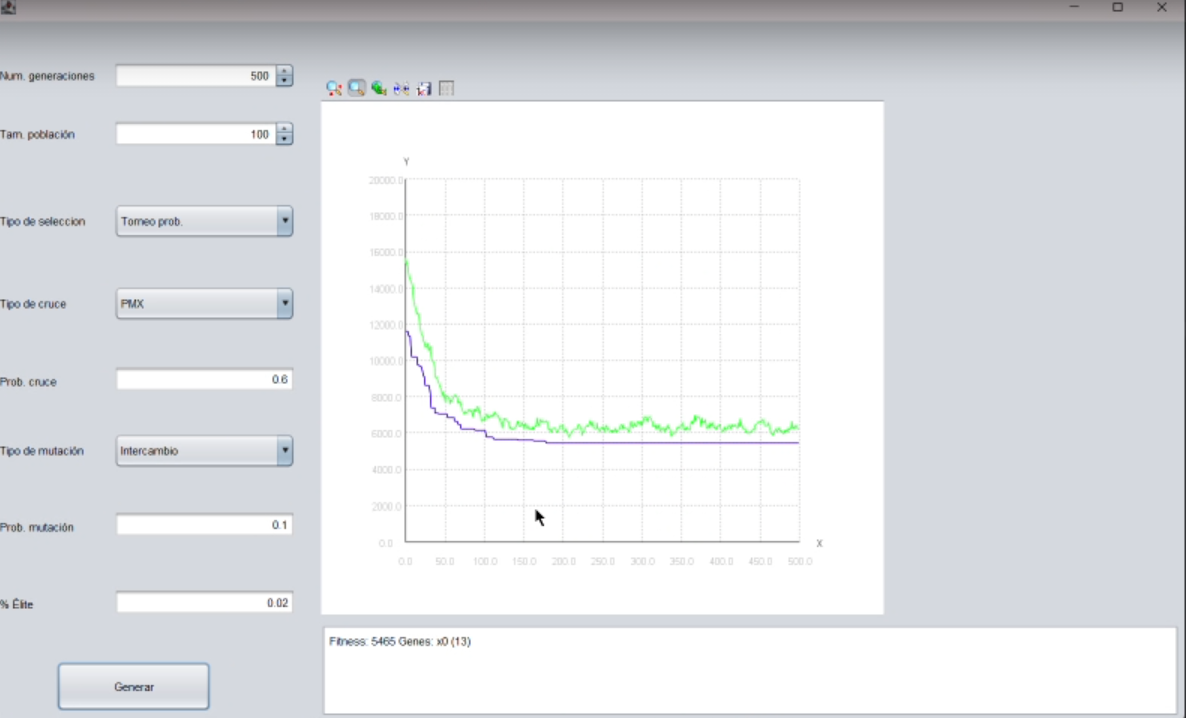
**Grupo 6**

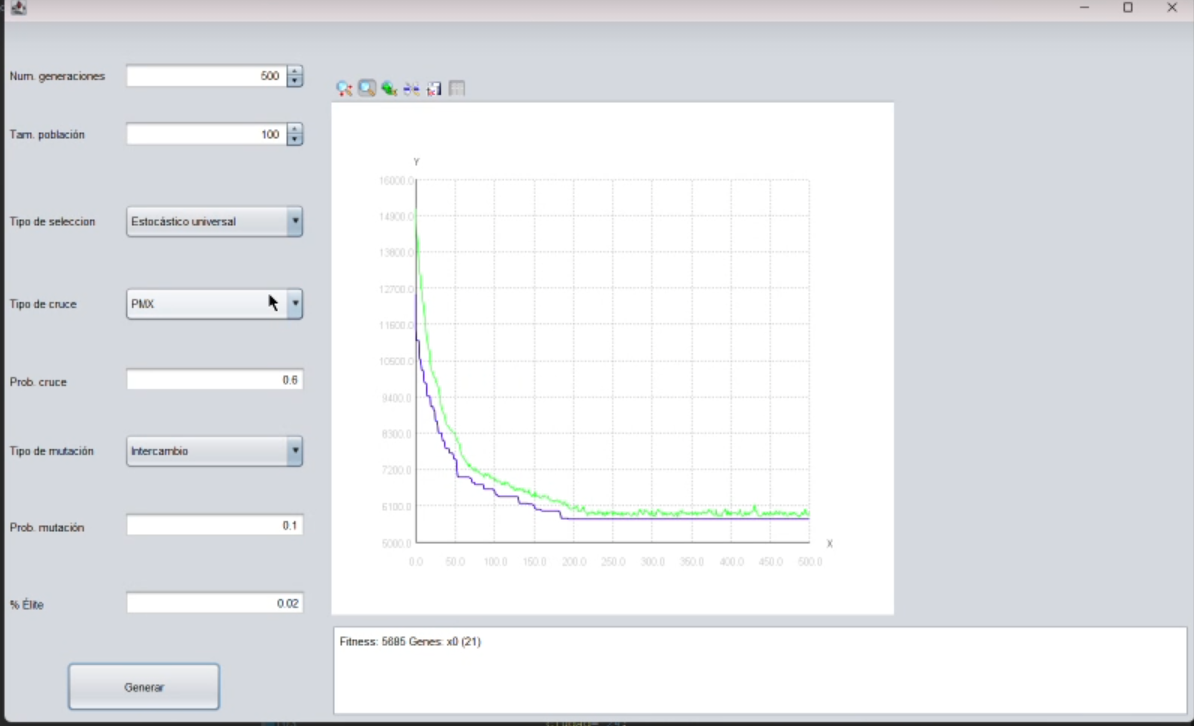
Práctica desarrollada por:

Eduardo Hernández Carriazo y [Juan Trillo Carreras](mailto:juantril@ucm.es)

# Gráficas

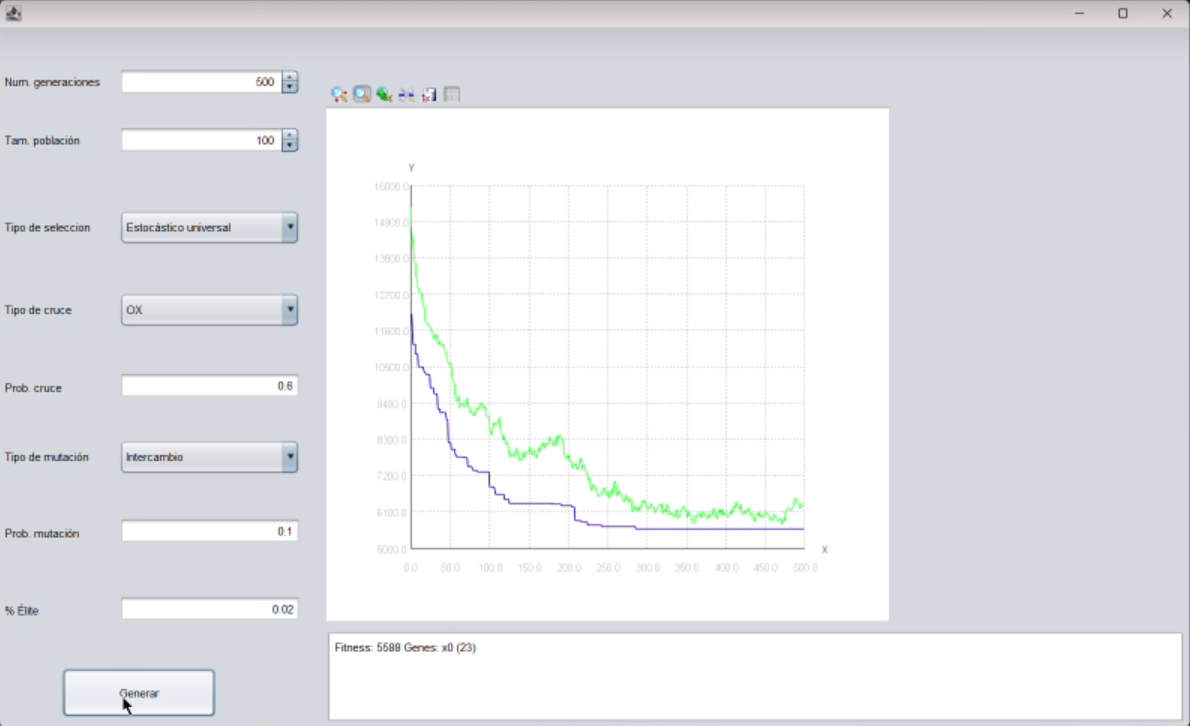
## TSP con PMX e Intercambio:



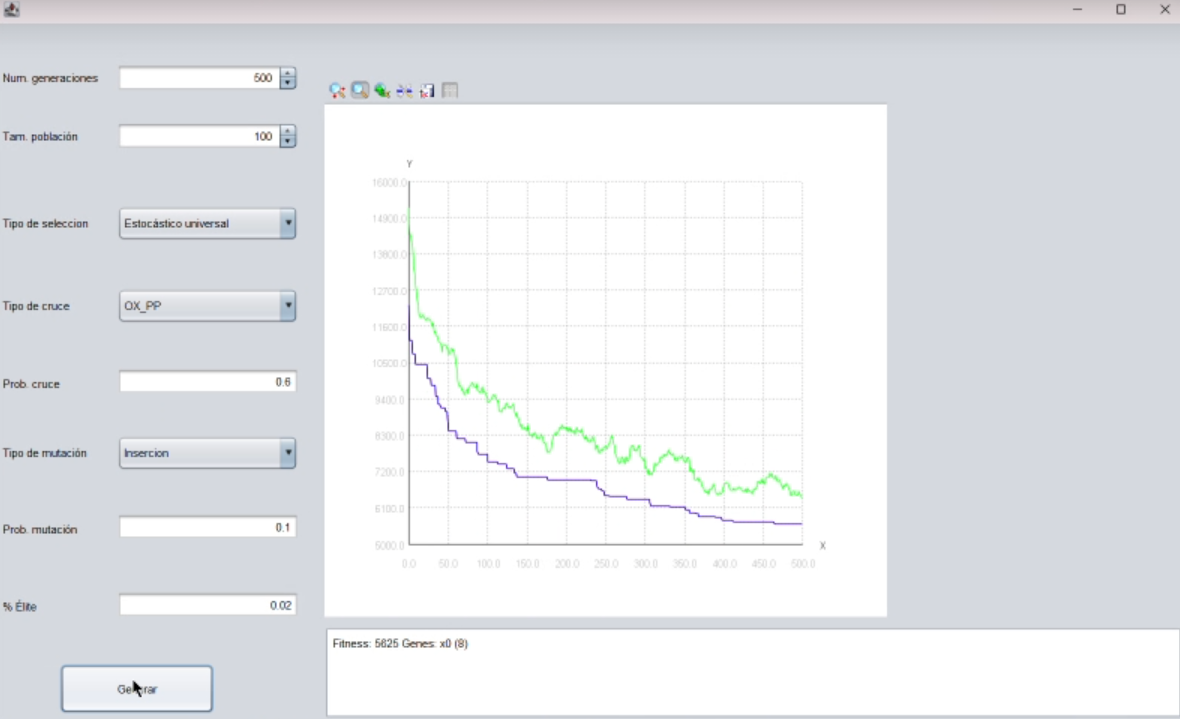


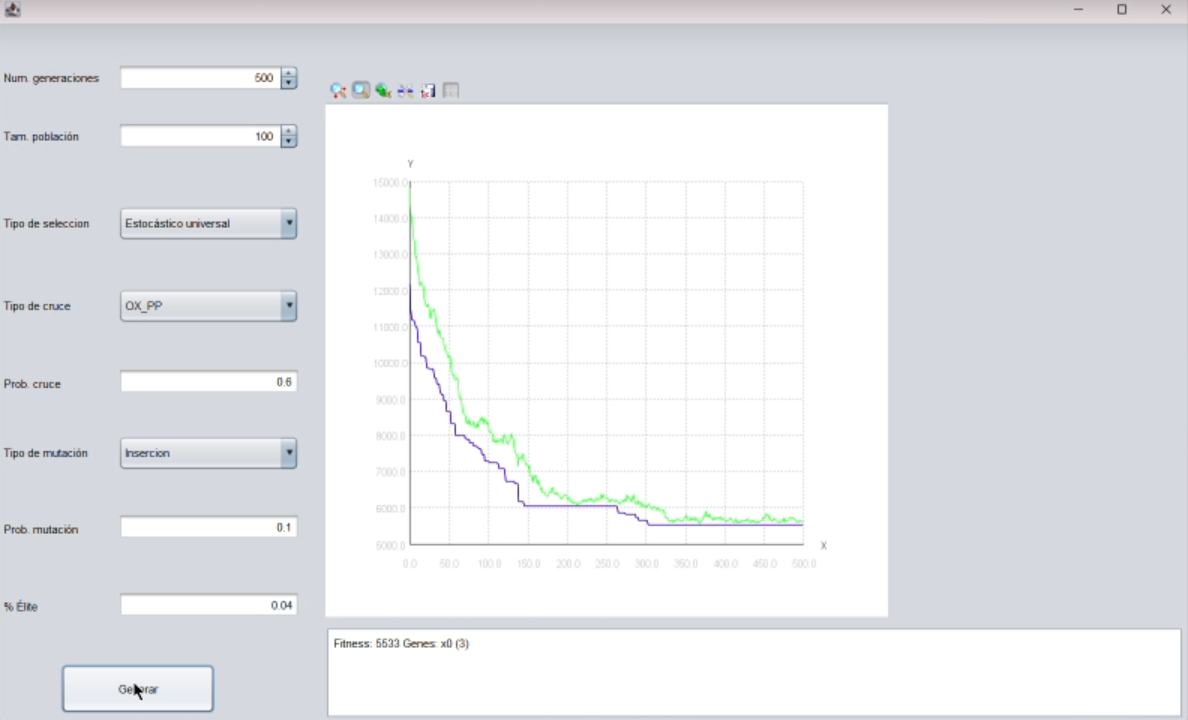
## TSP con OX y Intercambio:

## 



## TSP con OX\_PP y Intercambio:

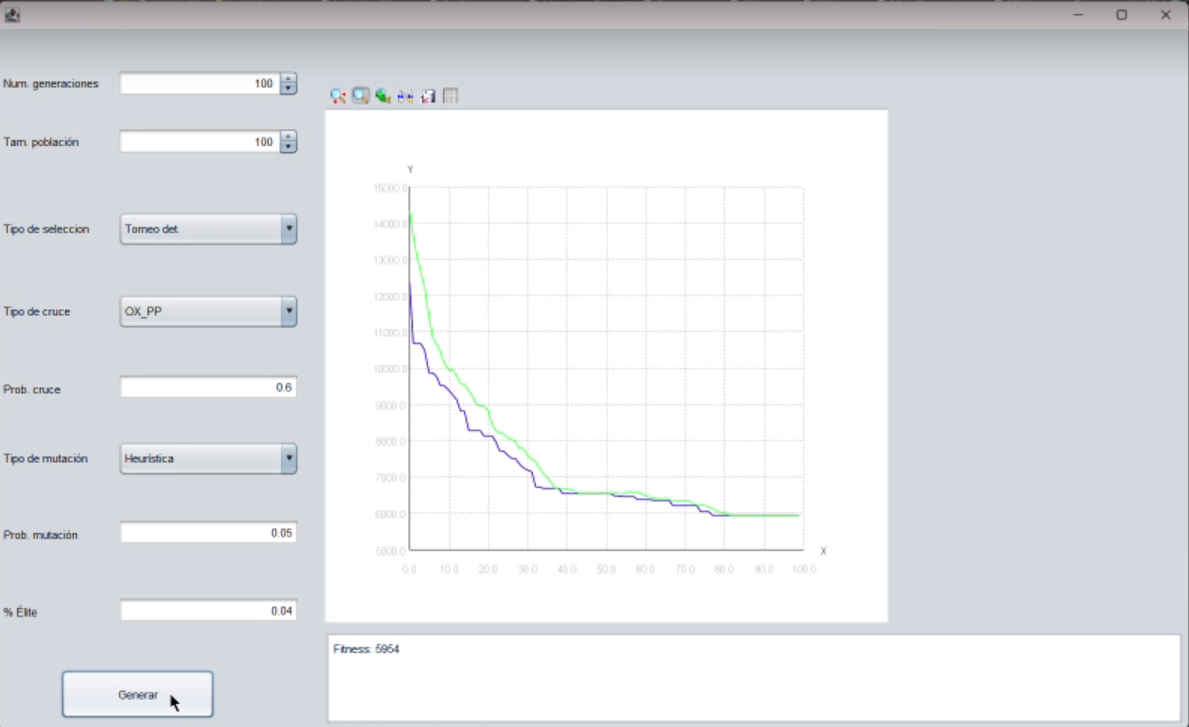




## TSP con PMX y Método propio de Mutación:

## 

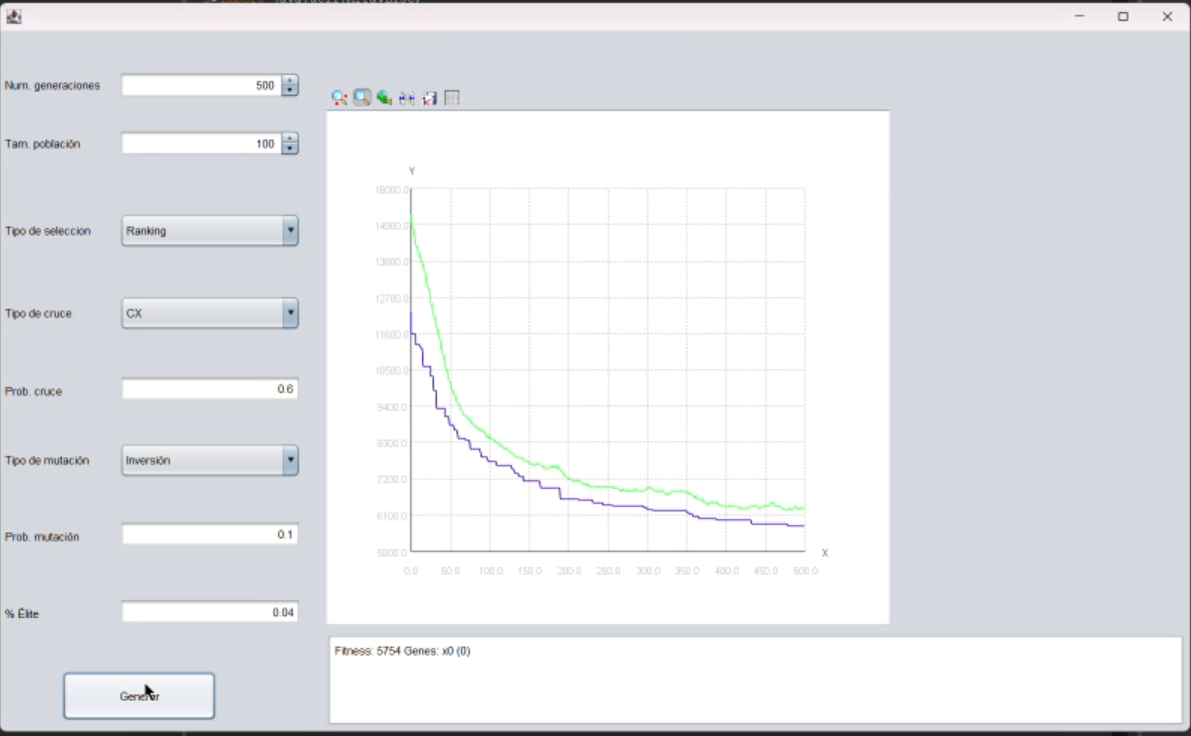
## TSP con OX\_PP y Mutación heuristica:

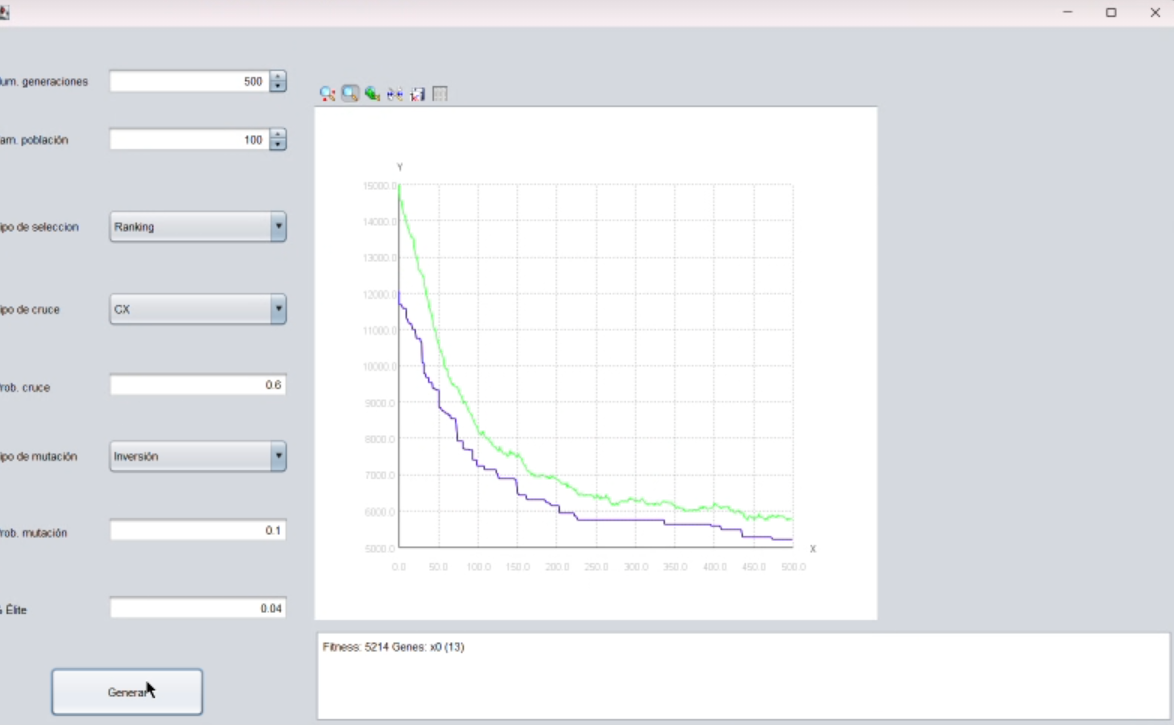


## TSP con OX\_OP y Insercion:

se nota que el elitismo afecta mucho, se recomienda un valor de 0.04 aumenta mucho los resultados

## TSP con CX y inversion:





# Conclusiones de los resultados

## ¿Cómo afecta el elitismo?

* 1. El elitismo hace que la diversidad de la población disminuya ya que siempre se van a conservar un % de individuos.
  2. El elitismo también hace que el algoritmo genético probablemente se quede estancado en un máximo/ mínimo local en vez de encontrar el global.
  3. En los casos en los que el elitismo tenga un porcentaje muy alto o se use con métodos muy elitistas (como truncamiento), hará probable una convergencia prematura.
  4. En esta práctica el elitismo es más que necesario si se quiere llegar a un buen resultado hemos encontrado que con 4% llegamos siempre a valores muy interesantes

## ¿Qué método de selección se comporta mejor?

* 1. En nuestra opinión,el peor método con diferencia es el ranking pero el resto se conforman de manera muy parecida si usamos un nivel apropiado de elitismo (4%).
  2. concluimos que el método de mutación heurística es uno de los peores mientras que el resto están muy bien

## Mejoras

* 1. Hemos incluido la corrección de escalado a la hora de realizar ruleta y selección estocástica universal.
  2. Hemos incluido 2 métodos propios
     1. Uno para cruce(Muy parecido a CO pero con cruce uniforme )
     2. Uno para mutación (que es básicamente un shuffle del cromosoma)
  3. También hemos incluido una bonita interfaz gráfica la cual permite generar varias gráficas cambiando cualquier dato en una misma ejecución.
  4. Hemos incluido una interfaz gráfica de NetBeans en un proyecto de Eclipse.
  5. Hemos incluido en la interfaz gráfica de forma más visible ,una representación para poder ver todos los genes de la función 4 independientemente de las dimensiones escogidas

Para verlos todos , el usuario debe introducir manualmente saltos de línea (pulsar enter) en el JTextArea de abajo para ir viendo los genes a su gusto.

## Análisis de la convergencia

* 1. A diferencia de nuestra primera práctica notamos que en la mayoría de casos independientemente del método de mutación llegamos a valores aceptables de entre 5100 y 5700 km
  2. En esta práctica notamos una gran evolución a partir de la 300 generación, para este problema consideramos óptimo usar 500 generaciones;100 de tamaño de población;0.6 prob de cruce y 0.1 de mutación;
     1. Todos los métodos a usar son decentes

## Problemas o curiosidades encontradas

* 1. Algo que resulta muy indeseable de este problema es la organización de la tabla de distancias (al estar madrid en la posición número 24 el problema se vuelve mucho más enrevesado a la hora de tocar los índices)
     1. Una posible solución a esto es simplemente cambiar el problema a salir desde la primera o última ciudad dada o cambiar a MANO la tabla de distancias
  2. Hemos encontrado esta práctica muy interesante a la par que desafiante, como la anterior aunque reconocemos que tener ya la primera práctica hecha de forma correcta nos ayudó mucho ya que la segunda está bastante basada en los conocimientos de la primera. Cabe destacar que para esta práctica hemos tenido apenas una semana en comparación a la primera (Esto obviamente por nuestra culpa, ya que tuvimos que hacer la repesca de la primera práctica \_Lo cual era IMPRESCINDIBLE para llevar a cabo la segunda con más facilidad) debido a ello no hemos podido implementar todas las opciones que nos hubiera gustado, ni terminar de forma correcta algunas implementaciones.

# Detalles de implementación

La función main de nuestro programa se encuentra en MainComposite de nuestro proyecto.

Para ver cómo se toman los valores y a continuación se ejecuta recomendamos encarecidamente leer la función “JButton1ActionPerformed” encontrada en main Composite (es la función utilizada al pulsar sobre el botón “Generar”).

Hemos incluido en la interfaz gráfica de forma más visible ,una representación para poder ver todos los genes de la función 4 independientemente de las dimensiones escogidas. Para verlos todos , el usuario debe introducir manualmente saltos de línea (pulsar enter) en el JTextArea de abajo para ir viendo los genes a su gusto.

***Recordamos que hay que incluir el Jar de la interfaz gráfica para el LinearPlot!***

# Reparto de tareas

Nos hemos repartido el trabajo de forma equitativa aunque Eduardo Hernández ha sido el que más tiempo ha estado trabajando sobre el código por diversos motivos en los que Juan Trillo no ha podido programar(fallos con GIT). Aunque siempre que se ha modificado el código los dos alumnos han estado presentes y trabajando codo con codo sobre el código;

Juan Trillo ha redactado esta memoria.

Tenemos la impresión de que se ha realizado en esta ocasión Eduardo ha participado más que Juan en la práctica para poner un porcentaje sería preciso decir: 60% Eduardo y 40% Juan.Debido principalmente a que durante la repesca [Juan Trillo Carreras](mailto:juantril@ucm.es) fue el que más se implicó y terminó de corregir todos los errores resultantes.

En general en esta asignatura sí que hemos dedicado el mismo tiempo ambos.