Ajuste de datos de lectura de un sensor

Table of Contents

ENcuentra los parametros m y b de la ecuación de linea recta

$$y = mx + b$$

que mejor se ajuste a los datos de los sensores

se utiliza AG.

Contenido

```
% * Definición del problema
% * Parametros iniciales
% * Población inicial
% * Iteraciones
% * Oden aleatorio de parejas
% * Cruce
% * Mutación
% * calculo de la variable del AG
% * Selección
% * Resultados
% * Conclusiones
```

Definición del Problema

Se genera un conjunto de datos sinteticos

$(x_i, y_i)$$ para i=1,2,3,..,n

donde la variable dependiente "y" es un vector de numeros aleatorios en el intevalo de 0 a 1 por lo que se espera que la mejor solucion sea cercana a m=0, b=0.5

```
x = linspace(1,100); % dominio de 1 a 100
y = rand(1,100);     % numeros aleatorios en el intervalo de 0 a 1
```

Parametros iniciales

Se configura el AG definiendo parametros analogos

```
generaciones = 800; %interaciones del AG
n = 200; %tamaño de la piblacion
probabilidad = 0.6; % probabilidad de que cada GEN mute
mutacion = [1 1]; % maximo valor que puede mutar
```

Poblacion inicial

Se genera aleatoriamente una poblacion inicial de posibles soluciones. primer columna representa el GEN que define la pendiente "n" de la recta. La segunda columna representa el GEN que define la ordenada al origen "b" . Cada uno de los "n" individuos (renglones) representa una posible solucion.

```
poblacion = rand(n,2);
```

Iteraciones

La poblacion de soluciones evoluciona mediante el cruce y la mutacion

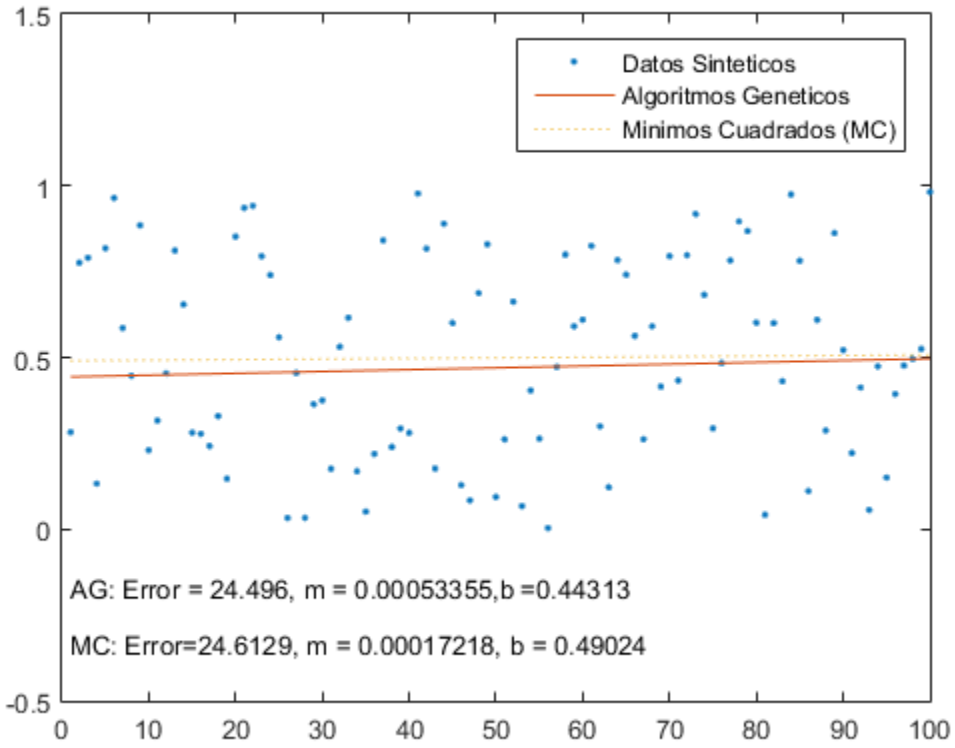
```
s = n;
hw = waitbar(0, 'Me gusta el pollo frito');
for i=1:generaciones
    %orden aleatorio de parejas
    poblacion(:,3)=rand(s,1);
    poblacion = sortrows(poblacion,+3);
    % cruce
    poblacion = [poblacion; poblacion(1:2:end-1,1)
 poblacion(2:2:end,2) zeros(fix(s/2),1); poblacion(2:2:end,1)
 poblacion(1:2:end-1,2) zeros(fix(s/2),1)];
    % Mutacion
    s = size(poblacion,1);
    poblacion = [poblacion; poblacion
+[((rand(s,3)<probabilidad)).*(rand(s,3)*2-1).*repmat([mutacion
 0],s,1)]];
    % calculo de la viabilidad del AG
    % la recera columna representara la viabilidad
    % solucion, menor valor indica una viabilidad mayor.
    s = size(poblacion, 1);
    m = repmat(poblacion(:,1),1,100);
    b = repmat(poblacion(:,2),1,100);
    X = repmat(x,s,1);
    Y = repmat(y,s,1);
    Y2 = m.*X+b;
    poblacion(:,3) = sum(abs(Y-Y2),2);

    %seleccion
    poblacion = unique(poblacion,'rows');
    poblacion = sortrows(poblacion,+3);
    s = min([size(poblacion,1),n]);
    poblacion = poblacion(1:s,:);
    waitbar(i/generaciones, hw)
end
close(hw)
```

Resultados

Se compara el resultado del AG contra el obtenido con minimos cuadrados.

```
error_ag = poblacion(1,3);
p = polyfit(x,y,1);
y3 = polyval (p,x);
error_mc = sum(abs(y3-y));
plot(x,y,'.', x,polyval(poblacion(1,1:2),x),x,y3,':')
axis([0 100 -0.5 1.5])
legend('Datos Sinteticos', 'Algoritmos Geneticos', 'Minimos Cuadrados
 (MC)')
text(1,-.167,['AG: Error = ' num2str(error_ag) ', m = '
 num2str(poblacion(1,1)) ',b =' num2str(poblacion(1,2))])
text(1,-.333,['MC: Error='  num2str(error_mc) ', m = '
 num2str(p(1)) ', b = ' num2str(p(2))])
```



Published with MATLAB® R2015a