<u>Programa que trabaja con algoritmos genéticos codificados con bits</u> [9]

A continuación se encuentran los programas adicionales necesarios para correr un algoritmo genético que compute básicamente lo mismo que el utilizado en el trabajo final pero con bits.
Para que el programa sea operativo deben encontrarse los siguientes files en la carpeta work del matlab.

```
market__GA_bits
bit_integer
population_bits
sorted_GA
chromo_cross
chromo_mutate_bits
benchmark


market__GA_bits%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear

global N bits rules b a k gen l
format long g

N=4
bits=11
rules=8
gen=2500;
l=1;
graf=2500;
rew=500

rand('state',0)

%parámetros de la demanda desconocida para los productores
b=0.4
a=15
k=0.36

%costos marginales constantes
c=4

clear

firm = population_bits(rules,bits, N);

for i=1:N
firm(i).r = 1;
end
```

[9] Programa correspondiente a la nota al pie 11 del trabajo final para el curso de Racionalidad Acotada. Profesores, Daniel Heymann, Roberto Perazzo y Martín Zimmermann. Maestría en Economía de la Universidad de San Andrés.
Alumna: Karina Otero. Noviembre, 2006.

```
while l<=gen;
  l

%cantidades ofrecidas al mercado

output= bit_integer( [firm(1).genome(firm(1).r ,:)]);
for i=2:N
output= bit_integer( [firm(i).genome(firm(i).r ,:)]) + output;
end

%precio de mercado
P= a * ones(size(output))- (output.^k) .*b;

if (l==1 | mem==0)
Pext=P*ones(rules,1);

%ingreso que los productores
for i=1:N
firm(i).income= bit_integer( [firm(i).genome]) .* Pext;
end

%costo de los productores
for i=1:N
firm(i).cost= bit_integer( [firm(i).genome]) .* c;
end

%fitness que perciben los productores de cada una de sus reglas
for i=1:N
firm(i).fitness=firm(i).income - firm(i).cost;
end

else

%ingreso de los productores
for i=1:N
firm(i).income(firm(i).r,:)=bit_integer(firm(i).genome(firm(i).r,:)) .* P;
end

%costo de los productores
for i=1:N
firm(i).cost(firm(i).r,:)=bit_integer(firm(i).genome(firm(i).r,:)) .* c;
end

%fitness que los productores
for i=1:N
firm(i).surplus(firm(i).r,:)=firm(i).income(firm(i).r,:) - ...
    firm(i).cost(firm(i).r,:);
end

for i=1:N
firm(i).fitness(firm(i).r,:)=sum(firm(i).surplus(firm(i).r,:), 2);
end
```

```
end

%Selección y evolution

for i=1:N
[sort_genome, sort_fitness]= sorted_GA(firm(i).genome, firm(i).fitness);
firm(i).sort_genome = sort_genome;
firm(i).sort_fitness = sort_fitness;
end

for i=1:N
firm(i).new_genome=firm(i).sort_genome;
j = randint(1,2,[1 (numel(firm(i).genome(:,1))-2)]);
if (j(1) ~= j(2))
[new1 new2] = chromo_cross(firm(i).sort_genome(j(1),:), firm(i).sort_genome(j(2),:));
firm(i).new_genome(numel(firm(i).genome(:,1)), :)=new1;
firm(i).new_genome(numel(firm(i).genome(:,1))-1, :)=new2;
end
end

for i=1:N
j = randint(1,1,[3 numel(firm(i).genome(:,1))]);
firm(i).new_genome(j,:) = chromo_mutate_bits(firm(i).new_genome(j,:));
end

%%%%
%Data agregada

data_output(l,1)=sum(output);

data_P(l,1)=sum(P);

for i=1:N
firm(i).benef=firm(i).fitness(firm(i).r,1);
end
mean_benef(l,1)= sum([firm.benef], 2)/N;

if (mod(l,graf)==0)
%últimos datos
disp('***************Ultimas cantidades ofrecidas por período **********')
output_data = data_output((l-8):l,1)
output_data_prom_ultimas_8 = sum(data_output((l-8):l,:))/...
    length(data_output((l-8):l,1))
disp('***************Ultimos precios observados por período ************')
P_data = data_P((l-8):l,1)
P_data_prom_ultimas_8=sum(data_P((l-8):l,:))/length(data_P((l-8):l,1))
disp('***************Ultimos beneficios promedio por período ***********')
benef_mean=mean_benef((l-8):l,1)
benef_mean_prom_ultimas_8 = sum(mean_benef((l-8):l,:))/...
    length(mean_benef((l-8):l,1))
%Gráfico
figure1 = figure('Color',[1 1 1]);
hold
subplot(3,1,1),plot(data_output, '-', 'Color',[0.8549 0.702 1])
ylabel('Cantidades', 'FontSize',8)
```

```
subplot(3,1,2), plot(data_P,'-g.', 'Color',[0 0.749 0.749], 'MarkerSize',5)
ylabel('Precios', 'FontSize',8)
subplot(3,1,3),plot(mean_benef,'-', 'Color',[0.8549 0.702 1])
xlabel('Períodos', 'FontSize',8)
ylabel('Beneficios', 'FontSize',8)
drawnow
pause(5)
end

%%%%evolución en torno al equilibrio

if (mod(l,graf)==0)
[P_Cour, Q_Cour, Benef_Cour, Pperf, Qperf] = ...
    benchmark( a, b, k, c, N, data_output, data_P)
drawnow
pause(5)
end

%%%%Data por productor

for i=1:N
firm(i).payoff(l,1)=firm(i).fitness(firm(i).r,:);
end

for i=1:N
firm(i).quant(l,1)=bit_integer(firm(i).genome(firm(i).r,:));
end

for i=1:N
firm(i).acum_std_quant=(firm(i).quant-(sum([firm.quant], 2)/N)).^2;
end

std_quant=sum([firm.acum_std_quant], 2)/N;

for i=1:N
firm(i).acum_average_payoff(l,1)=sum(firm(i).payoff)/length(firm(i).payoff);
end

if (mod(l,graf)==0)
disp('************** Evolución por firma, cantidades y beneficios********')
figure1 = figure('Color',[1 1 1])
subplot(2,1,1), plot([firm.quant], 'Color',[0.8549 0.702 1])
ylabel('Cantidades', 'FontSize',8)
subplot(2,1,2), plot([firm.payoff], 'Color',[0 0.749 0.749])
ylabel('Beneficios','FontSize',8)
xlabel('Períodos','FontSize',8)
drawnow
pause(5)
end

if (mod(l,graf)==0)
disp('**************Desviación estandar de las cantidades medias por período')
plot(std_quant, 'Color',[0.8549 0.702 1])
xlabel('Períodos', 'FontSize',8)
ylabel('Desviación estandar de las cant.', 'FontSize',8)
```

```
drawnow
pause(5)
end

%Evolución para 4 firmas

if (mod(l,graf)==0)
if N>3
  disp('************** Evolución para las primeras 4 firmas')
  figure1 = figure('Color',[1 1 1])
  for i=1:4
  subplot(4,1,i), plot(firm(i).payoff, 'Color',[0.8549 0.702 1])
  xlabel('Períodos', 'FontSize',8)
  ylabel('Beneficios', 'FontSize',8)
  end
else

end
drawnow
pause(5)
end

if (mod(l,graf)==0)
if N>3
  disp('**************Muestra: evolución para 4 firmas')
  figure1 = figure('Color',[1 1 1])
  for i=1:4
  subplot(4,1,i), plot(firm(i).quant, 'Color',[0.8549 0.702 1])
  xlabel('Períodos','FontSize',8)
  ylabel('Cantidades','FontSize',8)
  end
else

end
drawnow
pause(5)
end

if (mod(l,graf)==0)
disp('************** Evolución del promedio de los beneficios acumulados')
figure1 = figure('Color',[1 1 1])
axes1 = axes(...
  'FontSize',8,...
  'Position',[0.1098 0.1685 0.7952 0.7565],...
  'Parent',figure1);
axis([0 500 500 5000]);
box('on');
hold('all');
subplot(1,1,1), plot([firm.acum_average_payoff]);
ylabel('Promedio  Benef.  Acum.');
xlabel('Períodos')
drawnow
pause(5)
end
```

```
%%%%%%%%%%historial

l=l+1;

for i=1:N
firm(i).genome=firm(i).new_genome;
end

for i=1:N
firm(i).fitness=firm(i).sort_fitness;
end

% Reglas
for i=1:N
X=zeros(1,2);

while X(1)== X(2)
X = randint(1,2,[1 (numel(firm(i).genome(:,1)))]);
end

if firm(i).fitness(X(1))>=firm(i).fitness(X(2));
firm(i).r = X(1);
else
firm(i).r = X(2);
end

end

if mem==0
for i=1:N
firm(i).r = 1;
end
end

end

l=l-1;

bit_integer%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function bit_integer = bit_integer( bits )

[x, pot] = size( bits );

for i = 1:pot
   num(i) = 2^(pot-i);
end

bit_integer = bits*num';

population_bits%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function firm = population(rules, bits, N)
```

```
for i=1:N;
firm(1,1,i).genome= rand(rules, bits)>= 0.5;
end

chromo_mutate_bits%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Operador de Mutación
function chromo = chromo_mutate_bits(chromo1)

len = length(chromo1);
x = randint(1,1,[1 len]);
chromo = chromo1;
chromo(x) = 1 - chromo(x);
```