**Nombre:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Fecha: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Instrucciones.** *Lee cuidadosamente cada pregunta y contesta lo que se te pide. No olvides escribir subir todos los archivos .m elaborados para cada inciso que lo requiera de uso de software y pegar el código de programa indentado y comentado.*

1. (**10 puntos**) Considere las siguientes variables:

A=Edad

B=Número del mes de nacimiento

C=Expediente

D=Los dos últimos dígitos de tu estatura en centímetros.

Si se tienen tantos números como ACD/B cuántos bits son necesarios para mapear todos los números a números enteros positivos.

Si suponemos que la edad puede ser de 0-99 años, número de mes de año de 1-12, expediente de 0-999999, dos últimos dígitos de estatura de 00-99, entonces; el mejor caso (menor cant. De bits) será 0\*0\*0/1 = 0 y con 1 bit es suficiente. Sin embargo, el peor caso (mayor cant. de bits) será 99\*999999\*99/1 = 9.8e9 y serán necesarios: ceil(log2((9.8e9) +1)) = 34 bits.

1. (**10 puntos**) Escribe cuatro factores que pueden ocasionar que un algoritmo genético falle y cuatro razones por las cuales podría fallar el PSO en la búsqueda del óptimo.

|  |
| --- |
| **Algoritmos Genéticos** |
| * El óptimo se encuentra fuera del rango establecido. |
| * El óptimo no se encuentra en alguno de los tamaños de paso establecidos (si vamos de 5 en 5, jamás se tocará el ‘24’, suponiendo que el 24 sea el deseado) |
| * Si se muta muy frecuentemente el algoritmo genético podría no converger a un óptimo antes de finalizar las iteraciones deseadas. |
| * Si no se muta o se muta muy poco se podría estancar el algoritmo genético. * Si se encuentra un Padre muy bueno podría estancarse en este. |

|  |
| --- |
| **PSO** |
| * Si la función es discontinua podría tener problemas para converger |
| * Si el tamaño de paso es muy grande podría saltar el punto al cual se desea converger |
| * Si el tamaño de paso es muy pequeño no se lograría converger antes de finalizar las iteraciones. |
| * Si la posición inicial de las partículas está muy alejada del óptimo podría no llegar a converger. |

1. **(20 puntos) Mutación modificada.** Desarrollar un código para modificar la mutación en un algoritmo genético.

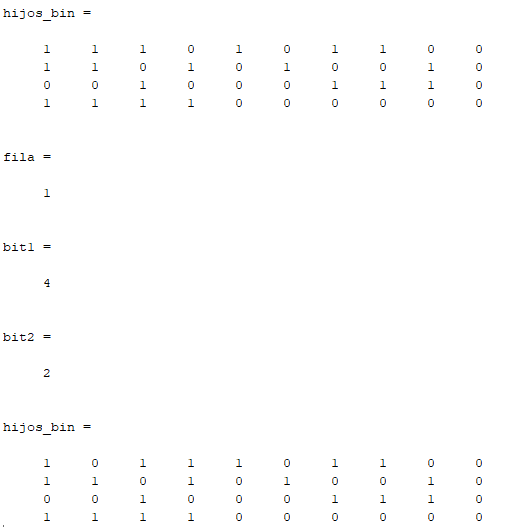
El algoritmo debe de realizar lo siguiente:

**Mutación:** Usar una probabilidad de mutación del 15%

Seleccionar un hijo al azar.

Seleccionar aleatoriamente dos bits diferentes del hijo seleccionado.

Intercambiar de posición los bits del hijo que mutó.



nhijos = 4

nbits = 10

hijos\_bin = randi([0,1],nhijos,nbits)

if rand()<=.9

fila = randi([1,nhijos])

bit1 = randi([1,nbits])

bit2 = randi([1,nbits-1])

if bit1==bit2

bit2 = bit2 + 1

end

tmp = hijos\_bin(fila,bit1);

hijos\_bin(fila,bit1) = hijos\_bin(fila,bit2);

hijos\_bin(fila,bit2) = tmp

end

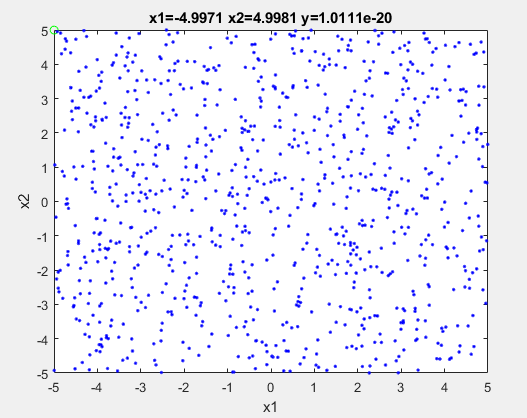
1. **(20 puntos)** Considere la siguiente función multivariable definida a trozos:

Determine el mínimo dentro del intervalo -5≤ x1 ≤5 y -5≤ x2 ≤5

Corriendo la primer simulación a trozos:

rest = 'a\*max(x(1)-5,0) - a\*min(x(1)+5,0) + a\*max(x(2)-5,0) - a\*min(x(2)+5,0) - a\*min(x(1) + x(2) -.00001,0) ';

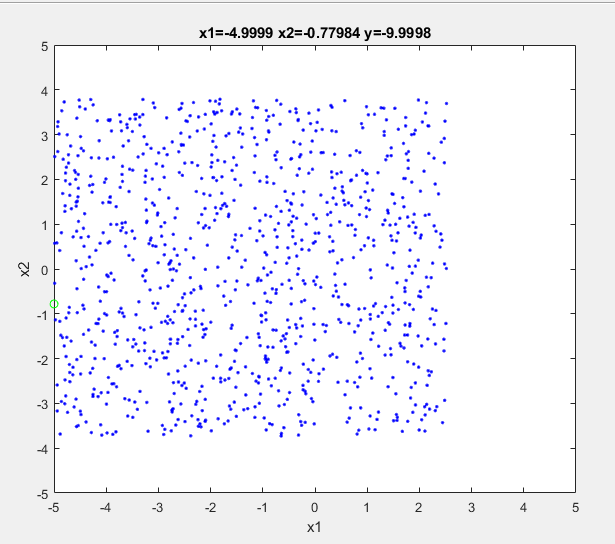
func = '((x(1).^2+x(2).^2)./(exp(x(1).^2+x(2).^2)+1)) + eval(rest)';



Evaluando la segunda función:

rest = 'a\*max(x(1)-5,0) - a\*min(x(1)+5,0) + a\*max(x(2)-5,0) - a\*min(x(2)+5,0) + a\*max(x(1) + x(2),0) ';

func = '(2\*x(1)) + eval(rest)';



Se puede notar que la segunda función evaluada en (-5,-0.7798) tiene la mejor evaluación, que es -9.9998.

1. **(20 puntos)** Una empresa produce y comercializa tres tipos de productos, P1, P2 y P3, que sirve en palés, que pueden o no estar completos (se puede entregar medio palé, un cuarto de palé o cualquier otra porción de palé). Por cada palé de estos productos, se obtienen unos ingresos netos de 4, 12 y 2 unidades monetarias, respectivamente. Existe una instalación de la que se dispone de un total de 600 días de trabajo. Producir un palé de P1 lleva 3 días, uno de P2 lleva 6 días y montar uno de P3 lleva 2 días. Además, existe un compromiso de entregar al menos el contenido conjunto equivalente a doscientos cuarenta palés. También se sabe que no se pueden vender más de 140 palés del producto P3 y no se vende menos de un palé del producto P2. Suponga que se tiene el espacio suficiente y los productos no se descomponen.

Formule un modelo adecuado para el problema programación lineal para este problema e indique la cantidad de palés necesario de cada producto que permitan maximizar los ingresos netos.

Con la siguiente combinación se logró el mejor rendimiento:

x1 =100.6000 x2=4.8200 x3=134.6400

x1=97.9600 x2 = 5.2600 x3=137.2800

El rendimiento obtenido es de 729.52 con cualquiera de las dos combinaciones propuestas en la parte superior.

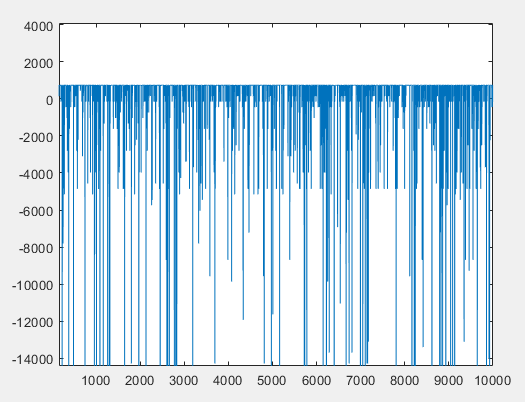
r1 = 'max(3\*x(1) + 6\*x(2) + 2\*x(3) - 600, 0)';

r2 = '- min(x(1) + x(2) + x(3) - 240, 0)';

r3 = ' - min(x(1),0) - min(x(2)-1,0) - min(x(3),0) + max(x(3)-140,0)';

rest = 'eval(r1) + eval(r2) + eval(r3)';

func = '(4\*x(1) + 12\*x(2) + 2\*x(3)) - a\*eval(rest)';



La imagen muestra a evolución del algoritmo genético en 10,000 iteraciones con 32 padres. Se puede apreciar que en todo momento (desde el inicio, prácticamente) llegan a una solución bastante cercana al máximo.

1. **(20 puntos)** Use el algoritmo adecuado para resolver el siguiente problema:

**Maximiza Z=**20x1+10x2+25x3+20x4

**S.A.**

x1 + x2 + x3 + 2x4 ≤ 120

3x1 + x2 + 2x3 + 2x4 ≤ 273

x1 + 2x2 + 5x3 + 3x4 ≤ 355

y

xj≥0 para j=1,2,3,4

xj es entero para j=1,2,3

la mejor combinación de x es:

x1 = 47.0000 x2 = 0 x3 = 57.0000 x4 = 7.6660

F(xi) = 28,653.32

r1 = 'max(x(1) + x(2) + x(3) + 2\*x(4) - 120, 0)';

r2 = 'max(3\*x(1) + x(2) + 2\*x(3) + 2\*x(4) - 273, 0)';

r3 = 'max(x(1) + 2\*x(2) + 5\*x(3) + 3\*x(4) - 355, 0)';

r4 = ' - min(x(1),0) - min(x(2),0) - min(x(3),0) - min(x(4),0)';

rest = 'eval(r1) + eval(r2) + eval(r3) + eval(r4)';

func = '(20\*x(1) + 10\*x(2) + 25\*x(3) + 20\*x(4)) - a\*eval(rest)';