punto y Recta.

Los datos de entrada de ambos métodos serán el nombre del fichero que contiene la serie y el número de puntos que contendrá la serie segmentada. Ambos métodos se basan en considerar que en principio todos los puntos de la serie son dominantes, y posteriormente se realizan múltiples iteraciones, de forma tal que en cada iteración se elimina un punto dominante. Las iteraciones terminan cuando el número de puntos dominantes que queden sea igual al número de puntos que ha de tener la serie segmentada. La diferencia entre ambos métodos estriba en la función objetivo que se quiere minimizar

Método 1. Segmentación minimizando ISE:
En este caso se eliminará el punto dominante que genere un menor incremento de ISE cuando sea eliminado. De esta forma estamos intentando minimizar el ISE de la segmentación.

Método 2. Segmentación minimizando eMax:
En este caso se eliminará el punto dominante que genere un menor incremento de eMax cuando sea eliminado. De esta forma estamos intentando minimizar el eMax de la segmentación.

Nota: Se recomienda usar en ambos métodos dos vectores locales, uno para ir almacenando la posición de los puntos dominantes y otro para almacenar el error que se genera si ese punto dominante es borrado.

Comprobación:
Se suministran varios archivos con series temporales, con extensión txt. Si se prueba el

archivo BBVA.txt con un número de puntos de la serie segmentada igual a 62 han de salir los siguientes resultados:

Método 1.

- *ISE = 791.407*
- *errorMaximo= 2.17721*
- *puntoErrorMaximo = 2568*

Método 2.

- *ISE = 1298.45*
- *errorMaximo= 1.53319*
- *puntoErrorMaximo = 2401*

Otras series temporales suministradas:

Arrhytmia.txt, probad con 236 puntos.
Método 1.
ISE = 8.81826
error máximo: 0.16 en el punto 7692

Método 2.
ISE = 11.8344
error máximo: 0.123855 en el punto 1396

b41043.txt, probad con 105 puntos.
Metodo1.
ISE = 1973.76
error máximo: 3.6437 en el punto 5531

Metodo2.
ISE = 2551.52
error máximo: 1.62704 en el punto 5704

HandOutlines, probad con eMax = 183 puntos
Método 1.
ISE = 0.240099
error máximo: 0.0260467 en el punto 3349

Método 2.
ISE = 0.299124
error máximo: 0.0197538 en el punto 1168

StarLightCurves.txt, probad con eMax = 214 puntos.
Método 1.
ISE = 1.97677
error máximo: 0.0715 en el punto 7426

Método 2.
ISE = 2.01341
error máximo: 0.04638 en el punto 2518

Fecha de comienzo: 18 de octubre de 2022.
Fecha máxima de entrega: 9 de noviembre de 2022.