

Algoritmo genético

Koldo Aranegui Huarte| Computación | 12/05/2021

# Planteamiento y resolución del problema

El problema del viajante consiste en elegir la ruta con menos distancia, viajando a todas las ciudades una sola vez, por lo que es un problema de permutaciones. Después de generar aleatoriamente una matriz de distancias, creamos una población donde cada cromosoma es generado por permutación (para que no haya dos genes iguales en un mismo cromosoma), donde cada gen representa una ciudad.

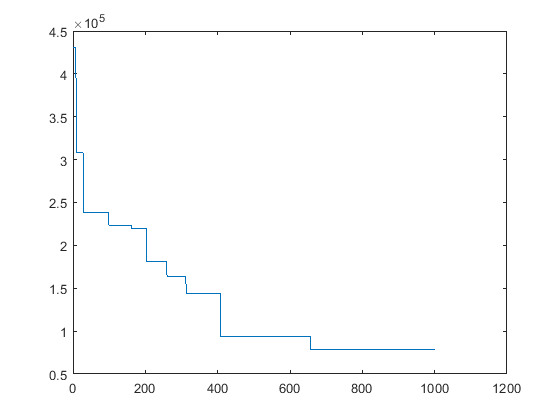
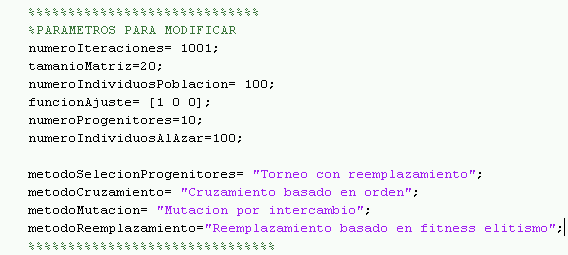
Para asignar el fitness a la población, tenemos una función de coste, y como queremos maximizar el fitness de la ruta con menor distancia, le paso a la función el valor 1/(sumatorio de distancias + 1 ). De esta forma garantizamos que el mayor fitness sea de la ruta con menor distancia.

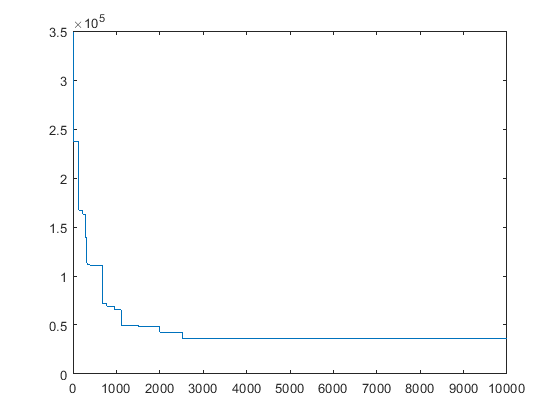
En cada iteración, seleccionamos los progenitores, realizamos el cruzamiento y la mutación (si existe) y posteriormente aplicamos un método de reemplazamiento a la población anterior. Este proceso se repite hasta llegar al número máximo de iteraciones, obteniendo el cromosoma con mejor fitness.

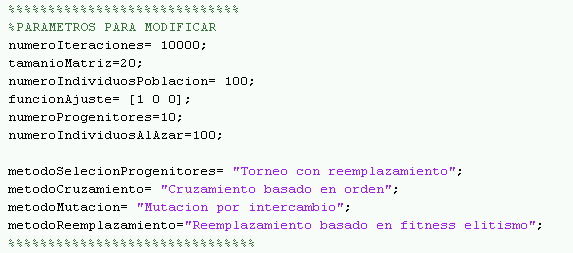
Hay distintos métodos de selección de progenitores, de mutaciones (o también la ausencia de mutaciones), de cruzamientos donde el usuario puede elegir, pasándolo por parámetro a la función AlgoritmoGenetico: para elegir los progenitores, se puede usar el método de la “Ruleta”, “Torneo sin reemplazamiento” y “Torneo con reemplazamiento”. En relación a los métodos de cruzamiento, elegimos entre “Cruzamiento basado en orden” y “Cruzamiento parcialmente mapeado” .Para elegir el método de mutación, seleccionamos entre “Mutacion por intercambio” y “Mutacion por inversión”. Por último, en la selección de supervivientes tenemos el "Reemplazamiento basado en fitness elitismo".

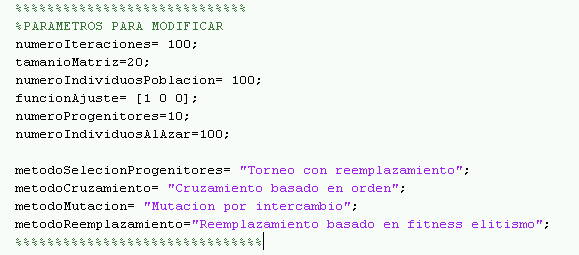
Al finalizar el número de iteraciones, obtenemos el cromosoma con mejor fitness de la última población obtenida.

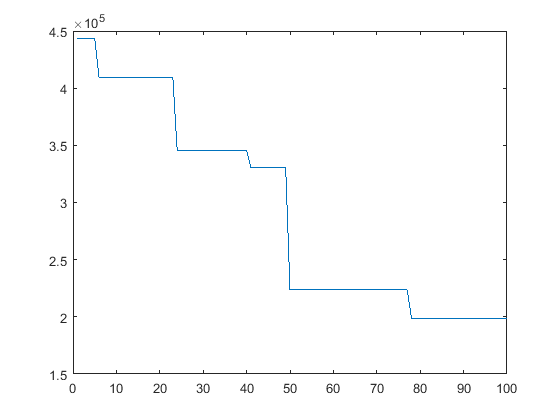
# Experimentos llevados a cabo

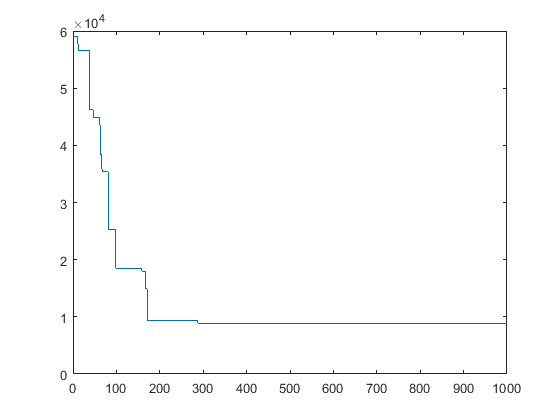
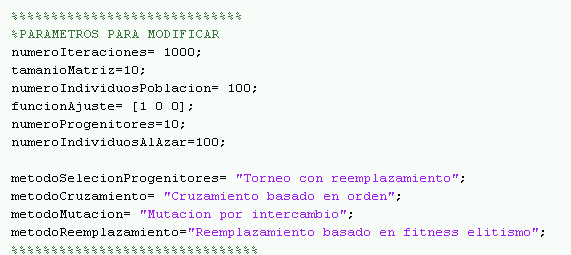


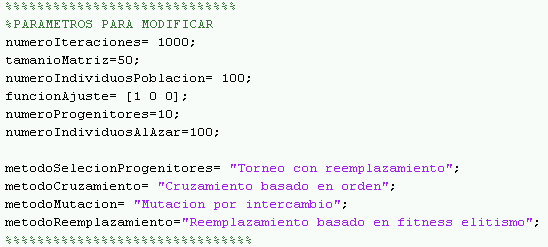


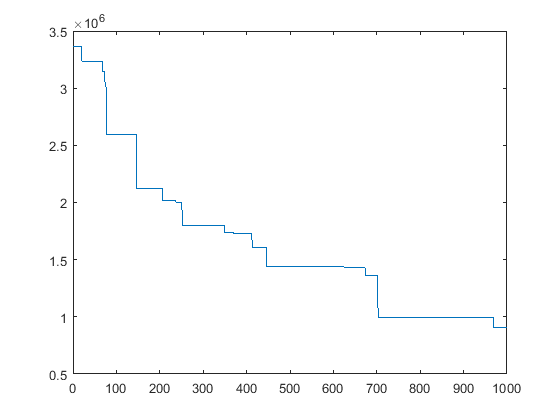


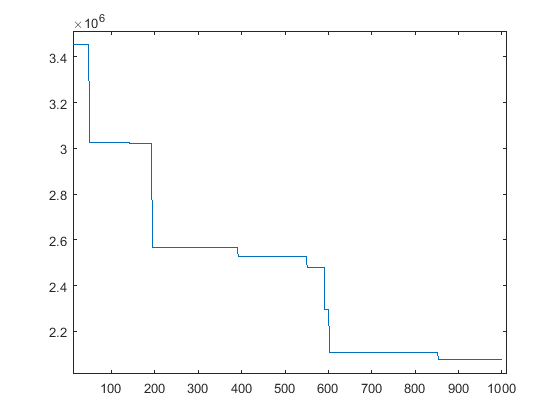
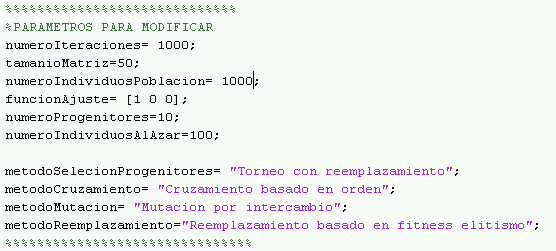


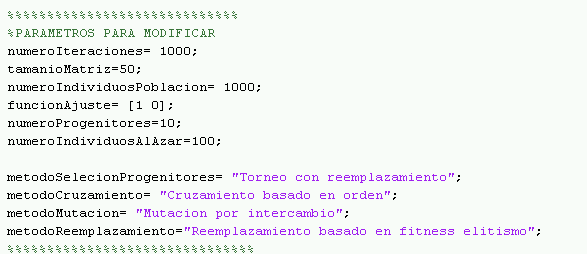


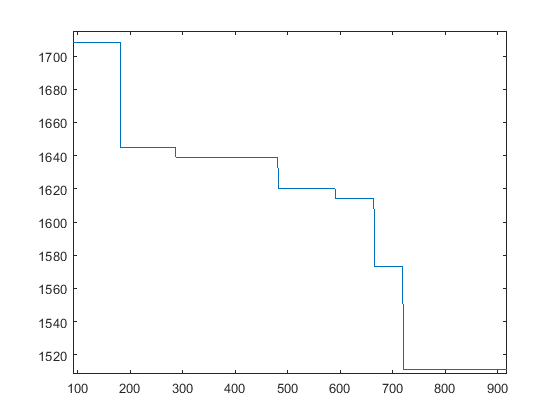


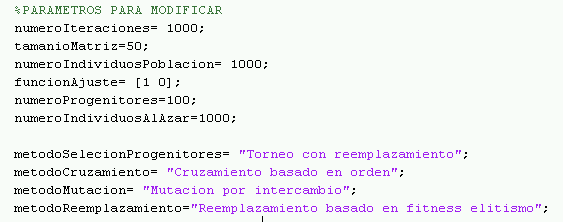


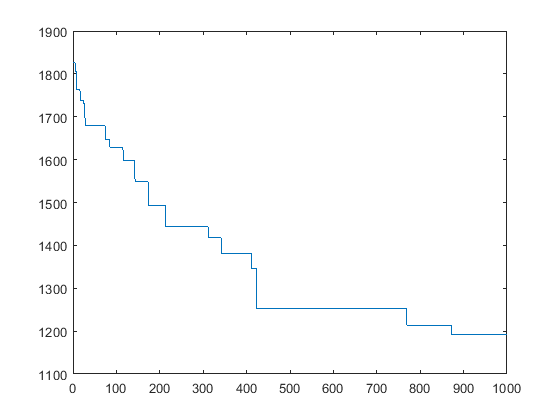


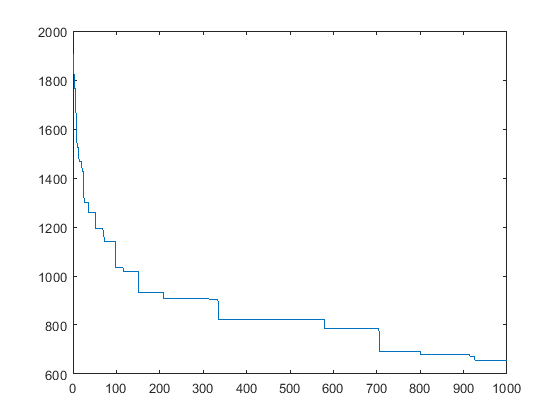
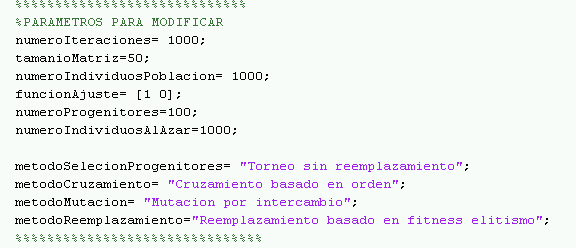


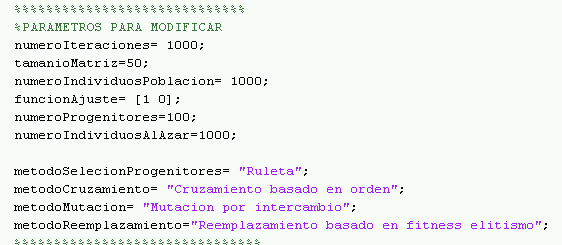


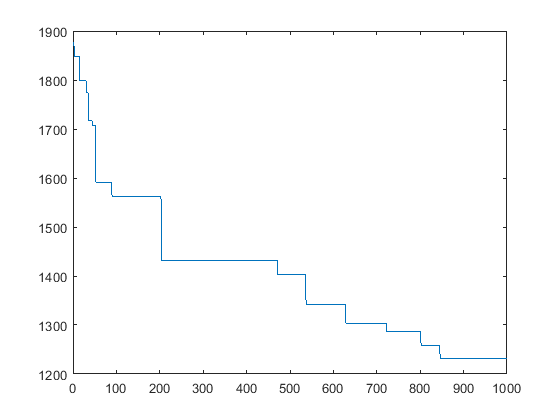


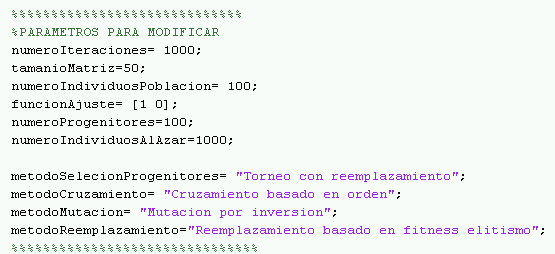


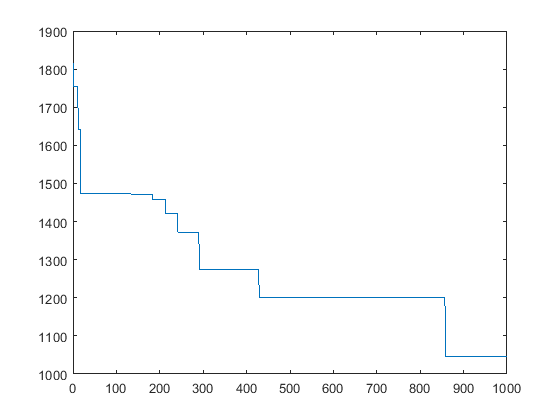


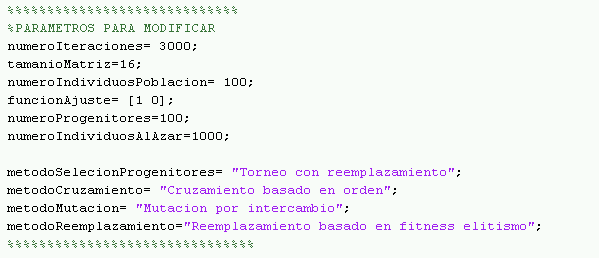


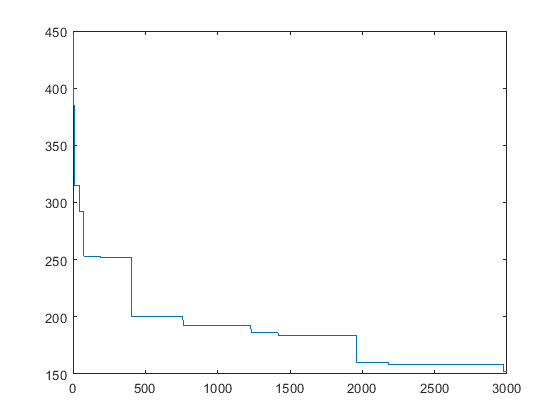


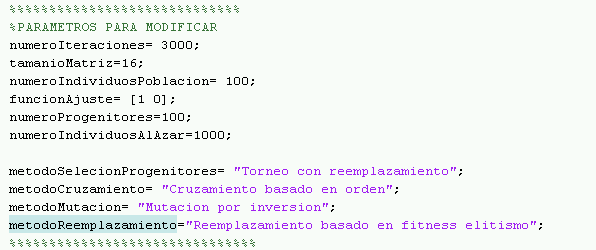


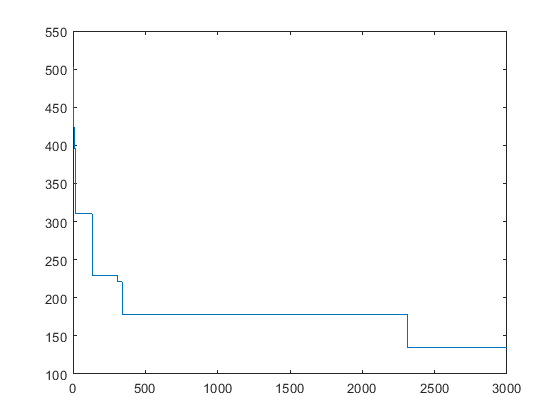


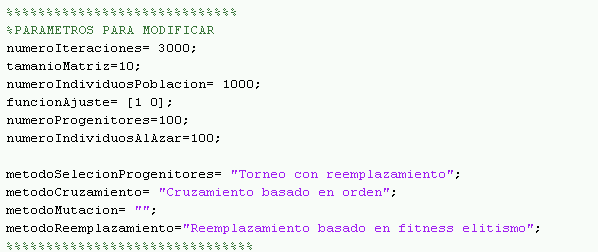


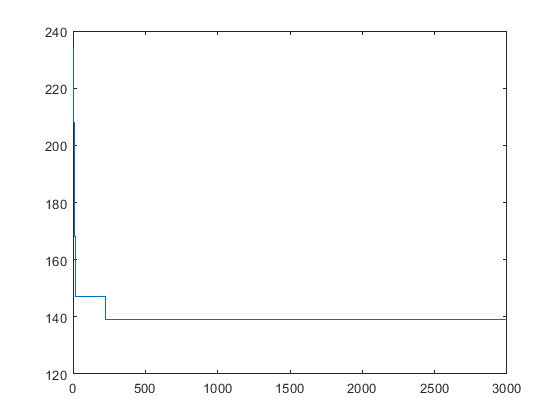


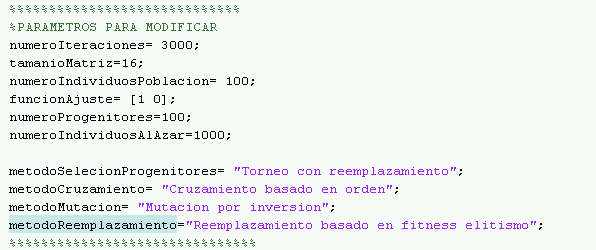


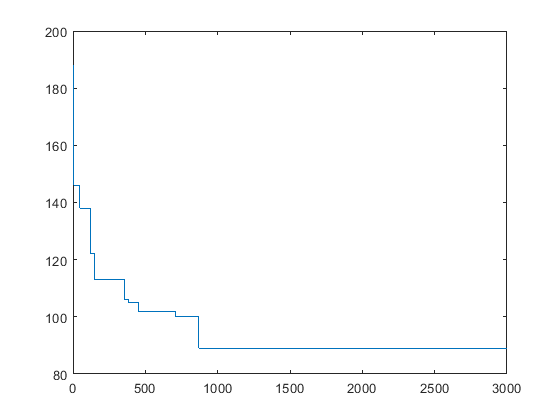


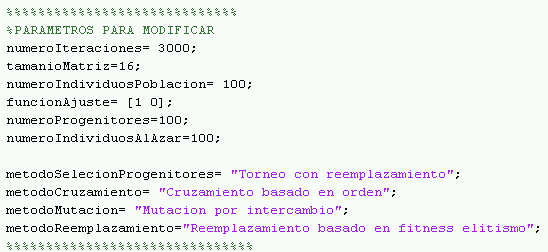


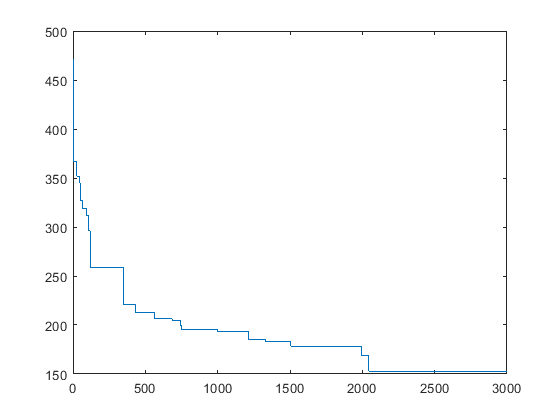












# Estudio del comportamiento del algoritmo

Tras realizar todos los experimentos, podemos observar que el algoritmo siempre converge a una solución.

El primer parámetro analizado ha sido el número de iteraciones. Al probar con los valores 100, 1000 y 10000, he concluido que 100 eran muy pocas iteraciones y 10000 demasiadas. 1000 iteraciones se acercaban más al número deseado, pero más tarde he podido comprobar que 3000 era un valor que se ajustaba mejor a nuestro algoritmo, ya que seguía encontrando mejores cromosomas.

Inicialmente he utilizado la función de Ajuste f(x)= x^2 (donde “x” es la distancia total), pero al cambiarla a una función f(x)=x, el algoritmo seguía convergiendo a una velocidad adecuada al número total de iteraciones que se habían asignado anteriormente.

En relación al número de ciudades (tamaño de la matriz), podemos ver que, cuantas más ciudades haya, necesitamos un mayor número de iteraciones. Para 10 ciudades, no necesitamos tener un número de iteraciones superior a 1000 (sería suficiente con 500) y con 50 ciudades nos ajustamos mejor al número de iteraciones, aunque sería mejor subirle el valor a 3000.

Tras probar los distintos métodos de selección de progenitores, he podido comprobar que el Torneo sin reemplazamiento es el método más adecuado para este algoritmo, ya que logra obtener un mejor fitness que el Torneo con reemplazamiento y el método de la Ruleta. La diferencia con los otros dos métodos es significativa. Entre el método de la Ruleta y el Torneo con reemplazamiento no hay apenas diferencia, por lo que sería más útil usar el método del Torneo ya que es más sencillo de implementar.

Al generar una población de 100 cromosomas y después una de 1000, vemos que obtenemos un cromosoma con mejor fitness con la población de 1000, ya que al generar más cromosomas, tenemos más probabilidad de seleccionar cromosomas con mejor fitness, por lo que la distancia del cromosoma solución será menor en este caso.

Al probar los cruzamientos, he seleccionado para las demás pruebas el Cruzamiento basado en orden, ya que el Cruzamiento Parcialmente Mapeado me da error al seleccionar un número alto de iteraciones.

Por último, tras las distintas pruebas con el método de mutación la presencia y la ausencia de mutación dan un resultado similar en la iteración 3000, sin embargo la presencia de mutaciones hace que el algoritmo tarde más en converger.