Optimización

Modelado, Optimización y Simulación

Profesor
Germán Montoya

Oficina ML648

Optimización MultiObjetivo

- Modelo General:

Optimize [minimize/maximize]

$$F(X) = \{f_1(X), f_2(X), \ldots, f_n(X)\}$$

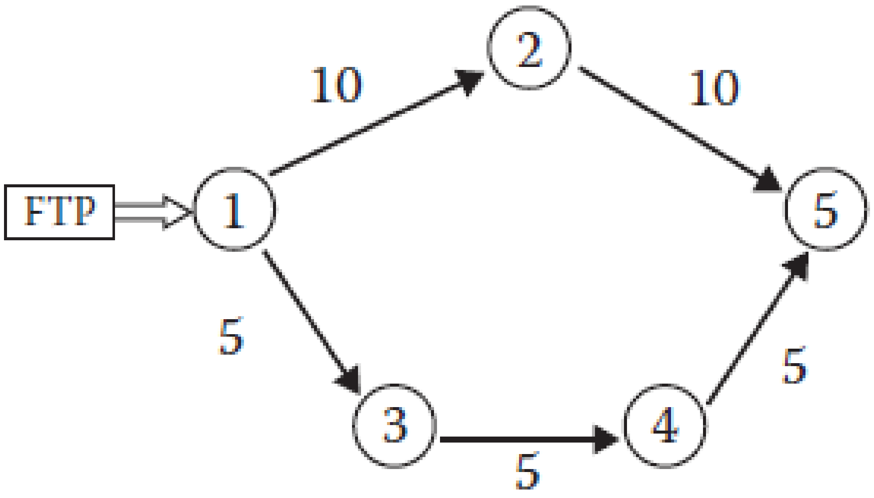
subject to

$$H(X) = 0$$

$$G(X) \geq 0$$

Optimización MultiObjetivo

- Ejemplo:



Optimización MultiObjetivo

- Método “Suma Ponderada”:

Optimize [minimize/maximize]

$$F'(X) = \sum_{i=1}^{n} r_i * f_i(X)$$

subject to

$$H(X) = 0$$

$$G(X) \geq 0$$

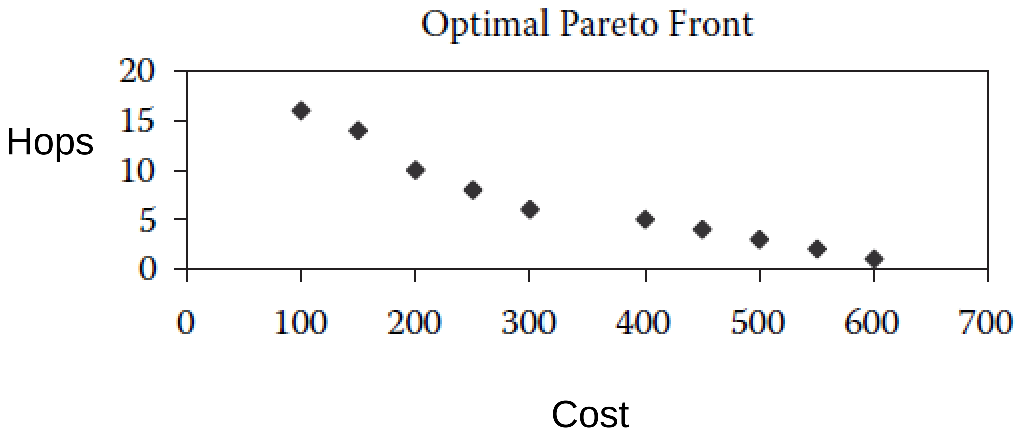
$$0 \leq r_i \leq 1,\ i = \{1,\ \ldots,\ n\}$$

$$\sum_{i=1}^{n} r_i = 1$$

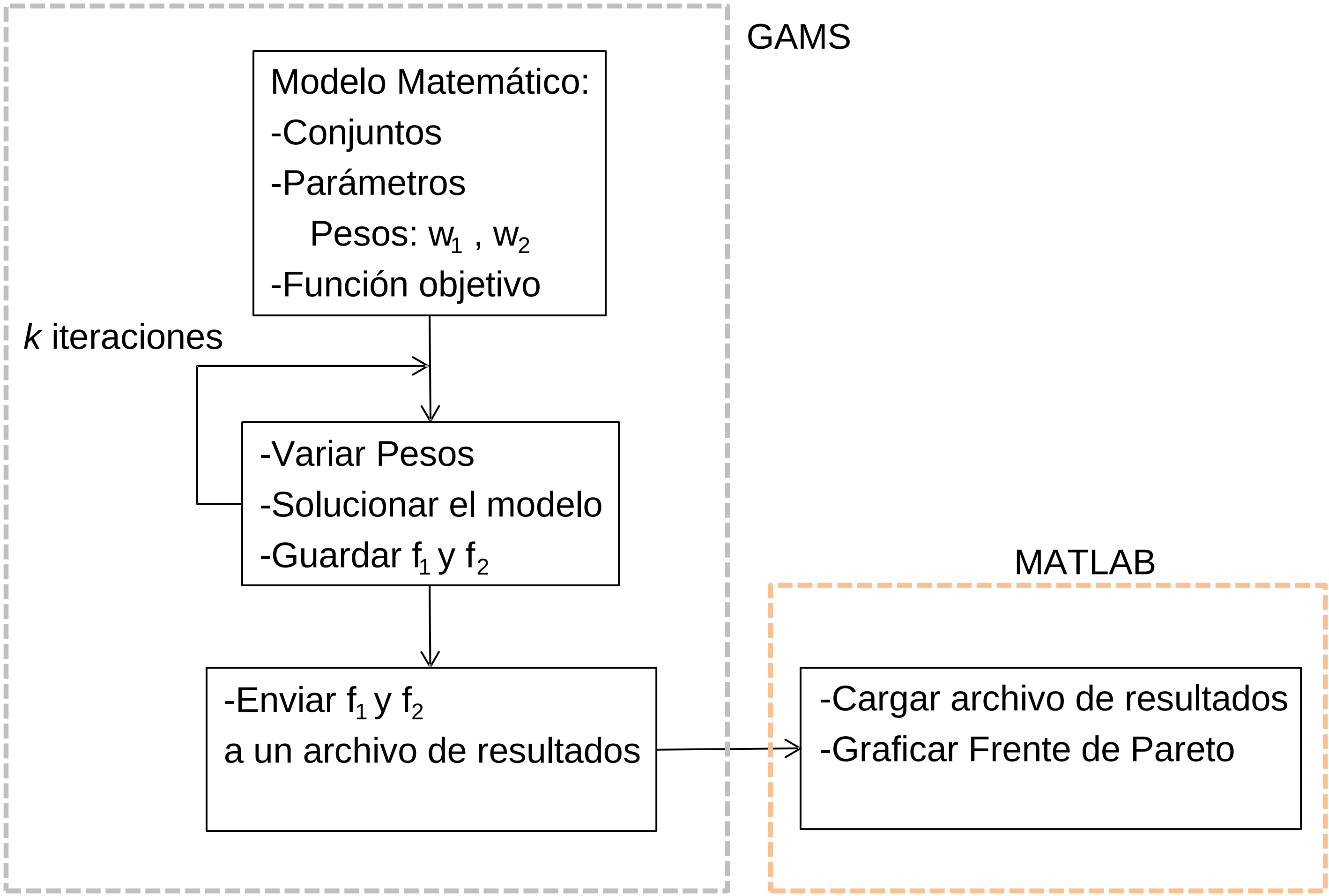
Optimización MultiObjetivo

- Método “Suma Ponderada”:
  - 2 funciones objetivo:

$$F(X) = r_1 \cdot f_1(X) + r_2 \cdot f_2(X)$$

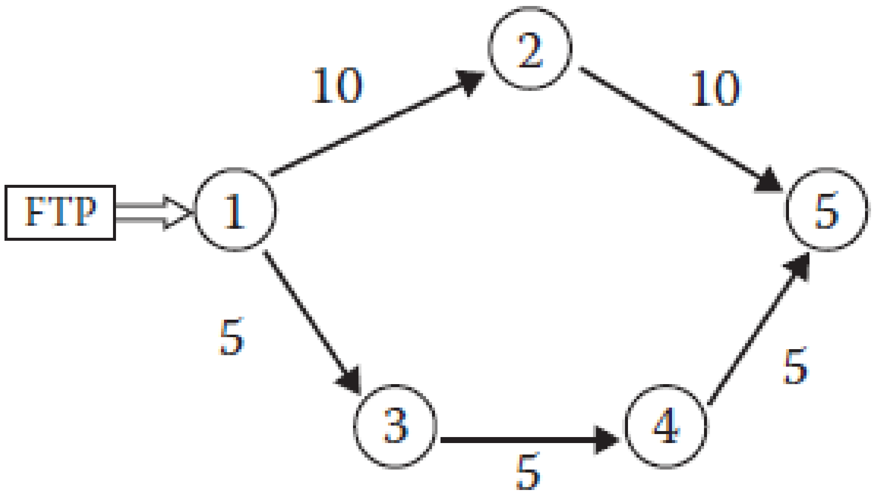


Implementación GAMS/MATLAB



Ejemplo 1

- Caso Shortest Path:
  - Ver archivo multiobjetivoHopsCosts.gms



Ejemplo 1

- Implementación en GAMS:

```
Sets
  i   network nodes / n1, n2, n3, n4, n5 /

alias(j,i);

set iter iteraciones /it1*it11/;

scalar w1 peso 1 / 0 /;
scalar w2 peso 2 / 0 /;

parameter w1_vec(iter) vector de valores de w1
                 /it1 1, it2 0.9, it3 0.8, it4 0.7, it5 0.6, it6 0.5, it7 0.4, it8 0.3, it9 0.2, it10 0.1, it11 0/;

parameter w2_vec(iter)  vector de valores de w1;
```

Universidad de los Andes

Ejemplo 1

- Implementación en GAMS:

```
Table h(i,j) link cost
                  n1       n2       n3       n4       n5
n1                999      1        1        999      999
n2                999      999      999      999      1
n3                999      999      999      1        999
n4                999      999      999      999      1
n5                999      999      999      999      999;

Table c(i,j) link cost
                  n1       n2       n3       n4       n5
n1                999      10       5        999      999
n2                999      999      999      999      10
n3                999      999      999      5        999
n4                999      999      999      999      5
n5                999      999      999      999      999;


Variables
  x(i,j)       Indicates if the link i-j is selected or not.

  z            Objective function

  f1           funcion 1
  f2           funcion 2;

Binary Variable x;
```

Universidad de los Andes

Ejemplo 1

- Implementación en GAMS:

```
Equations
objectiveFunction        objective function
sourceNode(i)            source node
destinationNode(j)       destination node
intermediateNode         intermediate node
valor_f1                 resultado f1
valor_f2                 resultado f2;

valor_f1                    ..      f1=e= sum((i,j), h(i,j) * x(i,j));

valor_f2                    ..      f2=e= sum((i,j), c(i,j) * x(i,j));

objectiveFunction           ..      z =e= w1*f1 + w2*f2;

sourceNode(i)$(ord(i) = 1)                        ..  sum((j), x(i,j)) =e= 1;

destinationNode(j)$(ord(j) = 5)                   ..  sum((i), x(i,j)) =e= 1;

intermediateNode(i)$(ord(i) <> 1 and ord(i) ne 5)  ..  sum((j), x(i,j)) - sum((j), x(j,i)) =e= 0;
```

Ejemplo 1

- Implementación en GAMS:

```
Model model1 /all/ ;

parameter z_res(iter) "z results to store";

parameter f1_res(iter) "f1 results to store";

parameter f2_res(iter) "f2 results to store";

parameter x_res(i,j,iter) "x results to store";

loop (iter,
    w1=w1_vec(iter);
    w2=1 - w1_vec(iter);
    w2_vec(iter)=w2;

    option mip=CPLEX;
    Solve model1 using mip minimizing z;
    z_res(iter)=z.l;
    f1_res(iter)=f1.l;
    f2_res(iter)=f2.l;
    x_res(i,j,iter)=x.l(i,j);

    );
```

Universidad de los Andes

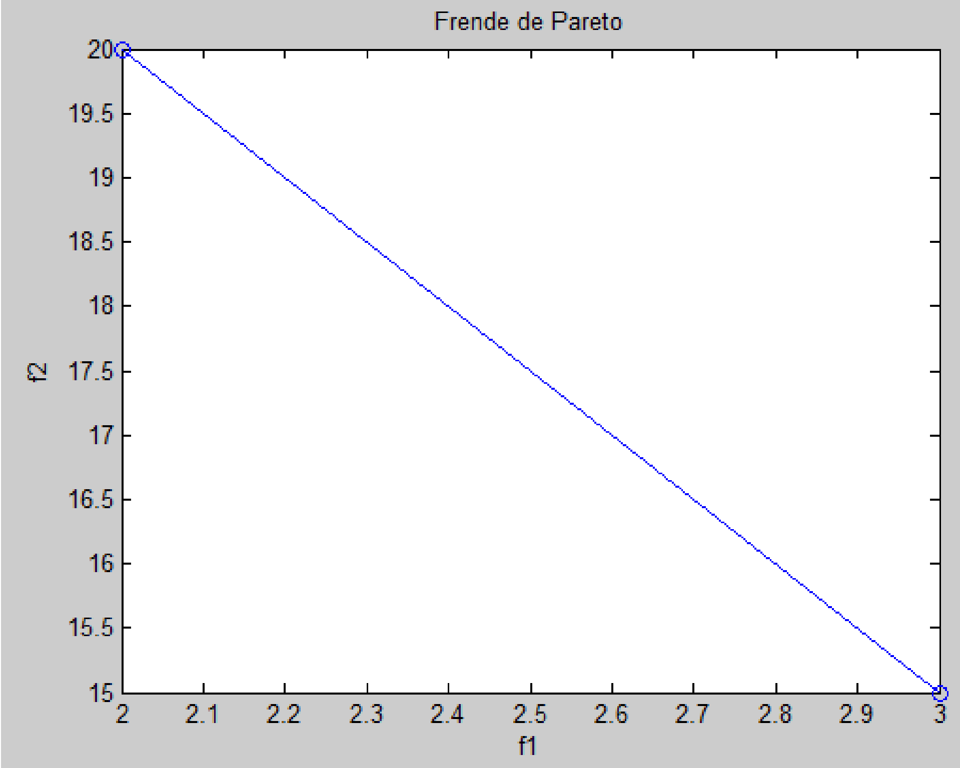
Ejemplo 1

- Implementación en GAMS:

```
display z_res;
display f1_res;
display f2_res;
display w1_vec;
display w2_vec;
display x_res;

file GAMSresults /D:\MSO\hopsCostGAMSresults_v1.dat/;

put GAMSresults;
loop(iter,
         put iter.tl, @5, f1_res(iter), @18, f2_res(iter) /

         );
```

Ejemplo 1

- Gráfica del frente de Pareto en MATLAB:
  - Descargar pareto.m para cargar el *.dat arrojado por GAMS.



Actividad en Casa

- Verificar que el anterior ejemplo funcione en GAMS y Matlab.

Ejemplo 2

- Implementar en GAMS el siguiente caso multiobjetivo:
  - Ver archivo multiobjetivoProcesadores.gms.
  - En una red de telecomunicaciones existen 2 tipos de paquetes: sin prioridad y con prioridad. La red posee 3 nodos origen y 4 nodos destino. Desde los 3 nodos origen se requiere enviar 60, 80 y 50 paquetes sin prioridad, respectivamente, y 20, 20 y 30 paquetes con prioridad respectivamente. En los nodos destino se requieren 50, 90, 40 y 10 paquetes sin prioridad respectivamente, y 10, 20, 10 y 30 paquetes con prioridad, respectivamente. A continuación se muestran los costos de envío por paquete de los nodos orígenes a los destinos:

Nodo Origen	Destino 1	Destino 2	Destino 3	Destino 4
1	10	9	10	11
2	9	10	11	10
3	11	9	10	10

Ejemplo 2

- Implementar en GAMS el siguiente caso multiobjetivo:
  - Adicionalmente es necesario tener en cuenta el tiempo de envío, el cual se muestra a continuación:

Nodo Origen	Destino 1	Destino 2	Destino 3	Destino 4
1	12	14	10	11
2	11	8	7	13
3	6	11	4	15

  - Plantee un problema de optimización multiobjetivo que minimice el costo total de transporte así como el tiempo de envío, encontrando el frente de Pareto mediante el método de Sumas Ponderadas. Implemente el modelo en GAMS para obtener la solución óptima del problema.

Universidad de los Andes

Ejemplo 2

- Implementación en GAMS:

```
Set i   packet types / p1, p2 /;
Set j   source nodes  / s1, s2, s3 /;
Set k   destination nodes / d1, d2, d3, d4 /;

set iter iterations /it1*it11/;
scalar w1 weight 1 / 0 /;
scalar w2 weight 2 / 0 /;

parameter w1_vec(iter) w1 values
                 /it1 1, it2 0.9, it3 0.8, it4 0.7, it5 0.6, it6 0.5,
                  it7 0.4, it8 0.3, it9 0.2, it10 0.1, it11 0/;
parameter w2_vec(iter) w2 values;

Table c(j,k) sending cost
                  d1       d2      d3       d4
s1               10        9      10       11
s2                9       10      11       10
s3               11        9      10       10;

Table t(j,k) sending delay
                 d1       d2      d3       d4
s1               12       14      10       11
s2               11        8       7       13
s3                6       11       4       15;

Table inv(i,j) inventory
                  s1       s2       s3
p1                60       80       50
p2                20       20       30;

Table dem(i,k) demand
                 d1      d2      d3       d4
p1               50      90      40       10
p2               10      20      10       30;
```

Ejemplo 2

- Implementación en GAMS:

```
Variables
  x(i,j,k)     Amount of i type packets sent from the source node j
               to the destination node k.
  z            minimization
  f1           function 1
  f2           function 2;

Positive Variable x;

Equations
funObj                          Objective Function

invConstraint(i,j)              inventory constraint

demConstraint(i,k)              demand constraint

f1_value                        f1 value
f2_value                        f2 value;

f1_value                ..      f1=e= sum((i,j,k), c(j,k) * x(i,j,k));

f2_value                ..      f2=e= sum((i,j,k), t(j,k) * x(i,j,k));

funObj                  ..      z =e= w1*f1 + w2*f2;

invConstraint(i,j)      ..      sum((k), x(i,j,k)) =l= inv(i,j);

demConstraint(i,k)      ..      sum((j), x(i,j,k)) =e= dem(i,k);
```

Ejemplo 2

- Implementación en GAMS:

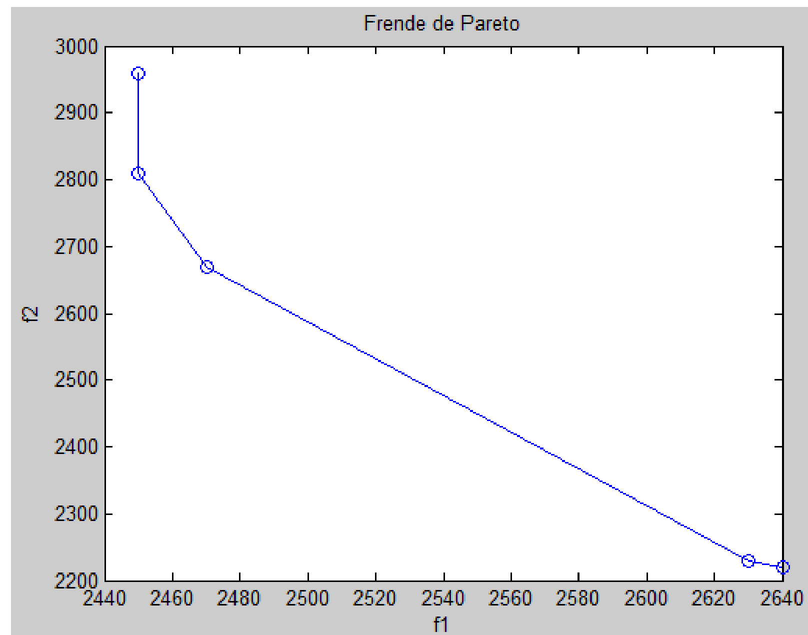
```
Model Model1 /all/ ;

parameter z_res(iter) "z results to store";
parameter f1_res(iter) "f1 results to store";
parameter f2_res(iter) "f2 results to store";
parameter x_res(i,j,k,iter) "x results to store";

loop (iter,
    w1=w1_vec(iter);
    w2=1 - w1_vec(iter);
    w2_vec(iter)=w2;

    option lp=CPLEX;
    Solve Model1 using lp minimizing z;
    z_res(iter)=z.l;
    f1_res(iter)=f1.l;
    f2_res(iter)=f2.l;
    x_res(i,j,k,iter)=x.l(i,j,k);
    );

display z_res;
display f1_res;
display f2_res;
display w1_vec;
display w2_vec;
display x_res;

file GAMSresults /D:\g_Doctorado\Docencia\monitoria modelado simulación y optimización\presentaciones clase\codes\results.dat/;
put GAMSresults;
loop(iter,
         put iter.tl, @5, f1_res(iter), @18, f2_res(iter) /

         );
```

Ejemplo 2

- Cómo debería resultar la Gráfica del frente de Pareto en MATLAB?

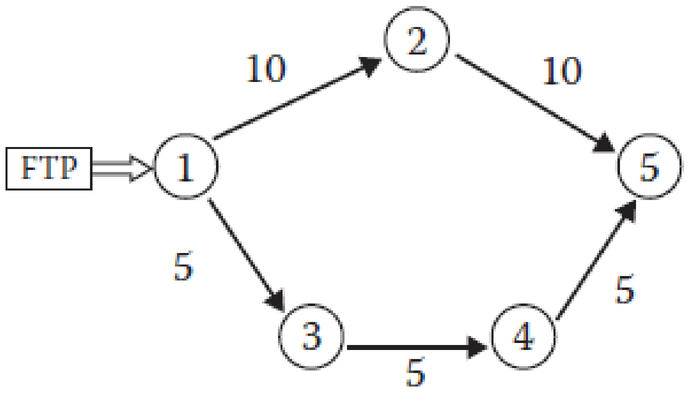
Editor - D:\g_Doctorado\Docencia\monitoria modelado simulaci

```
clc, clear all, close all

[iter, f1, f2] = textread('TransPaqGAMSresults_v1.dat', '%s %f %f', 20);

figure
plot(f1,f2,'-o')
title('Frende de Pareto')
xlabel('f1')
ylabel('f2')
```

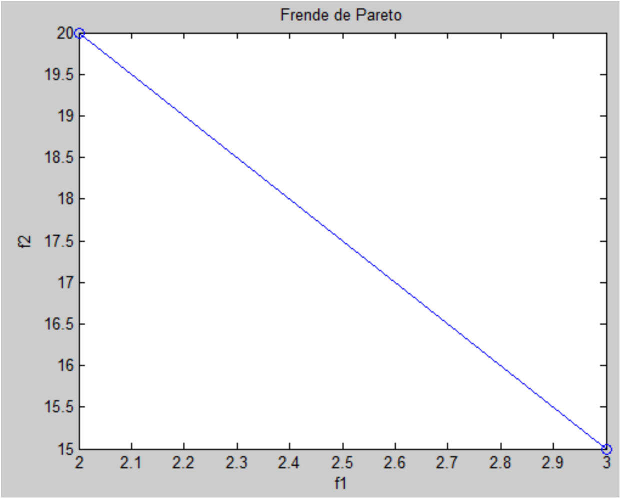


Optimización MultiObjetivo

- En GAMS resolvimos el siguiente caso:



  – El frente de Pareto fué:

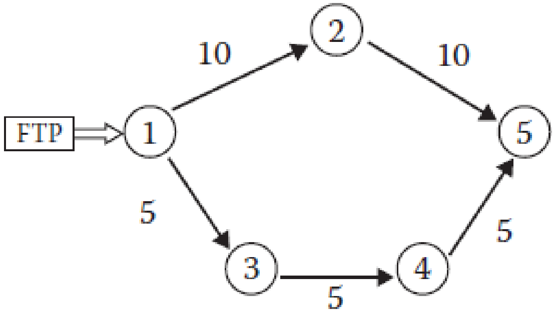


Universidad de los Andes

Optimización MultiObjetivo

- Vamos a obtener el mismo frente en Pyomo:
  - Código fuente: multiobjetivoHopsCosts_sumasPonderadas.py
  - En otras palabras, vamos a ejecutar varias veces el modelo de manera automática.

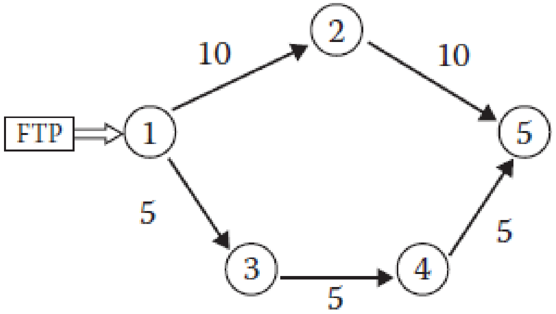
```python
##############################################################################
#####################        LIBRERÍAS        ##############################
##############################################################################

#Plot Imports
import matplotlib.pyplot as plt

#Pyomo Imports (Modelo Matematico)
from pyomo.environ import *
from pyomo.opt import SolverFactory

##############################################################################
#####################        FUNCIONES        ##############################
##############################################################################

#FUNCION ELIMINAR COMPONENTE
def delete_component(Model, comp_name):

        list_del = [vr for vr in vars(Model)
                    if comp_name == vr
                    or vr.startswith(comp_name + '_index')
                    or vr.startswith(comp_name + '_domain')]

        list_del_str = ', '.join(list_del)
        print('Deleting model components ({}).'.format(list_del_str))

        for kk in list_del:
            Model.del_component(kk)

```



Optimización MultiObjetivo

- Vamos a obtener el mismo frente en Pyomo:
  - En otras palabras, vamos a ejecutar varias veces el modelo de manera automática.

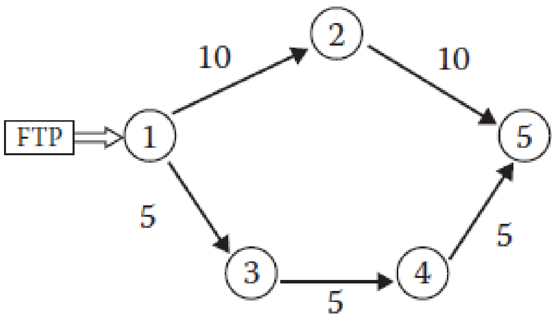
```
###################################################
#####################        MODELO           #####
###################################################

#Configuración Iteraciones-------------------------
numIteraciones=11
iteraciones=range(numIteraciones)
w2_vec=[]
for i in iteraciones:
    valorIter1=i/(numIteraciones-1)
    w2_vec.append(valorIter1)

w1=0
w2=0

#Creación Modelo-----------------------------------
Model = ConcreteModel()

#sets & parameters---------------------------------
numNodes = 5
Model.N=RangeSet(1,numNodes)
```



Optimización MultiObjetivo

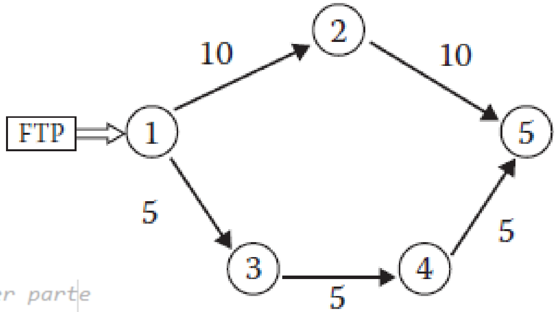
- Vamos a obtener el mismo frente en Pyomo:
  - En otras palabras, vamos a ejecutar varias veces el modelo de manera automática.

```
#hops-----------------------------------------------
Model.h =Param(Model.N, Model.N, mutable=True)

for i in Model.N:
    for j in Model.N:
        Model.h[i,j] = 999


Model.h[1,2] = 1
Model.h[1,3] = 1
Model.h[2,5] = 1
Model.h[3,4] = 1
Model.h[4,5] = 1

#costos---------------------------------------------
Model.c =Param(Model.N, Model.N, mutable=True)

for i in Model.N:
    for j in Model.N:
        Model.c[i,j] = 999

Model.c[1,2] = 10
Model.c[1,3] = 5
Model.c[2,5] = 10
Model.c[3,4] = 5
Model.c[4,5] = 5

#origen y destino-----------------------------------

s = 1
d = 5
```



Optimización MultiObjetivo

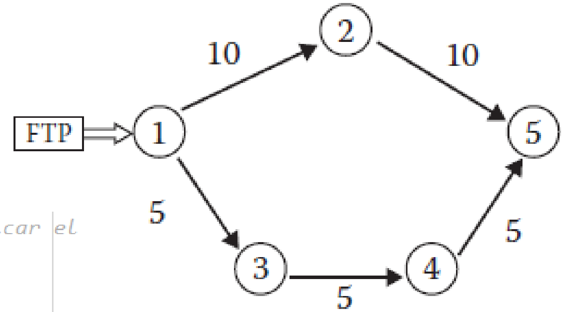
- Vamos a obtener el mismo frente en Pyomo:
  - En otras palabras, vamos a ejecutar varias veces el modelo de manera automática.




```
#Variable binaria que indica si el enlace (i,j) es seleccionado para hacer parte
#del camino que va del nodo fuente al nodo destino.
Model.x = Var(Model.N,Model.N, domain=Binary)


# # OBJECTIVE FUNCTIONS*************************************************************

#Función hops
Model.f1 = sum(Model.x[i,j] * Model.h[i,j] for i in Model.N for j in Model.N)

#Función de costos
Model.f2 = sum(Model.x[i,j] * Model.c[i,j] for i in Model.N for j in Model.N)
```

Optimización MultiObjetivo

- Vamos a obtener el mismo frente en Pyomo:
  - En otras palabras, vamos a ejecutar varias veces el modelo de manera automática.

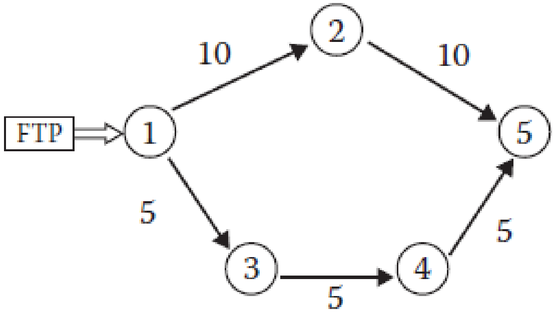



```
#Proceso para ejecutar varias veces el modelo matemático con el fin de aplicar el
#método de Smas Ponderadas.
cont=-1
f1_vec=[]
f2_vec=[]
for k in w2_vec:
    cont=cont+1
    w2=w2_vec[cont]
    w1=1-w2

    #Función objetivo general
    Model.O_z = Objective(expr= w1*Model.f1 + w2*Model.f2, sense=minimize)
```

Optimización MultiObjetivo

- Vamos a obtener el mismo frente en Pyomo:
  - En otras palabras, vamos a ejecutar varias veces el modelo de manera automática.

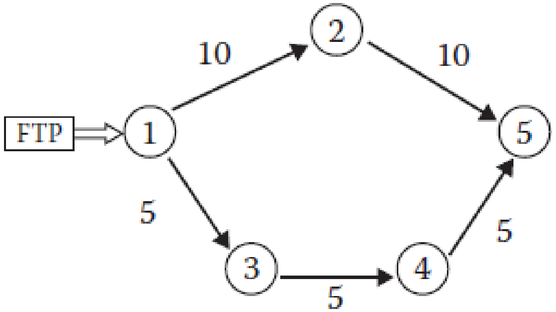
```python
##Restricción nodo origen
def source_rule(Model,i):
    if i==s:
        return sum(Model.x[i,j] for j in Model.N)==1
    else:
        return Constraint.Skip

Model.source=Constraint(Model.N, rule=source_rule)

#Restricción nodo destino
def destination_rule(Model,j):
    if j==d:
        return sum(Model.x[i,j] for i in Model.N)==1
    else:
        return Constraint.Skip

Model.destination=Constraint(Model.N, rule=destination_rule)

#Restricción nodo intermedio
def intermediate_rule(Model,i):
    if i!=s and i!=d:
        return sum(Model.x[i,j] for j in Model.N) - sum(Model.x[j,i] for j in Model.N)==0
    else:
        return Constraint.Skip

Model.intermediate=Constraint(Model.N, rule=intermediate_rule)
```



Optimización MultiObjetivo

- Vamos a obtener el mismo frente en Pyomo:
  - En otras palabras, vamos a ejecutar varias veces el modelo de manera automática.

```
    SolverFactory('glpk').solve(Model)

    valorF1=value(Model.f1)
    valorF2=value(Model.f2)
    f1_vec.append(valorF1)
    f2_vec.append(valorF2)

    delete_component(Model, 'O_z')
    delete_component(Model, 'source')
    delete_component(Model, 'destination')
    delete_component(Model, 'intermediate')

    #end for
```

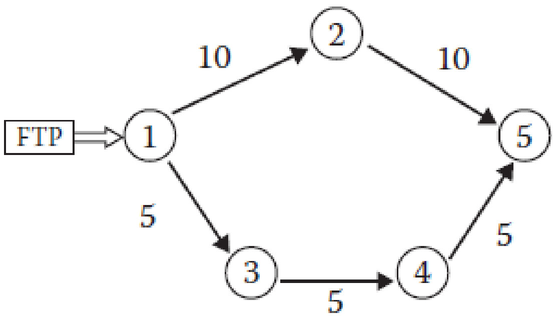


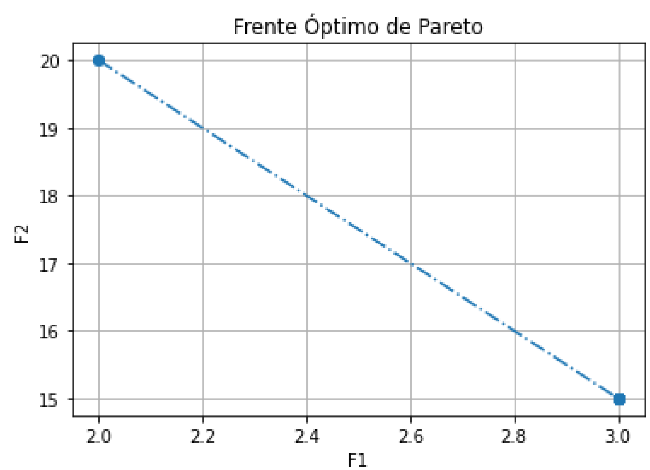
Esto es obligatorio, sino generaría errores en Pyomo!

Optimización MultiObjetivo

- Vamos a obtener el mismo frente en Pyomo:
  - En otras palabras, vamos a ejecutar varias veces el modelo de manera automática.

```
plt.plot(f1_vec,f2_vec,'o-.');
plt.title('Frente Óptimo de Pareto');
plt.xlabel('F1')
plt.ylabel('F2')

plt.grid(True);
plt.show()
```





Optimización MultiObjetivo

- Tips:
  - Existen otros métodos para encontrar el frente óptimo de Pareto.
    - Método de sumas ponderadas.
    - **Método de e-constraint.**
    - Método de metricas con pesos.
    - Método de Benson.

Optimización MultiObjetivo

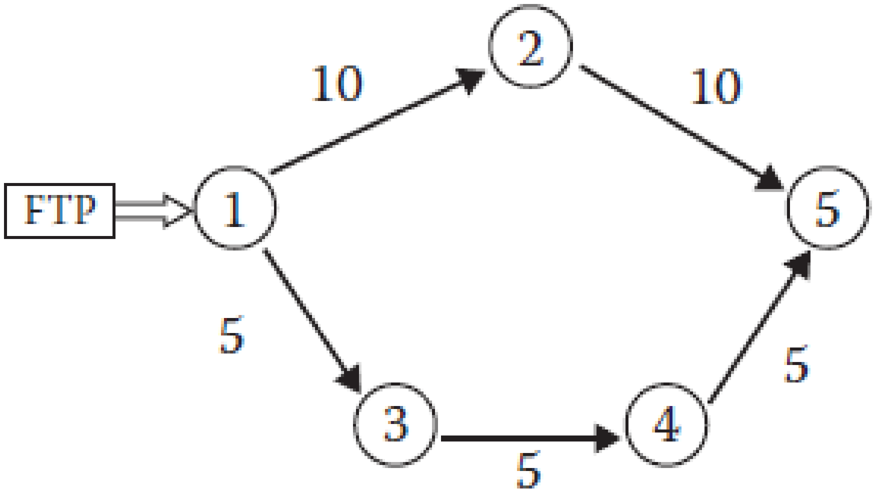
- Tips:
  - Existen otros métodos para encontrar el frente óptimo de Pareto.
    - **Método de e-constraint.**

$$
\begin{array}{ll}
\max & (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_p(\mathbf{x})) \\
\text{st} & \\
 & \mathbf{x} \in S,
\end{array}
\quad \Longrightarrow \quad
\begin{array}{ll}
\max & f_1(\mathbf{x}) \\
\text{st} & \\
 & f_2(\mathbf{x}) \geqslant e_2, \\
 & f_3(\mathbf{x}) \geqslant e_3, \\
 & \ldots \\
 & f_p(\mathbf{x}) \geqslant e_p, \\
 & \mathbf{x} \in S.
\end{array}
$$

Nota: Si minimizamos, entonces el operador Relacional sería ‘menor o igual’ (<=).

Optimización MultiObjetivo

- Tips:
  - Existen otros métodos para encontrar el frente óptimo de Pareto.
    - **Método de e-constraint.**
      - **Ejemplo: minimizar costo y minimizar hops**



Recordemos:

Método de Sumas Ponderadas

$$F(X) = r_1 \cdot f_1(X) + r_2 \cdot f_2(X)$$

$$H(X) = 0$$

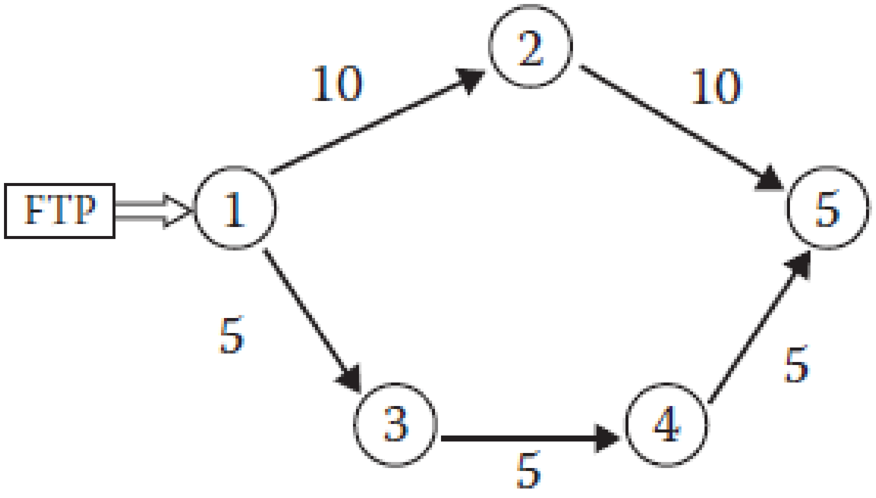
$$G(X) \geq 0$$

$$0 \leq r_i \leq 1,\ i = \{1, \ldots, n\}$$

$$\sum_{i=1}^{n} r_i = 1$$

Optimización MultiObjetivo

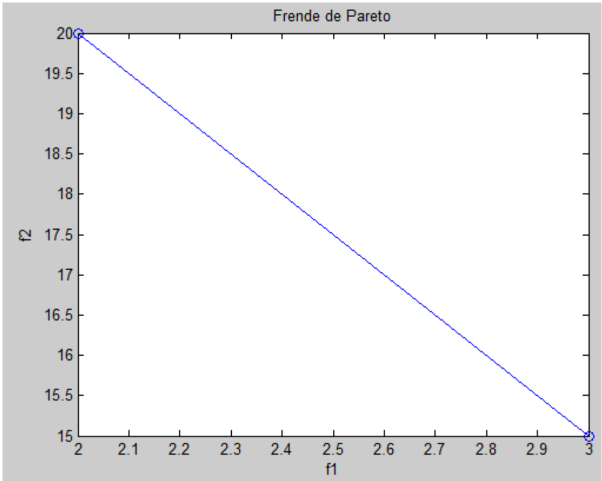
- Tips:
  - Existen otros métodos para encontrar el frente óptimo de Pareto.
    - **Método de e-constraint.**
      - **Ejemplo: minimizar costo y minimizar hops**



Recordemos:

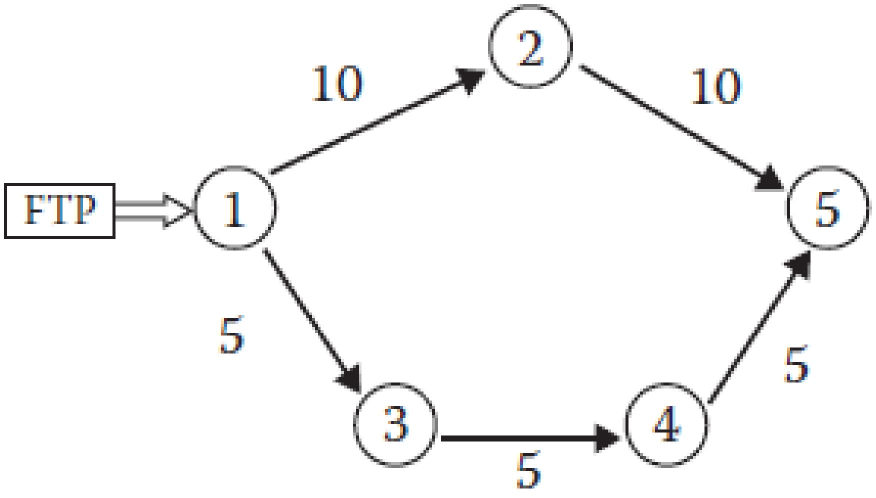
Método de Sumas Ponderadas

Frente óptimo de Pareto:



Optimización MultiObjetivo

- Tips:
  – Existen otros métodos para encontrar el frente óptimo de Pareto.
    - **Método de e-constraint.**
      – **Ejemplo: minimizar costo y minimizar hops**



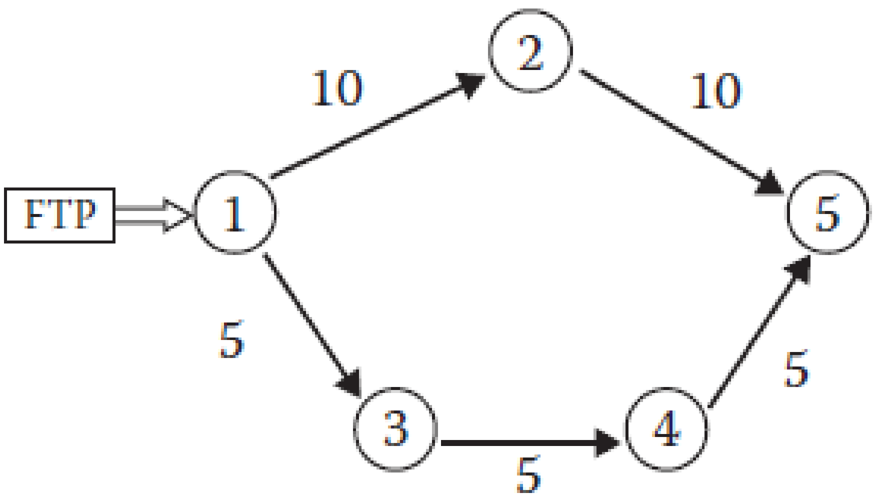
Aplicando el método de e-constraint:

```
Sets
  i   network nodes / n1, n2, n3, n4, n5 /

alias(j,i);

Table h(i,j) link capacity
                  n1       n2     n3      n4      n5
n1               999        1      1     999     999
n2               999      999    999     999       1
n3               999      999    999       1     999
n4               999      999    999     999       1
n5               999      999    999     999     999;

Table c(i,j) link cost
                 n1       n2       n3       n4       n5
n1               999      10       5        999      999
n2               999      999      999      999      10
n3               999      999      999      5        999
n4               999      999      999      999      5
n5               999      999      999      999      999;
```

Optimización MultiObjetivo

- Tips:
  - Existen otros métodos para encontrar el frente óptimo de Pareto.
    - **Método de e-constraint.**
      - **Ejemplo: minimizar costo y minimizar hops**

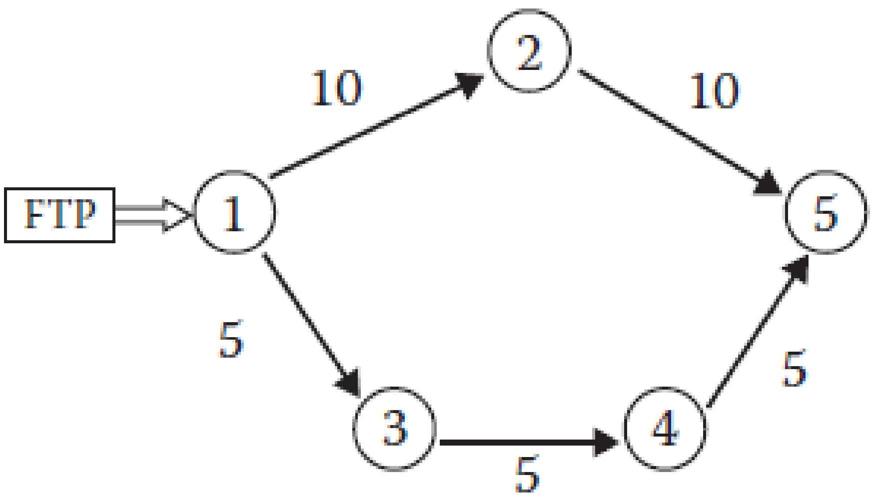
Aplicando el método de e-constraint: Empezamos con f1 <= 5 y vamos decrementando el 5 hasta obtener distintas soluciones.




```
valor_f1                    ..      f1=e= sum((i,j), h(i,j) * x(i,j));

valor_f2                    ..      f2=e= sum((i,j), c(i,j) * x(i,j));

objectiveFunction           ..      z =e= f2;

sourceNode(i)$(ord(i) = 1)   ..  sum((j), x(i,j)) =e= 1;

destinationNode(j)$(ord(j) = 5)  ..  sum((i), x(i,j)) =e= 1;

intermediateNode(i)$(ord(i) <> 1 and ord(i) ne 5)
..  sum((j), x(i,j)) - sum((j), x(j,i)) =e= 0;

f1constraint                    ..  f1 =l= 5;

Model model1 /all/ ;

option mip=CPLEX;
Solve model1 using mip minimizing z;
```

Optimización MultiObjetivo

- Tips:
  - Existen otros métodos para encontrar el frente óptimo de Pareto.
    - **Método de e-constraint.**
      - **Ejemplo: minimizar costo y minimizar hops**

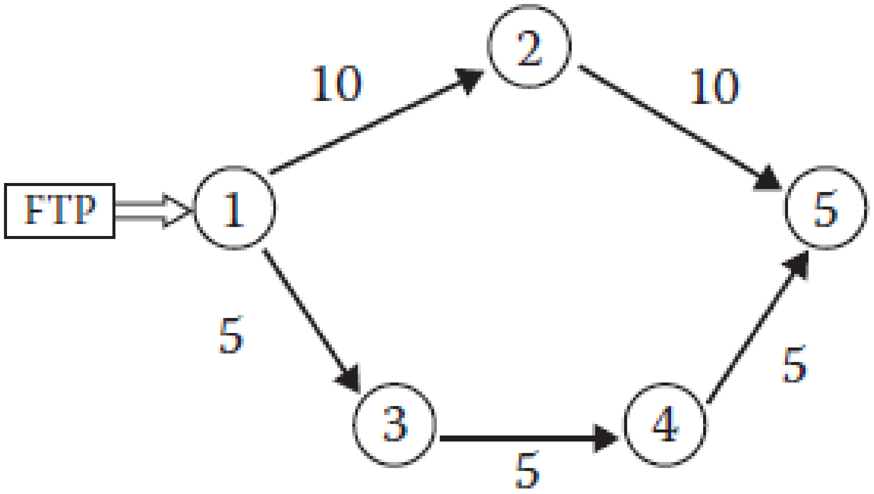


Aplicando el método de e-constraint: Empezamos con f1 <= 5 y vamos decrementando el 5 hasta obtener distintas soluciones.

```
72 VARIABLE f1.L          =       3.000  funcion 1
73 VARIABLE f2.L          =      15.000  funcion 2
```

Optimización MultiObjetivo

- Tips:
  - Existen otros métodos para encontrar el frente óptimo de Pareto.
    - **Método de e-constraint.**
      - **Ejemplo: minimizar costo y minimizar hops**



Aplicandoel método de e-constraint:

Seguimos decrementando:

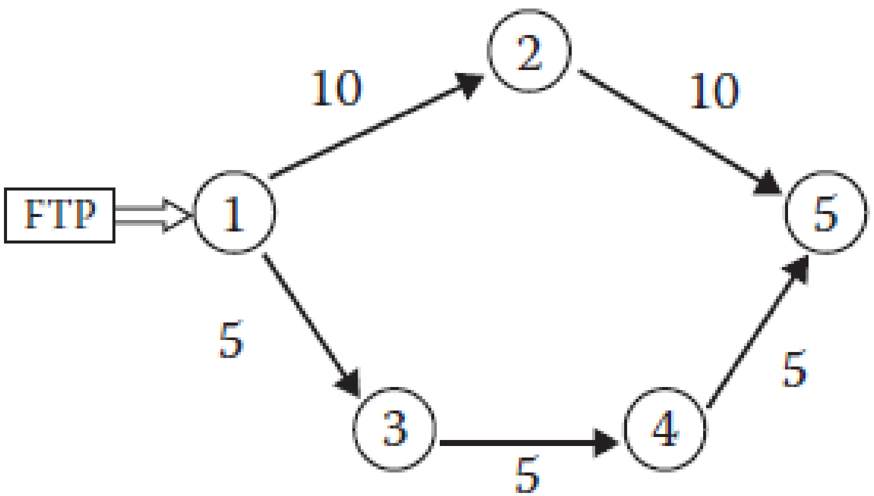
f1 <= 4.

```
72 VARIABLE f1.L          =        3.000  funcion 1
73 VARIABLE f2.L          =       15.000  funcion 2
```

Optimización MultiObjetivo

- Tips:
  - Existen otros métodos para encontrar el frente óptimo de Pareto.
    - **Método de e-constraint.**
      - **Ejemplo: minimizar costo y minimizar hops**

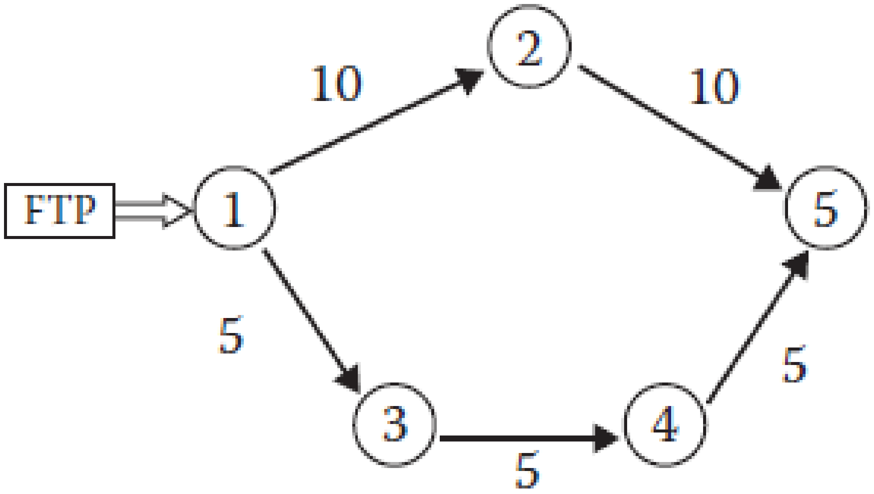
Aplicando el método de e-constraint:
Seguimos decrementando:
f1 <= 3.




```
72 VARIABLE f1.L          =        3.000  funcion 1
73 VARIABLE f2.L          =       15.000  funcion 2
```

Optimización MultiObjetivo

- Tips:
  - Existen otros métodos para encontrar el frente óptimo de Pareto.
    - **Método de e-constraint.**
      - **Ejemplo: minimizar costo y minimizar hops**

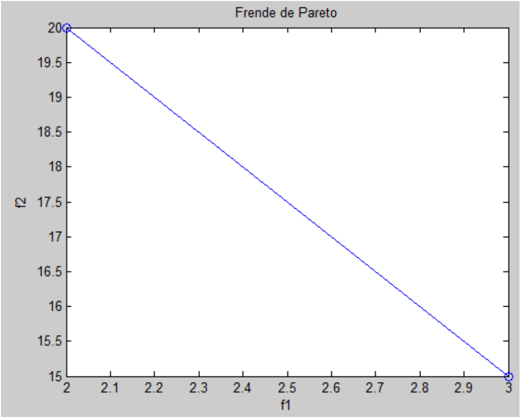


Aplicandoel método de e-constraint:

Seguimos decrementando:

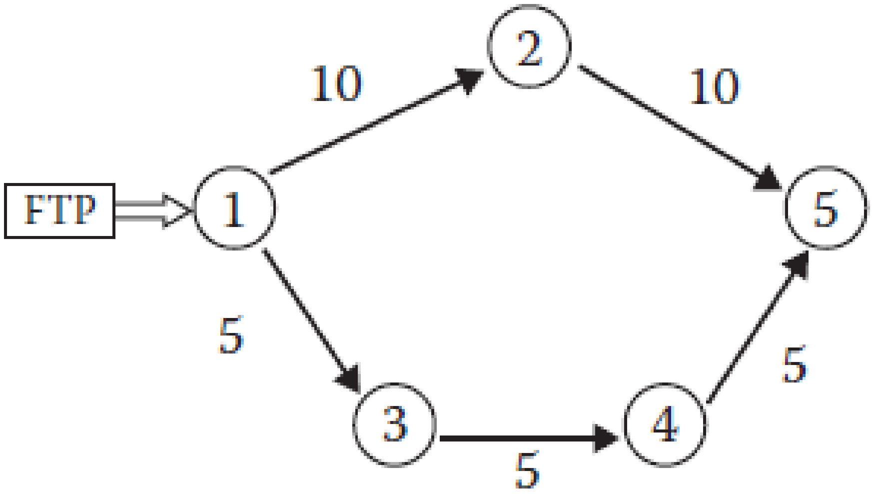
f1 <= 2.

```
VARIABLE f1.L                 =         2.000  funcion 1
VARIABLE f2.L                 =        20.000  funcion 2
```



Optimización MultiObjetivo

- Tips:
  - Existen otros métodos para encontrar el frente óptimo de Pareto.
    - **Método de e-constraint.**
      - **Ejemplo: minimizar costo y minimizar hops**



Aplicando el método de e-constraint:

Y si seguimos decrementando?

f1 <= 1.

```
MIP status(103): integer infeasible
Cplex Time: 0.00sec (det. 0.02 ticks)
Problem is integer infeasible.
```

El modelo nos resulta **infactible**, por lo tanto, aquí finaliza el método.

Optimización MultiObjetivo

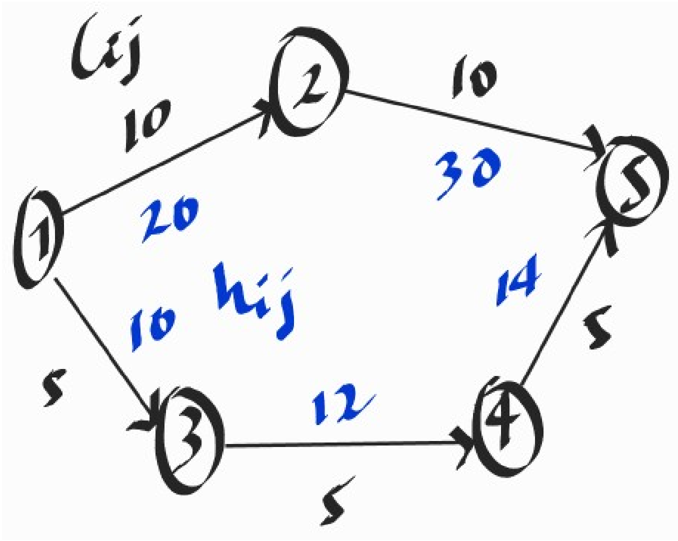
- Actividad en casa:
  - Descargar “multiobjetivoHopsCosts_eConstraint.gms” y obtener los dos puntos del frente óptimo de Pareto previamente vistos.

Optimización MultiObjetivo

- Qué sucede si deseo minimizar una función y maximizar una segunda función?
  - Ejemplos:
    - Minimizar costo y maximizar la capacidad de los enlaces en una red.
    - Minimizar retardo y maximizar cantidad de paquetes a enviar en una red.
    - Minimizar retardo y maximizar ganancias en una red.

Optimización MultiObjetivo

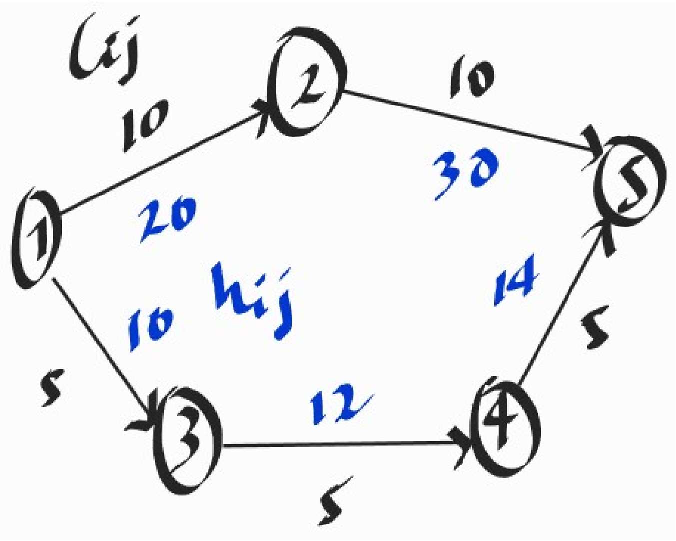
- Qué sucede si deseo maximizar F1 y minimizar F2 en el siguiente caso?
  - Ejemplo: Maximizar la capacidad y minimizar el costo en una red.



Cij: costo del enlace (i,j)
hij:capacidad del enlace (i,j)

Optimización MultiObjetivo

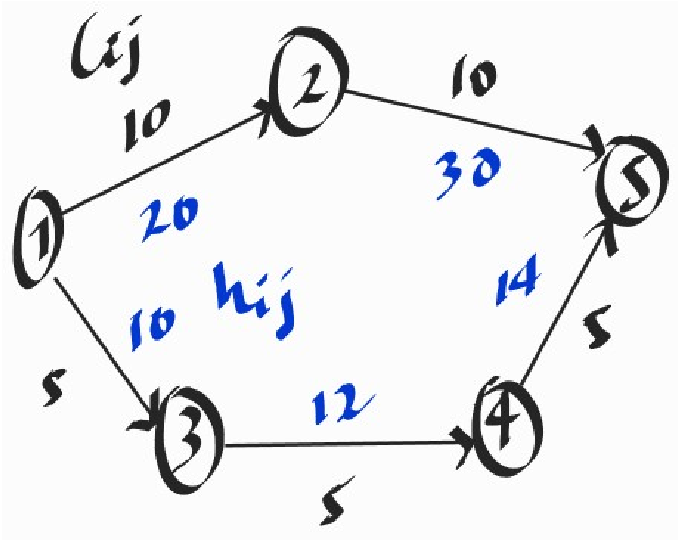
- Ejemplo: Minimizar costo y maximizar la capacidad de los enlaces en una red.
  - Si le damos toda la importancia a la función de costo:
    - La solución sería: 1-3-4-5



Cij: costo del enlace (i,j)
hij:capacidad del enlace (i,j)

Optimización MultiObjetivo

- Ejemplo: Minimizar costo y maximizar la capacidad de los enlaces en una red.
  - Si le damos toda la importancia a la función de capacidad:
    - La solución sería: 1-2-5



Cij: costo del enlace (i,j)
hij:capacidad del enlace (i,j)

Optimización MultiObjetivo

- Ejemplo: Minimizar costo y maximizar la capacidad de los enlaces en una red.
  - Cómo gestionamos una función general que contiene dos funciones contrarias? Es decir, una minimizando y otra maximizando.

    - Función general: min Z= w1*F1 + w2*F2 :(Problema!

      - Si minimizamos la Z, entonces minimizamos F1 y F2.   :(

      - Si maximizamos la Z, entonces maximizamos F1 y F2.   :(

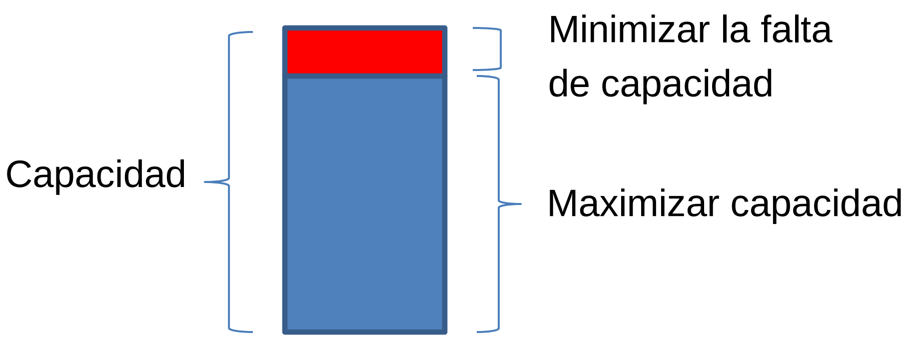
Universidad de los Andes

Optimización MultiObjetivo

- Ejemplo: Minimizar costo y maximizar la capacidad de los enlaces en una red.
  - Cómo gestionamos una función general que contiene dos funciones contrarias? Es decir, una minimizando y otra maximizando.
    - Cómo hacemos entonces para que al minimizar la Z, minimicemos la F2 PERO maximicemos la F1?
      - Es decir, cómo hacemos para que al minimizar la F1, es como si la estuviesemos maximizando?
        - » Solución: minimizando el complemento de lo que significa la F1.

Optimización MultiObjetivo

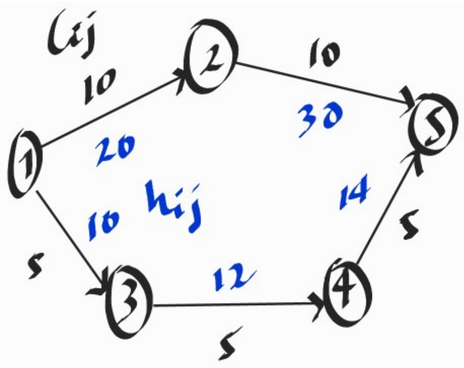
- Ejemplo: Minimizar costo y maximizar la capacidad de los enlaces en una red.
  - Cómo gestionamos una función general que contiene dos funciones contrarias? Es decir, una minimizando y otra maximizando.
    - Solución: minimizando el complemento de lo que significa la F1.



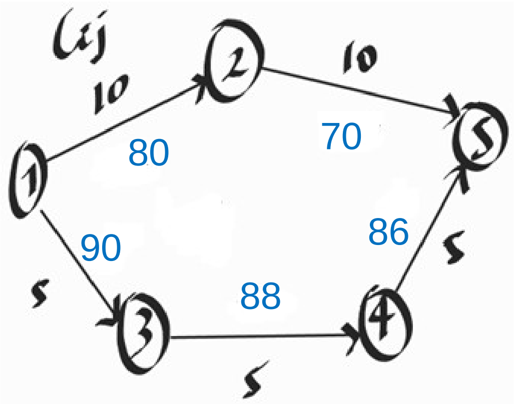
Minimizar la falta de capacidad = Maximizar capacidad

Optimización MultiObjetivo

- Pasos para minimizar el complemento de la capacidad.
  - Paso 1: donde hay enlace, a una cota superior de capacidad le restamos la capacidad actual.
    - Objetivo de este paso: transformar la capacidad en “falta de capacidad”.
    - Valor de la cota superior: corresponde a un valor mayor a la capacidad máxima en el grafo.

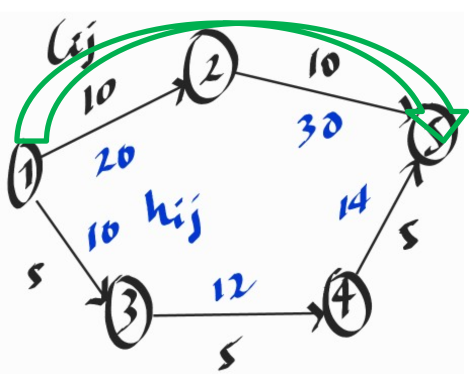


Valor cota superior	>	Máxima capacidad
Valor cota superior	>	30
Valor cota superior	=	100

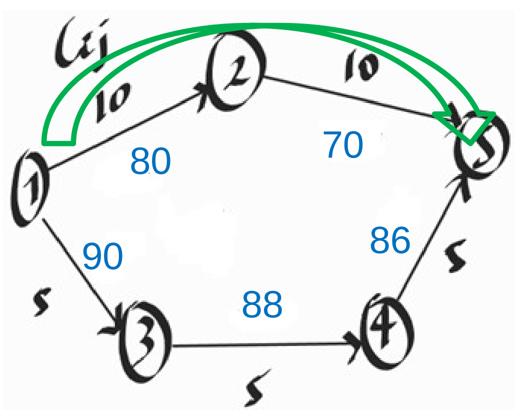


Optimización MultiObjetivo

- Pasos para minimizar el complemento de la capacidad.
  - Consecuencia del Paso 1:
    - Al minimizar la falta de capacidad, estariamos maximizando la capacidad!



Valor cota superior	>	Máxima capacidad
Valor cota superior	>	30
Valor cota superior	=	100



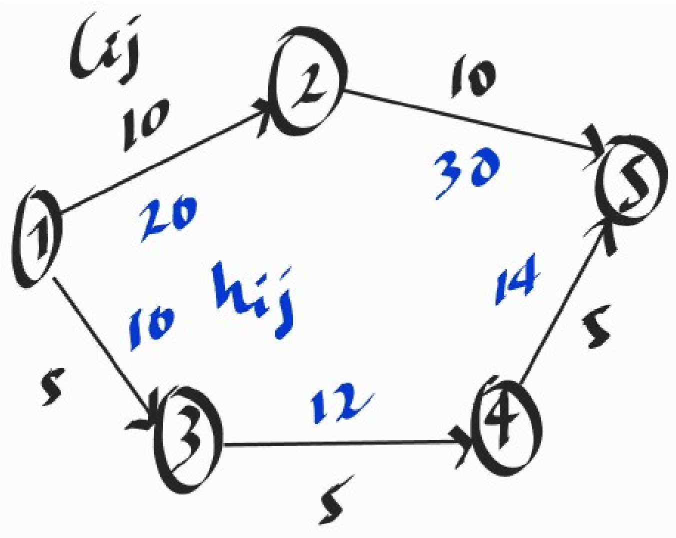
Optimización MultiObjetivo

- Pasos para minimizar el complemento de la capacidad.
  - Paso 2: donde no hay enlace en el grafo, asignamos un valor de falta de capacidad muy alto.
    - Valor alto parecido al método de la Big M visto anteriormente en el curso. Ej: 1 millón.

Optimización MultiObjetivo

- Ejemplo: Minimizar costo y maximizar la capacidad de los enlaces en una red.
  - Implementación: multiobjetivoCapCosts.gms.

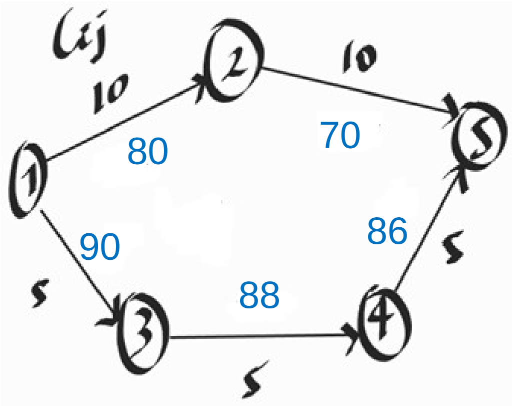
```
Sets
  i   network nodes / n1, n2, n3, n4, n5 /

alias(j,i);

scalar w1 peso 1 / 1 /;
scalar w2 peso 2 / 0 /;

scalar M capacidad donde no hay enlace / 1000000 /;
scalar cota capacidad máxima donde hay enlace / 100 /;

Table h(i,j) link capacity
                  n1      n2     n3      n4      n5
n1                 0      20     10       0       0
n2                 0       0      0       0      30
n3                 0       0      0      12       0
n4                 0       0      0       0      14
n5                 0       0      0       0       0;
```



Optimización MultiObjetivo

- Ejemplo: Minimizar costo y maximizar la capacidad de los enlaces en una red.
  - Implementación → multiobjetivoCapCosts.gms.

```
loop((i,j),
    if(h(i,j)=0,
        h(i,j)=M;
    else
        h(i,j)=cota - h(i,j);
    );
);


Table c(i,j) link cost
                n1      n2      n3      n4      n5
n1              999     10      5       999     999
n2              999     999     999     999     10
n3              999     999     999     5       999
n4              999     999     999     999     5
n5              999     999     999     999     999;
```



Optimización MultiObjetivo

- Ejemplo: Minimizar costo y maximizar la capacidad de los enlaces en una red.
  - Implementación → multiobjetivoCapCosts.gms.

```
Variables
  x(i,j)        Indicates if the link i-j is selected or not.

  z             Objective function

  f1            funcion 1
  f2            funcion 2;

Binary Variable x;
```

```
valor_f1                    ..      f1=e= sum((i,j), h(i,j) * x(i,j));

valor_f2                    ..      f2=e= sum((i,j), c(i,j) * x(i,j));

objectiveFunction           ..      z =e= w1*f1 + w2*f2;

sourceNode(i)$(ord(i) = 1)                     ..  sum((j), x(i,j)) =e= 1;

destinationNode(j)$(ord(j) = 5)                ..  sum((i), x(i,j)) =e= 1;

intermediateNode(i)$(ord(i) <> 1 and ord(i) ne 5)  ..  sum((j), x(i,j)) - sum((j), x(j,i)) =e= 0;
```

Optimización MultiObjetivo

- Ejemplo: Minimizar costo y maximizar la capacidad de los enlaces en una red.
  - Implementación → multiobjetivoCapCosts.gms.

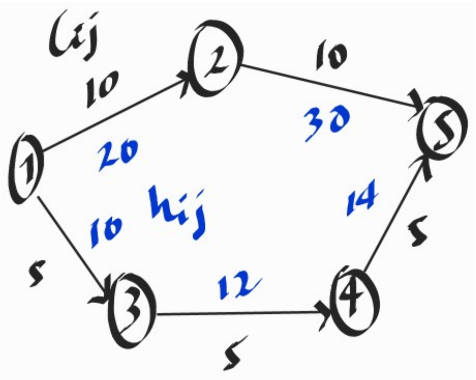
```
Model model1 /all/ ;

option mip=cplex;
Solve model1 using mip minimizing z;

f1.l=sum((i,j), (cota-h(i,j))*x.l(i,j));

display h;
display z.l;
display x.l;
display f1.l;
display f2.l;
```

→ Transformamos la F1 (falta de capacidad) en capacidad, restándose de la cota superior.

Optimización MultiObjetivo

- Ejemplo: Minimizar costo y maximizar la capacidad de los enlaces en una red.
  - Implementación → multiobjetivoCapCosts.gms.
    - Resultados cuando w1=1 y w2=0:

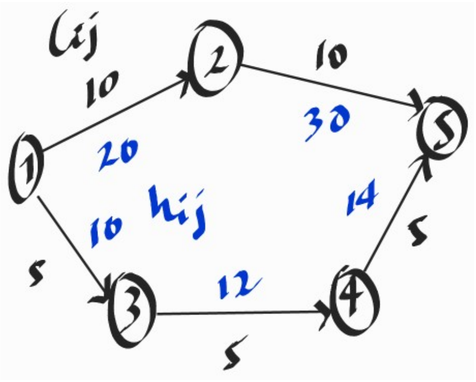
```
----      85 VARIABLE x.L  Indicates if the link i-j is selected or not.

             n2          n5

n1        1.000
n2                    1.000


----      86 VARIABLE f1.L                   =       50.000  funcion 1

----      87 VARIABLE f2.L                   =       20.000  funcion 2
```



Optimización MultiObjetivo

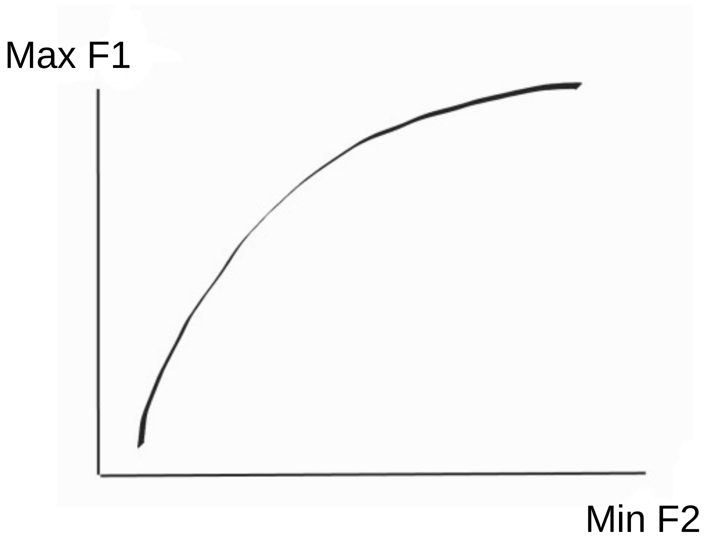
- Ejemplo: Minimizar costo y maximizar la capacidad de los enlaces en una red.
  - Implementación → multiobjetivoCapCosts.gms.
    - Resultados cuando w1=0 y w2=1:

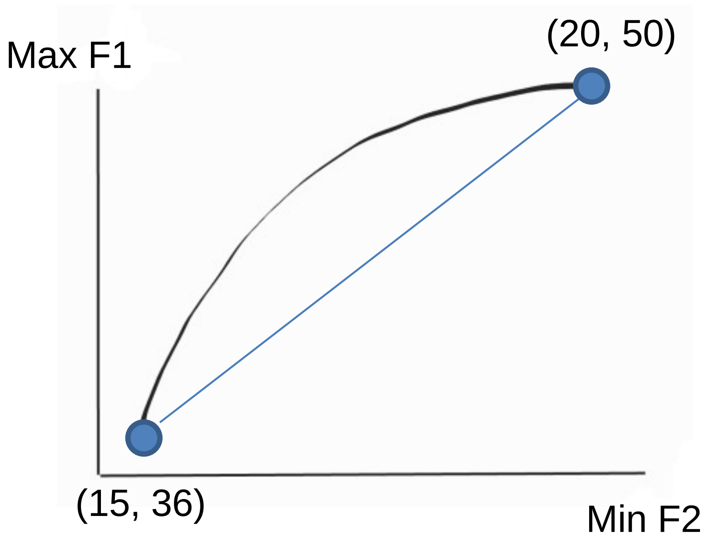
```
----      89 VARIABLE x.L  Indicates if the link i-j is selected or not.

             n3          n4          n5

n1        1.000
n3                    1.000
n4                                1.000


----      90 VARIABLE f1.L                   =        36.000  funcion 1

----      91 VARIABLE f2.L                   =        15.000  funcion 2
```



Optimización MultiObjetivo

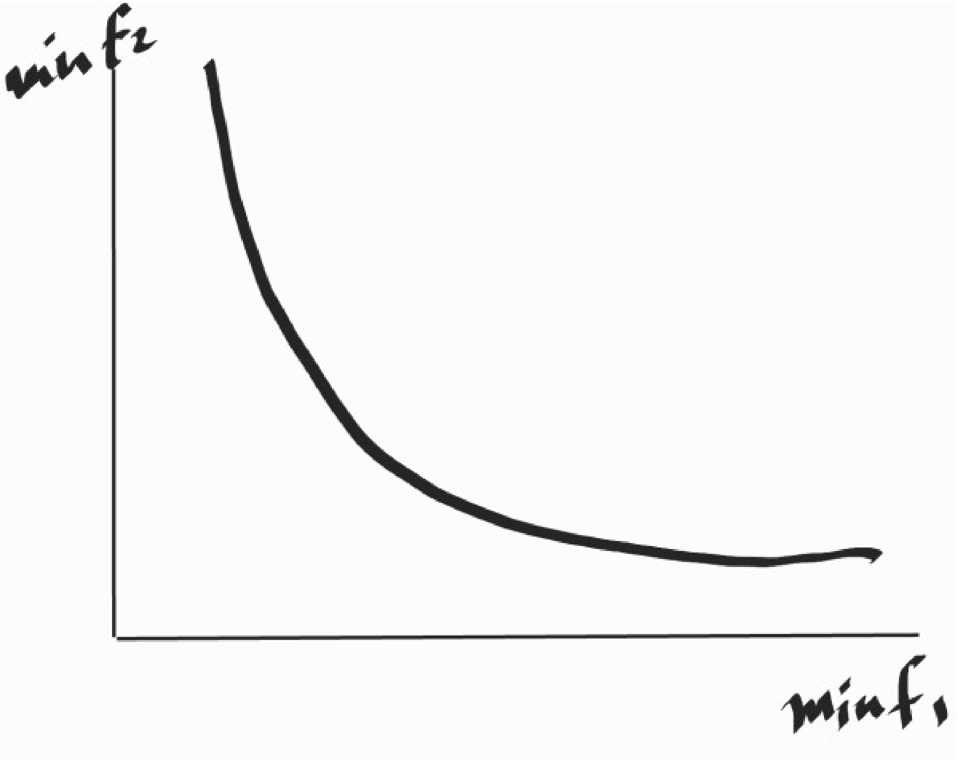
- Ejemplo: Minimizar costo y maximizar la capacidad de los enlaces en una red.
  - Implementación: multiobjetivoCapCosts.gms.
    - Cómo sería el Frente de Pareto?





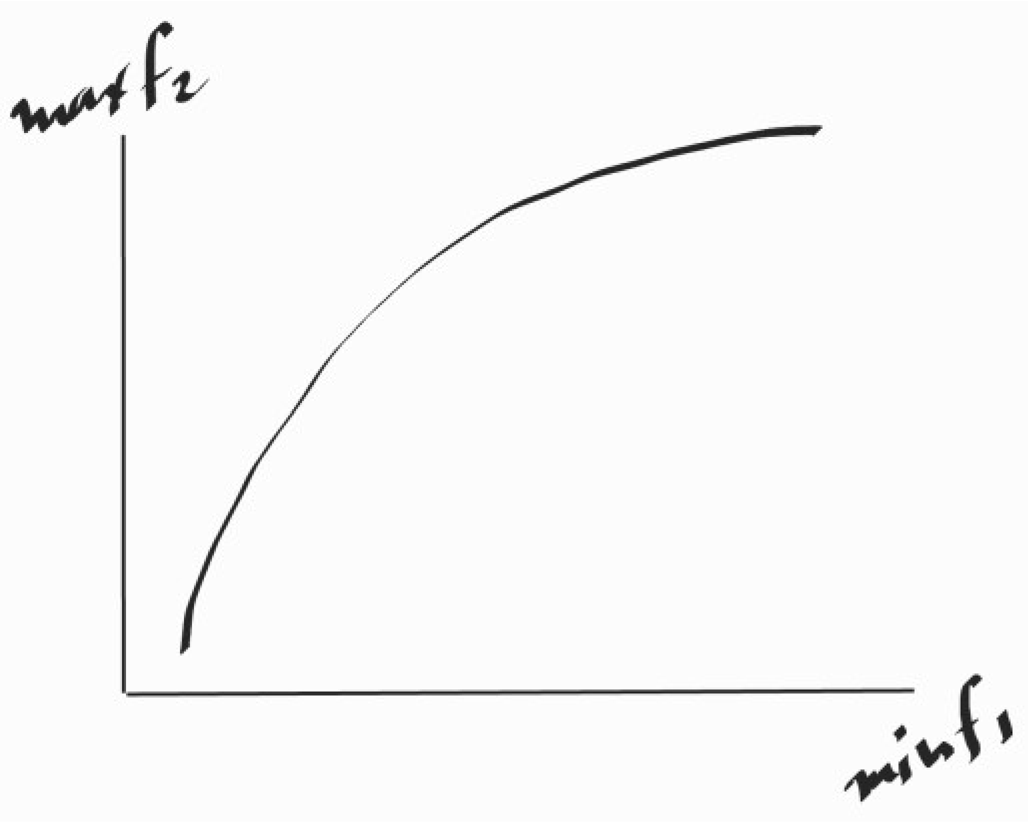
Optimización MultiObjetivo

- Tipos de Frente de Pareto:
  - Cuando minimizamos las dos funciones:



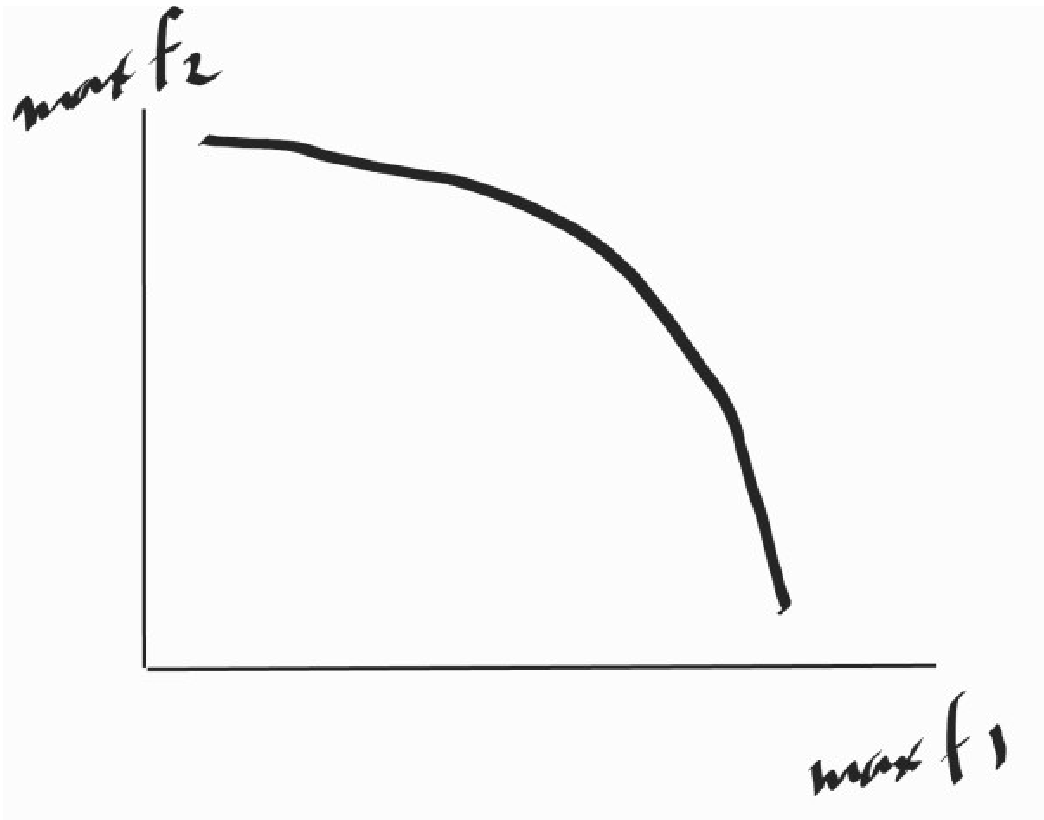
Optimización MultiObjetivo

- Tipos de Frente de Pareto:
  - Cuando minimizamos una función y maximizamos la otra:



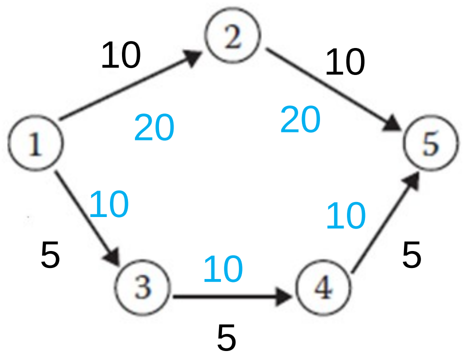
Optimización MultiObjetivo

- Tipos de Frente de Pareto:
  - Cuando maximizamos las dos funciones:



Optimización MultiObjetivo

- Tips:
  - Dos o mas funciones a optimizar no siempre implica tener que aplicar el método de sumas ponderadas.
    - Sucede cuando 2 o mas funciones son perfectamente proporcionales entre si.
      - Ejemplo: Minimizar distancias y minimizar tiempo en una red vehicular.
        - » Nota: cuando el tiempo es proporcional a la distancia.



Dij: distancia

Tij: tiempo

$\textbf{Si } Dij\ \alpha\ Tij \forall i, j$

Entonces:

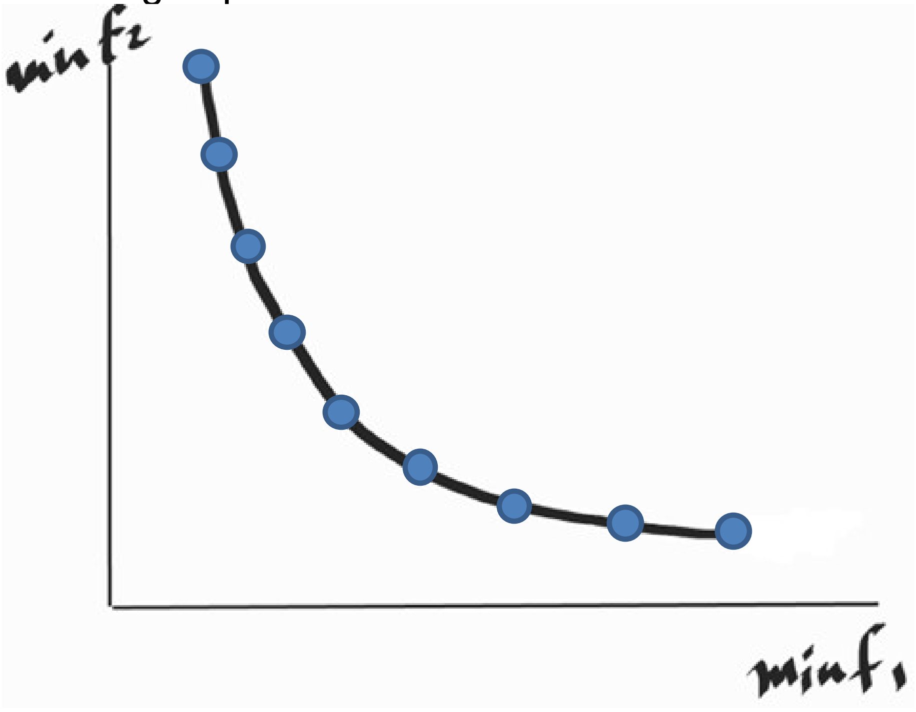
Las n funciones se asumen como una sola función y NO es necesario obtener frente de Pareto.

Sino:

Existiría contraste entre las funciones y sería necesario obtener el frente de Pareto.

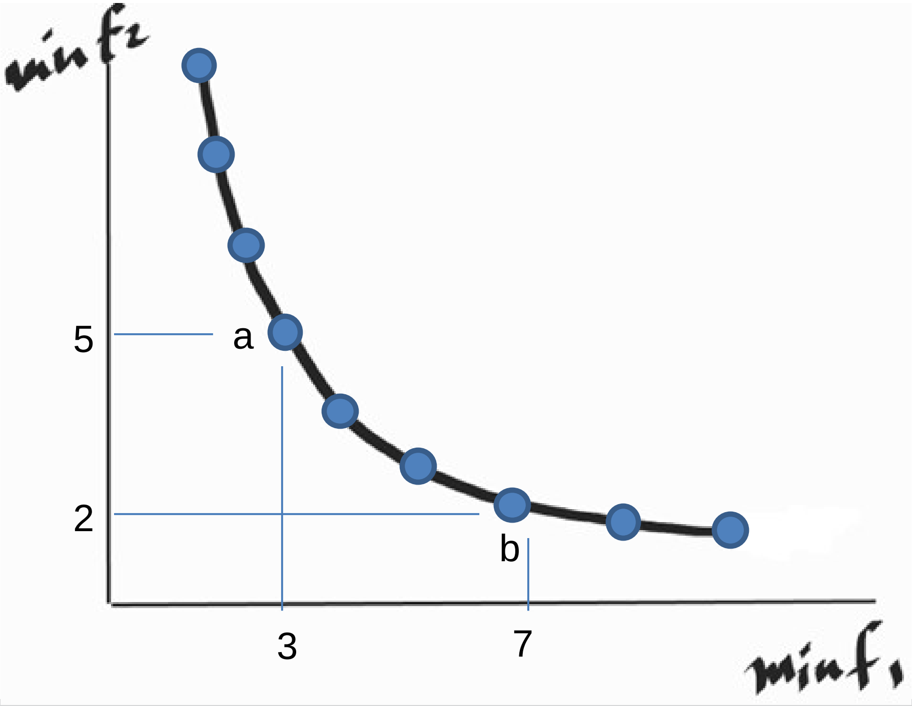
Optimización MultiObjetivo

- Tips:
  - Todos los puntos del frente óptimo de Pareto son los mejores, NO hay uno mejor que otro, TODOS son óptimos de acuerdo a sus pesos.
    - Ningún punto domina a otro.



Optimización MultiObjetivo

- Tips:
  - Todos los puntos del frente óptimo de Pareto son los mejores, no hay uno mejor que otro, todos son óptimos de acuerdo a sus pesos.
    - Ningún punto domina a otro.



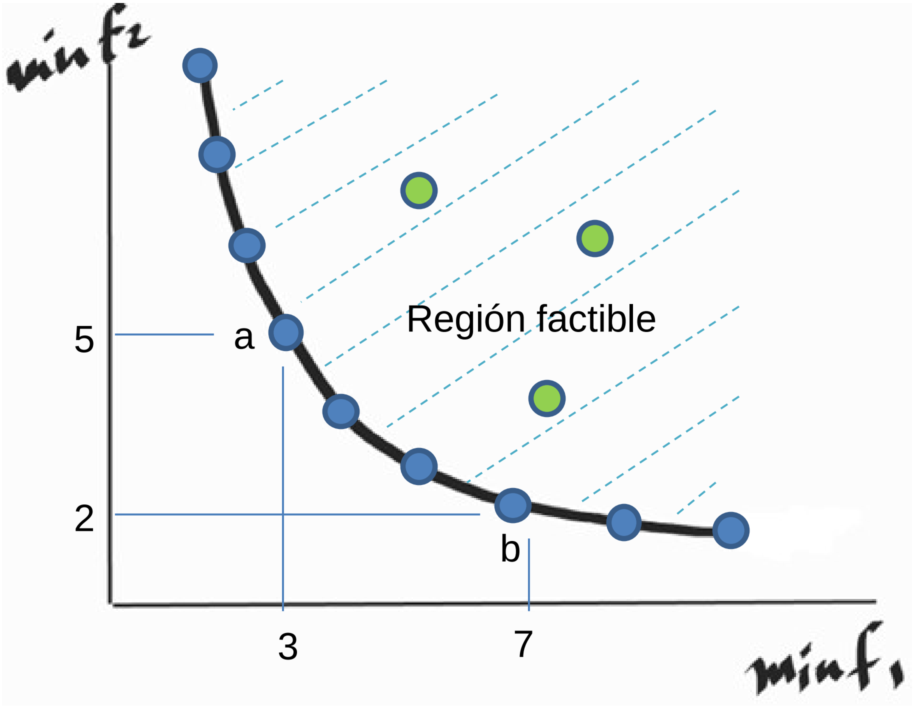
Qué significa que ningún punto Domina a otro?

Rta. Que no existe un punto que supere en las dos funciones objetivo a cualquier otro punto.

Ejemplo: a es mejor que b en la f1 ya que 3 es menor que 7, y b es mejor que a en f2 ya que 2 es menor que 5.

Optimización MultiObjetivo

- Tips:
  - Adicionalmente:
    - Existen puntos que son factibles pero que no hacen parte del frente óptimo de Pareto.

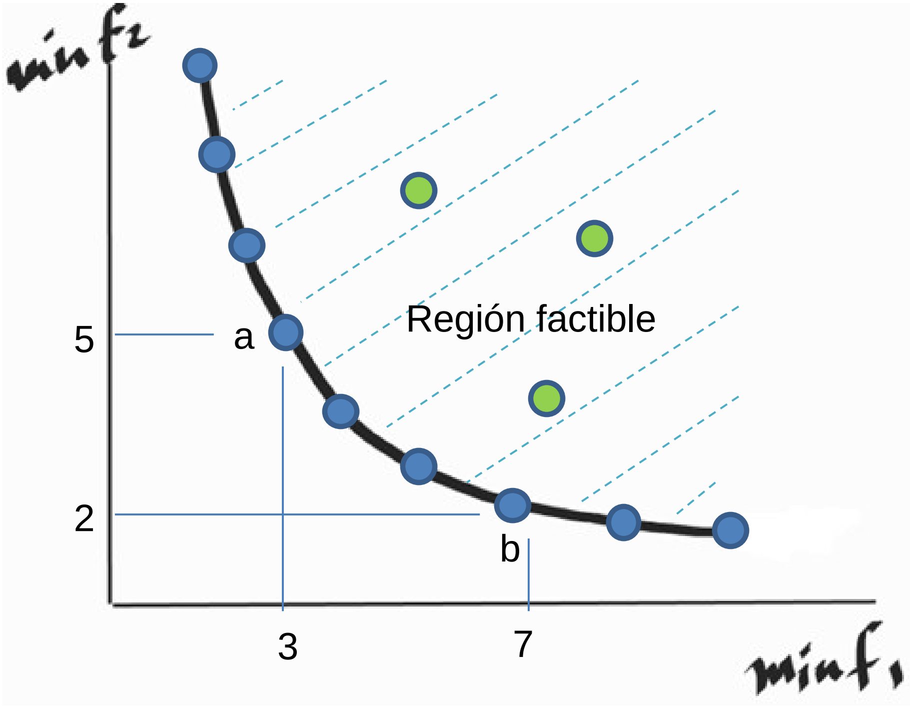


Punto factible pero que NO conformael el frente Optimo de Pareto.

Estos puntos se denominan puntos **Dominados**por los puntos del frente óptimo de Pareto.

Optimización MultiObjetivo

- Tips:
  - Adicionalmente:
    - Existen puntos que son factibles pero que no hacen parte del frente óptimo de Pareto.

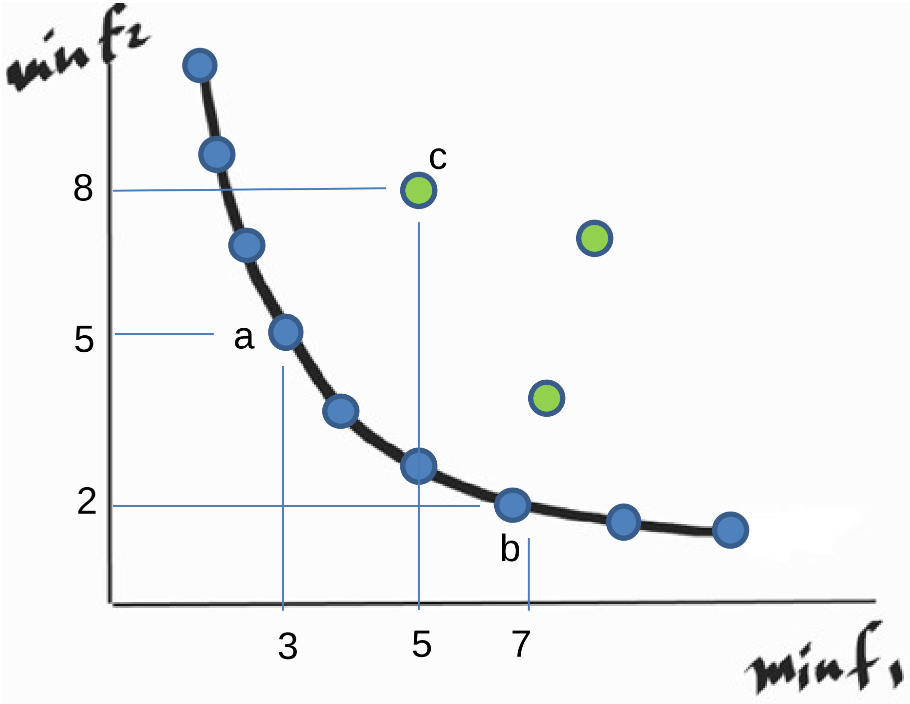


Estos puntos son **Dominados** porque hay al menos un punto del frente óptimo de Pareto que:

-Le gana en las dos funciones o
-Le gana en una función y le Iguala en la otra.

Optimización MultiObjetivo

- Tips:
  - Adicionalmente:
    - Existen puntos que son factibles pero que no hacen parte del frente óptimo de Pareto.



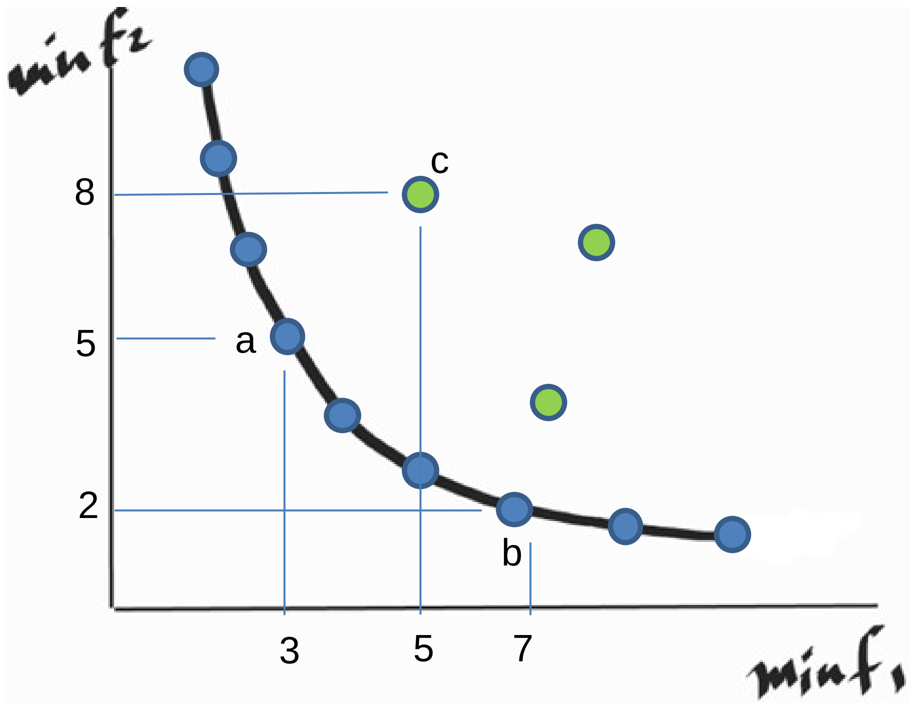
Estos puntos son **Dominados** porque hay al menos un punto del frente óptimo de Pareto que:

-**Le gana en las dos funciones**

Ejemplo: **b es mejor que c** en f2 porque 2 < 8, pero **c es mejor que b** en f1 porque 5 < 7. Entonces son **NO comparables**. Puede haber otro punto que lo domine.

Optimización MultiObjetivo

- Tips:
  - Adicionalmente:
    - Existen puntos que son factibles pero que no hacen parte del frente óptimo de Pareto.



Estos puntos son **Dominados** porque hay al menos un punto del frente óptimo de Pareto que:

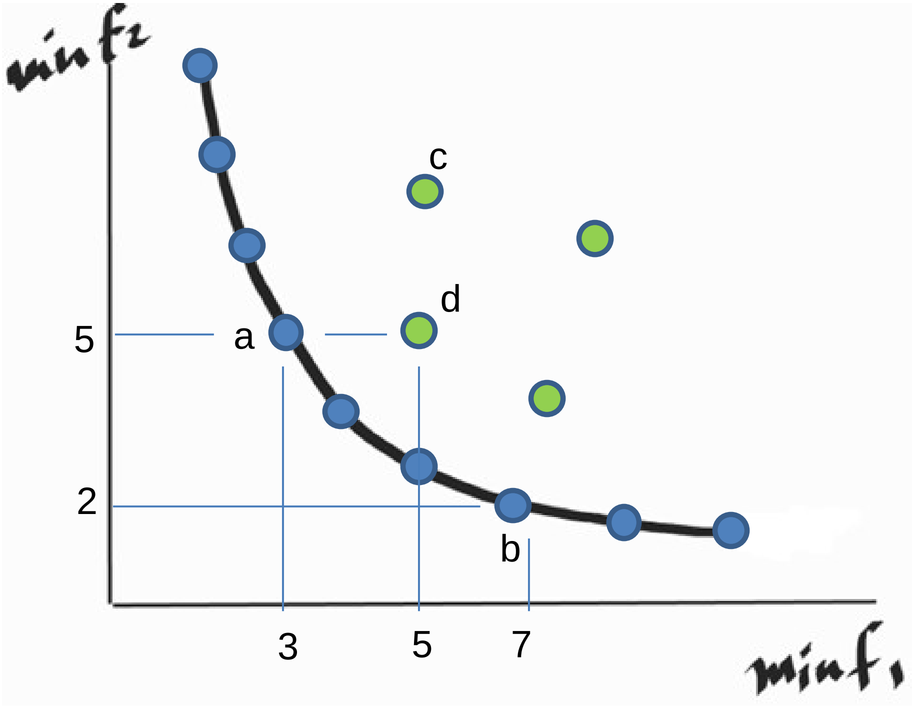
-**Le gana en las dos funciones**

Ejemplo: **a es mejor que c** en f1 porque 3 < 5, y **a es mejor que c** en f2 porque 5 < 8.

a domina c en las dos funciones por lo cual c **no hace parte del frente óptimo de Pareto**.

Optimización MultiObjetivo

- Tips:
  - Adicionalmente:
    - Existen puntos que son factibles pero que no hacen parte del frente óptimo de Pareto.



Estos puntos son **Dominados** porque hay al menos un punto del frente óptimo de Pareto que:

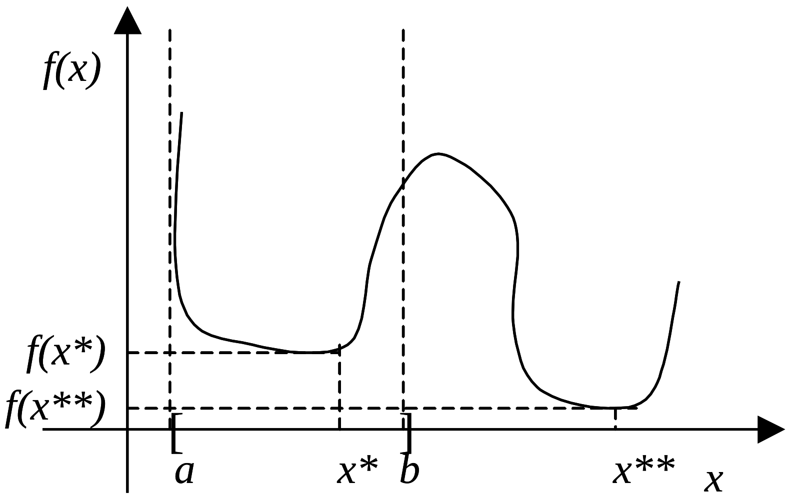
-Le gana en las dos funciones o

-**Le gana en una función y le iguala en la otra.**

Ejemplo: d iguala a en f2, pero A es mejor que d en f1 ya que 3 < 5.
a domina d, por lo cual d **no hace parte del frente óptimo de Pareto**.

Mínimos locales y globales

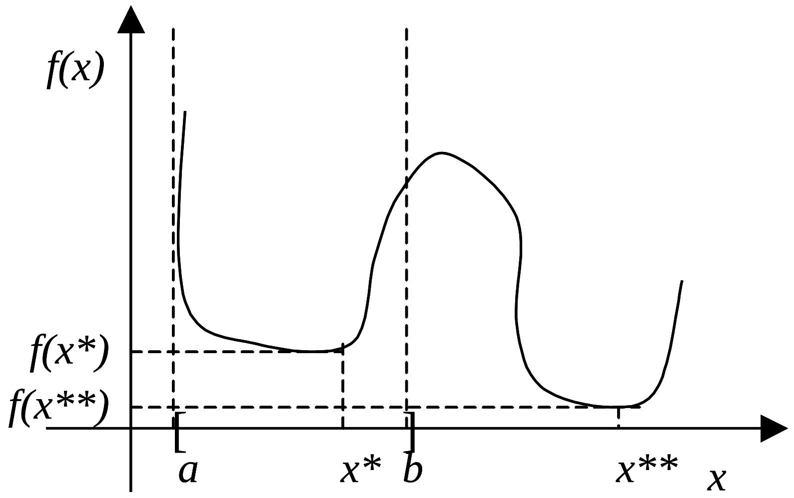
- Cuando al optimizar una función f(x) se quiera encontrar el valor mínimo en un intervalo [a,b]; es decir, $a \le x \le b$, este valor mínimo recibe el nombre de **mínimo local [**f(x*)**]**



- Tradicionalmente las técnicas de búsqueda de mínimos locales son más sencillas que las técnicas de búsqueda de mínimos globales, debido entre varias razones a la complejidad que se genera en el espacio de búsqueda cuando el intervalo es $(-\infty,\infty)$.

Mínimos locales y globales

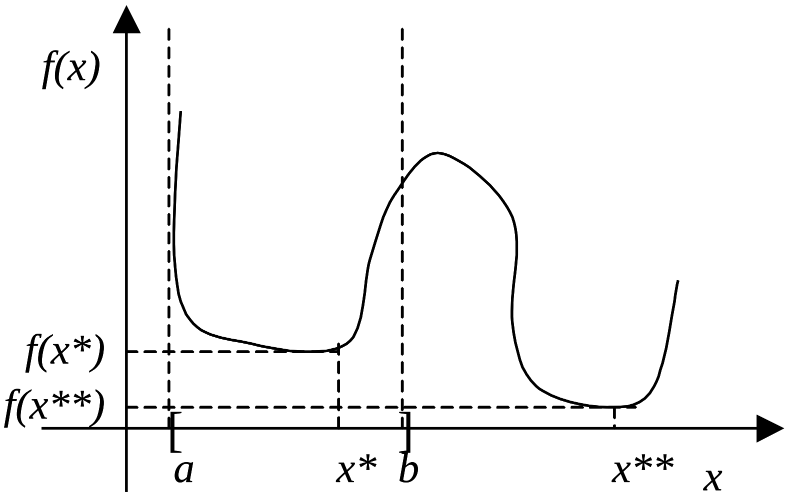
- Cuando la función a minimizar no se encuentra restringida a un intervalo en particular de la función se dice entonces que el valor encontrado es un **mínimo global [**f(x**)**]**. En este caso el intervalo del espacio de búsqueda se encuentra asociado a (-∞,∞).



- A pesar de que en la Figura el valor de la función f(x*) es un mínimo, podemos observar que f(x**) < f(x*).
- Si no existe otro valor f(x'), tal que f(x') < f(x**) en el intervalo (-∞,∞), se dice entonces que f(x**) es un mínimo global de la función f(x)

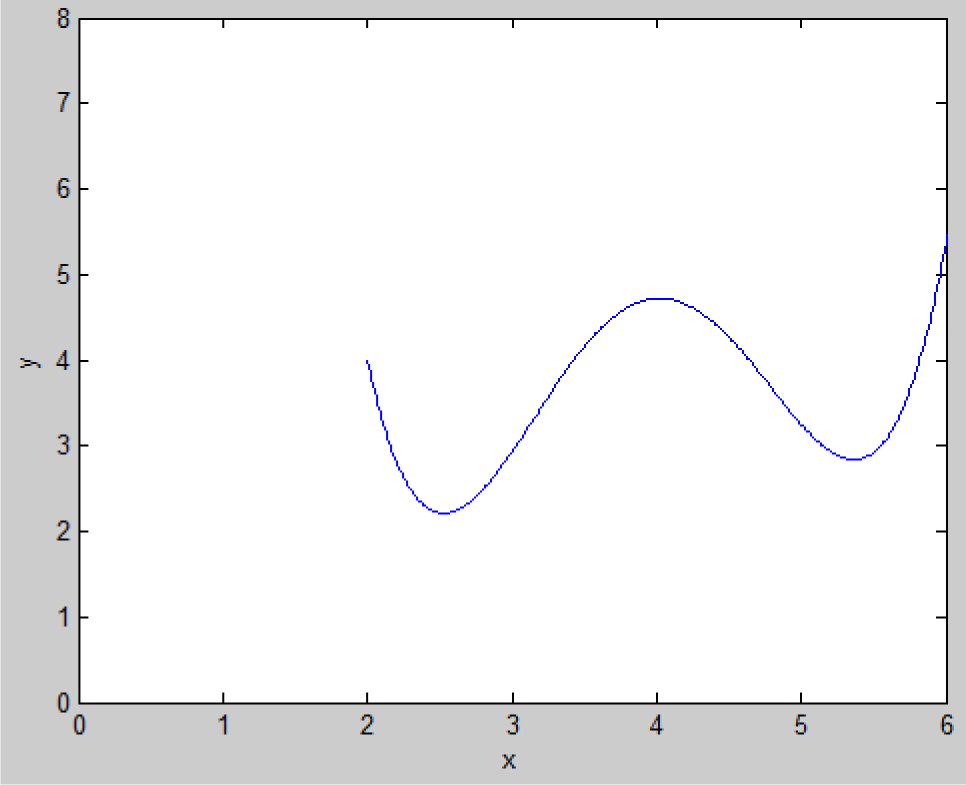
Mínimos locales y globales

- Para la búsqueda del valor **máximo global** de una función el análisis sería exactamente el mismo, pero en este caso no debe existir otro valor f(x’), tal que f(x’) > f(x**) en el intervalo (-∞,∞).



Mínimos locales y globales

- Ejemplo:
  - Ejecutar el archivo « ejMinLocal.gms»



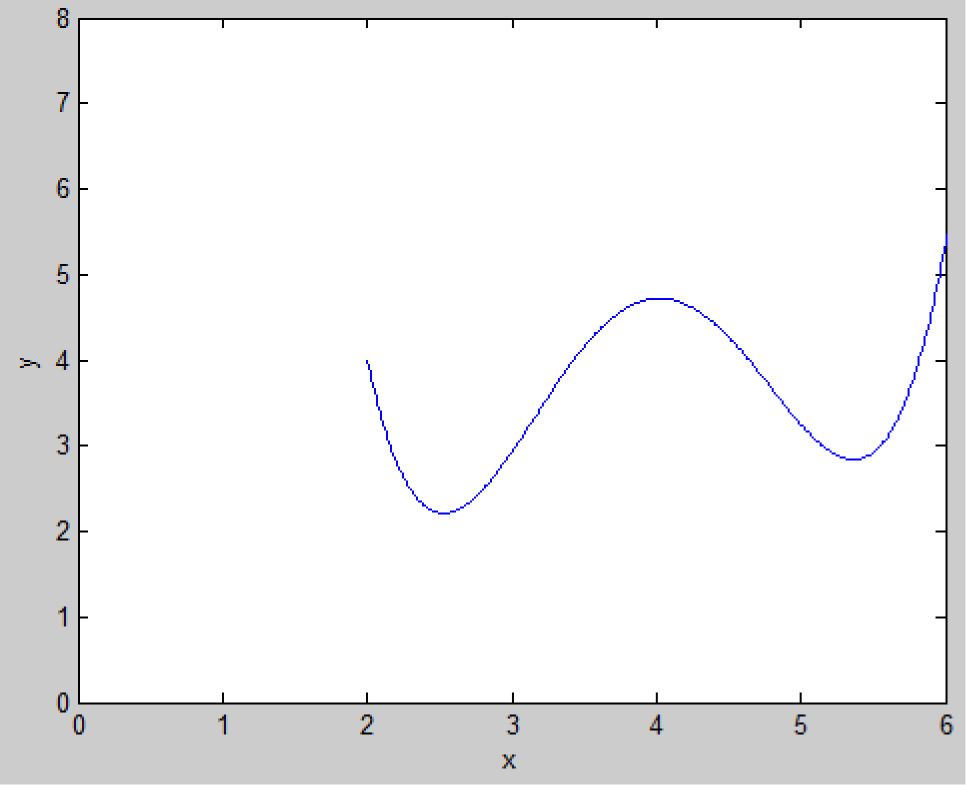
Mínimos locales y globales

- Ejemplo:
  - Ejecutar el archivo «ejMinLocal.gms»
  - Encontramos un mínimo local en el intervalo [4,6].

```
Variables
  x           Indica si el enlace i j es utilizado o no en el SP
  z           minimizacion ;

Equations
Func_Obj            Funcion Objetivo
restriccion_1       restriccion 1
restriccion_2       restriccion 2

;


Func_Obj              ..  z =e= 104 - (117.9167*x) + (48.9583*SQR(x)) - (8.583*POWER(x,3)) + (0.54167*POWER(x,4));
restriccion_1         ..  x =g= 4;
restriccion_2         ..  x =l= 6;

Model Transport /all/ ;
option nlp=COUENNE
Solve transport using nlp minimizing z;

Display x.l
Display z.l
```

Mínimos locales y globales

- Ejemplo:
  - Ejecutar el archivo «ejMinLocal.gms»
  - Encontramos un mínimo local en el intervalo [4,6].




```
VARIABLE x.L                    =        5.256

VARIABLE z.L                    =        3.865
```

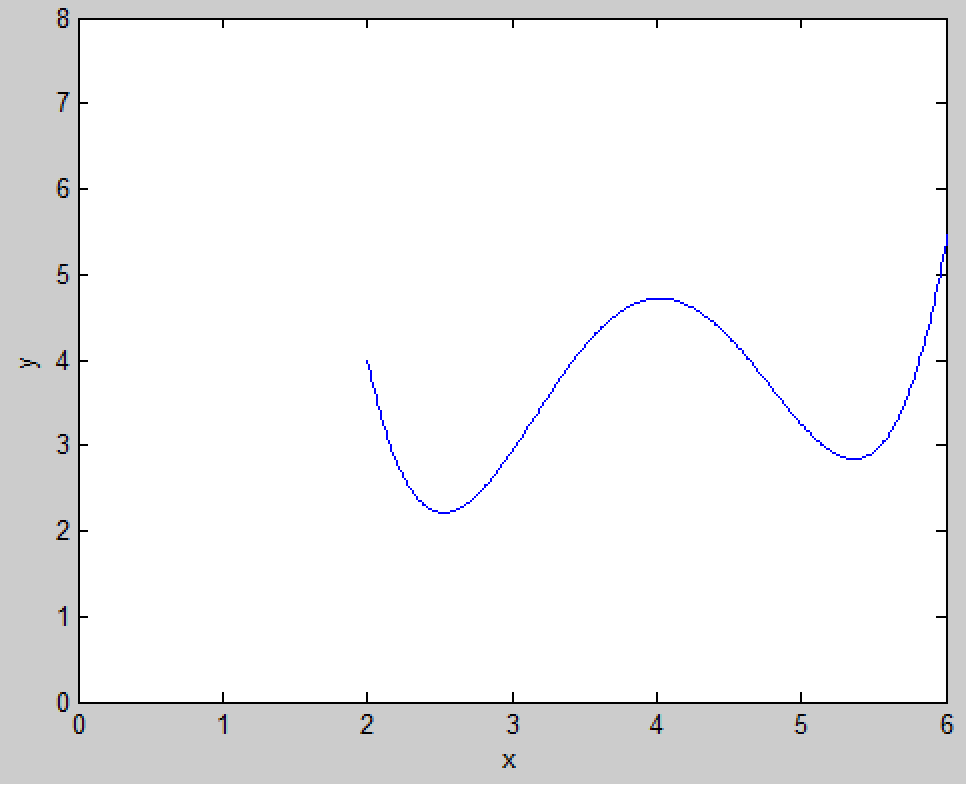
Mínimos locales y globales

- Ejemplo:
  - Ejecutar el archivo «ejMinLocal.gms»
  - Encontramos un mínimo global en el intervalo [-inf,+inf].

```
Variables
  x          Indica si el enlace i j es utilizado o no en el SP
  z          minimizacion ;

Equations
Func_Obj             Funcion Objetivo
*restriccion_1        restriccion 1
*restriccion_2        restriccion 2

;


Func_Obj               ..  z =e= 104 - (117.9167*x) + (48.9583*SQR(x)) - (8.583*POWER(x,3)) + (0.54167*POWER(x,4));
*restriccion_1          ..  x =g= 4;
*restriccion_2          ..  x =l= 6;

Model Transport /all/ ;
option nlp=COUENNE
Solve transport using nlp minimizing z;

Display x.l
Display z.l
```

Mínimos locales y globales

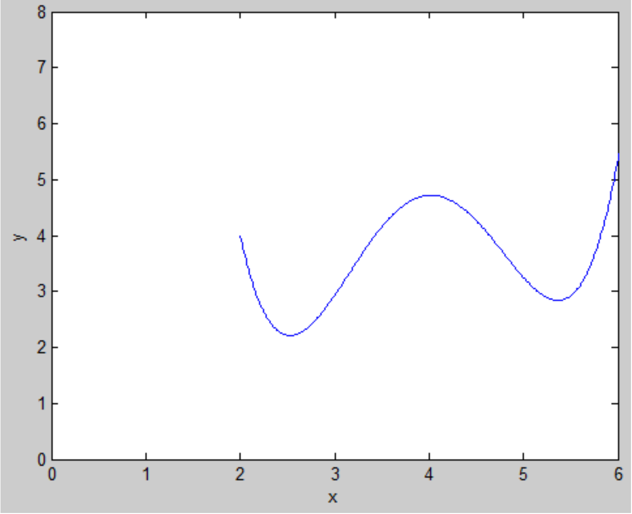
- Ejemplo:
  - Ejecutar el archivo «ejMinLocal.gms»
  - Encontramos un mínimo local en el intervalo [-inf,+inf].




```
VARIABLE x.L                    =        2.511

VARIABLE z.L                    =        2.246
```

Máximo/Mínimo de una función

- Algoritmo de Newton-Raphson:
  - Encuentra un máximo o un mínimo local de una función dependiendo del punto de arranque.
  - El algoritmo por si solo no encuentra un máximo o mínimo global.
    - Para encontrarlo es necesario extraer el máximo/mínimo global de distintos máximos/mínimos locales extraidos previamente.

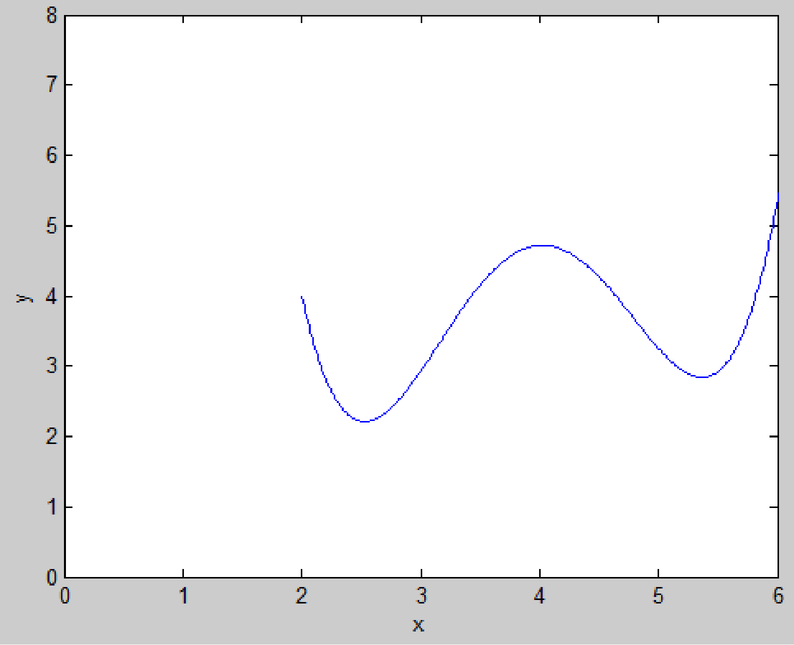


Máximo/Mínimo de una función

- Algoritmo de Newton-Raphson:

Algorithm Pseudocodigo de Newton Raphson para 2 dimensiones

$i \leftarrow 1$
Inicializar x_i
$\alpha \leftarrow 1$
$convergencia \leftarrow 0.001$
while do $| f'(x_i) | > convergencia$
    $x_{i+1} \leftarrow x_i - \alpha \frac{f'(x_i)}{f''(x_i)}$
    $x_i \leftarrow x_{i+1}$
end while
$\hat{x} \leftarrow x_i$
return $\hat{x}$



Máximo/Mínimo de una función

- Algoritmo de Newton-Raphson:
  - Ejemplo: ejNRaphson.m

Algorithm Pseudocodigo de Newton Raphson para 2 dimensiones

$i \leftarrow 1$
Inicializar x_i
$\alpha \leftarrow 1$
convergencia $\leftarrow 0.001$
while do $|\, f'(x_i) \,| > convergencia$
  $x_{i+1} \leftarrow x_i - \alpha \frac{f'(x_i)}{f''(x_i)}$
  $x_i \leftarrow x_{i+1}$
end while
$\hat{x} \leftarrow x_i$
return $\hat{x}$

```
clc, clear all, close all

syms f_x x;

f_x=3*x.^3 - 10*x.^2 - 56*x + 505;

figure
ezplot(f_x)
hold on;

x_i=-6;  %valor inicial

d1_f_x=diff(f_x); %primera derivada
d2_f_x=diff(d1_f_x); %segunda derivada

d1_f_x_i=double(subs(d1_f_x, x, x_i));

convergencia=0.001;  % convergencia

a=0.1;
```

Máximo/Mínimo de una función

- Algoritmo de Newton-Raphson:
  - Ejemplo:

Algorithm Pseudocodigo de Newton Raphson para 2 dimensiones

$i \leftarrow 1$

Inicializar x_i

$\alpha \leftarrow 1$

$convergencia \leftarrow 0.001$

while do $| f'(x_i) | > convergencia$

$\quad x_{i+1} \leftarrow x_i - \alpha \frac{f'(x_i)}{f''(x_i)}$

$\quad x_i \leftarrow x_{i+1}$

end while

$\hat{x} \leftarrow x_i$

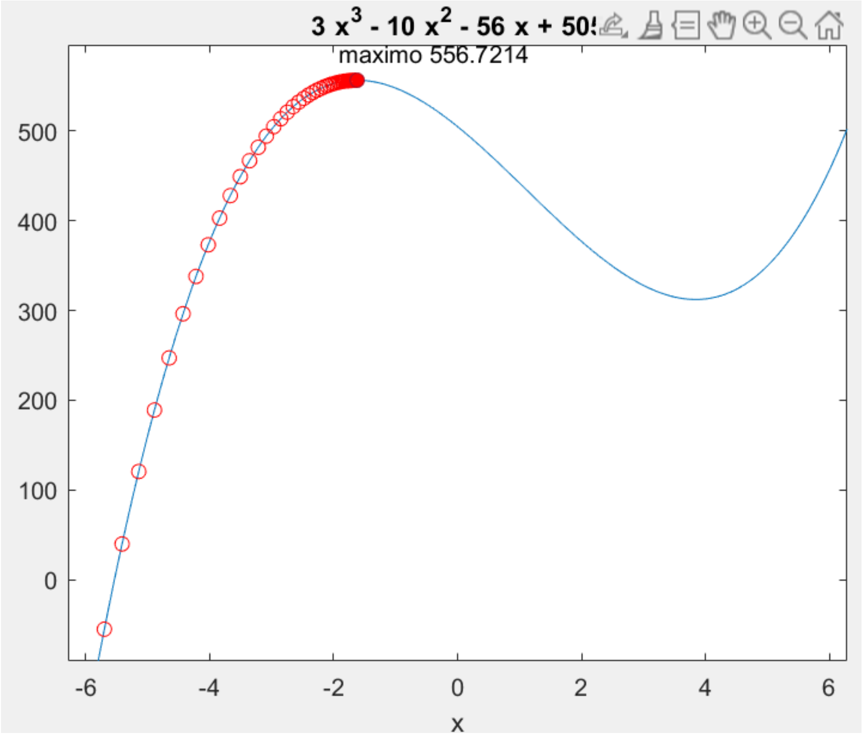
return $\hat{x}$

```
cont=1;
while abs(d1_f_x_i)>convergencia
    cont=cont+1;

    d1_f_x_i=double(subs(d1_f_x,x_i)); %Primera derivada evaluada en x_i

    d2_f_x_i=double(subs(d2_f_x,x_i)); %Segunda derivada evaluada en x_i

    x_i_new=x_i - a*(d1_f_x_i/d2_f_x_i); %Expresión de Newton Raphson: x(i+1) = x(i) - a*f'(x(i))/f''(x(i))

    x_i=x_i_new;  %Actualizamos el x_i

    f_x_i=double(subs(f_x,x_i)); %Evaluamos el x_i en la f(x) para graficar posteriormente

    plot(x_i, f_x_i, 'or')

end
```

Universidad de los Andes

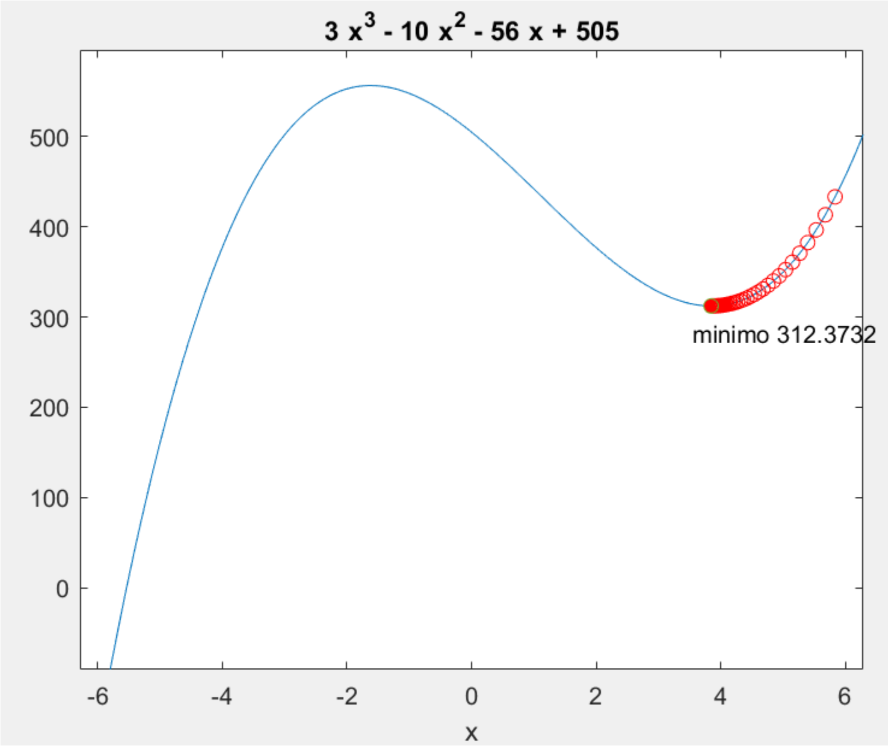
Máximo/Mínimo de una función

- Algoritmo de Newton-Raphson:
  - Ejemplo:
    - Alfa=0.1, Punto de arranque=-6



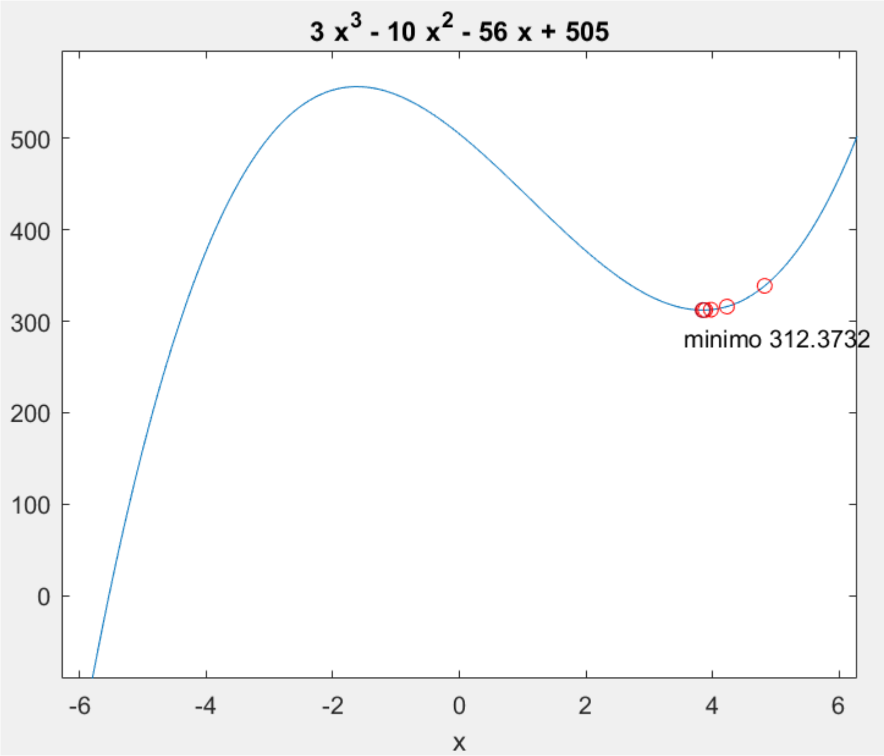
Máximo/Mínimo de una función

- Algoritmo de Newton-Raphson:
  - Ejemplo:
    - Alfa=0.1, Punto de arranque=6



Máximo/Mínimo de una función

- Algoritmo de Newton-Raphson:
  - Ejemplo:
    - Alfa=0.7, Punto de arranque=6



Máximo/Mínimo de una función

- Actividad en casa:
  - Descargar y probar que funciona el ejemplo "ejNRaphson.m"

Máximo/Mínimo de una función

- Deducción Algoritmo de Newton-Raphson:
  - Teorema de las Series de Taylor:

$$f(x+h) = f(x) + f'(x)h + \frac{1}{2}f''(x)h^2 + \cdots + \frac{1}{k!}f^{[k]}(x)h^k + \frac{1}{(k+1)!}f^{[k+1]}(w)h^{k+1}$$

  - Ejemplos:

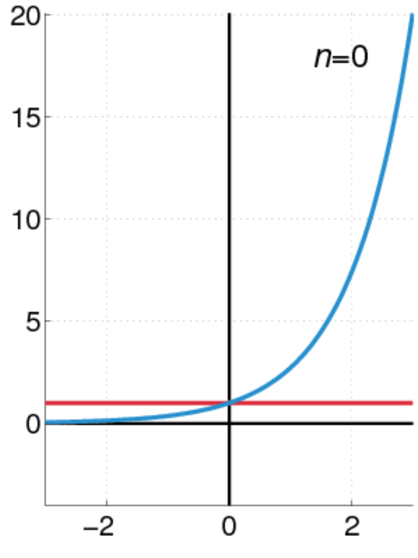
$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} \quad , \forall x; n \in \mathbb{N}_0$$

$$\ln(1+x) = \sum^{\infty} \frac{(-1)^{n+1}}{n} x^n \quad , \text{para } |x| < 1$$

$$\sin x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1} \quad , \forall x$$

$$\cos x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n} \quad , \forall x$$

$$\tan x = \sum_{n=1}^{\infty} \frac{B_{2n}(-4)^n(1-4^n)}{(2n)!} x^{2n-1} \quad , \text{para } |x| < \frac{\pi}{2}$$



Máximo/Mínimo de una función

- Deducción Algoritmo de Newton-Raphson:
  - Teorema de las Series de Taylor:

$$f(x+h) = f(x) + f'(x)h + \frac{1}{2}f''(x)h^2 + \cdots + \frac{1}{k!}f^{[k]}(x)h^k + \frac{1}{(k+1)!}f^{[k+1]}(w)h^{k+1}$$

  - A medida que *h* tiende a 0, los términos de orden alto tienden a 0 mucho mas rápido que los términos de menor orden.
    - Para valores pequeños de h (aproximación de primer orden de Taylor):

$$f(x+h) \approx f(x) + f'(x)h$$

    - Aproximación de segundo orden de Taylor:

$$f(x+h) \approx f(x) + f'(x)h + \frac{1}{2}f''(x)h^2$$

Máximo/Mínimo de una función

- Deducción Algoritmo de Newton-Raphson:
  - Aproximación de Primer orden:

$$f(x+h) \approx f(x) + f'(x)h \quad \Rightarrow \quad f(x+h) \approx a + bh$$

$$a = f(x)$$

$$b = f'(x)$$

  - Aproximación de Segundo orden:

$$f(x+h) \approx f(x) + f'(x)h + \frac{1}{2}f''(x)h^2 \quad \Rightarrow \quad f(x+h) \approx a + bh + \frac{1}{2}ch^2$$

$$a = f(x)$$

$$b = f'(x)$$

$$c = f''(x)$$

Máximo/Mínimo de una función

- Deducción Algoritmo de Newton-Raphson:
  - Segundo orden:

$$f(x+h) \approx f(x) + f'(x)h + \frac{1}{2}f''(x)h^2 \quad \Rightarrow \quad f(x+h) \approx a + bh + \frac{1}{2}ch^2$$

$$a = f(x)$$

$$b = f'(x)$$

$$c = f''(x)$$

  - Primera derivada:

$$f'(x+h) \approx b + ch$$

  - Condición para encontrar un máximo/mínimo:

$$0 = b + c\hat{h} \quad \Rightarrow \quad \hat{h} = -\frac{b}{c}$$

Máximo/Mínimo de una función

- Deducción Algoritmo de Newton-Raphson:
  - Condición para encontrar un máximo/mínimo:

$$f'(x+h) \approx b + ch$$

$$0 = b + c\hat{h} \quad \Rightarrow \quad \hat{h} = -\frac{b}{c}$$

  - Sumamos a cada lado *x*:

$$x + \hat{h} = x - \frac{b}{c}$$

  - Reemplazamos *b* y *c*:

$$x + \hat{h} = x - \frac{1}{f''(x)} f'(x)$$

Máximo/Mínimo de una función

- Expresión general:

$$\boxed{x + \hat{h} = x - \frac{1}{f''(x)} f'(x)}$$

- Algoritmo de Newton Raphson:

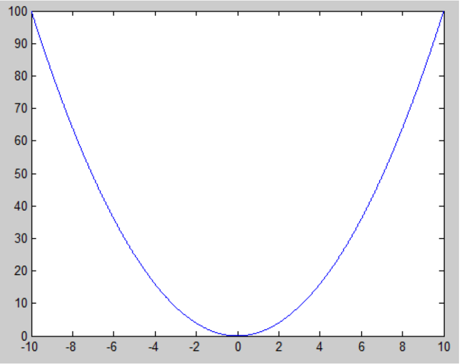
Algorithm Pseudocodigo de Newton Raphson para 2 dimensiones

$i \leftarrow 1$
Inicializar x_i
$\alpha \leftarrow 1$
$convergencia \leftarrow 0.001$
while do $\mid f'(x_i) \mid > convergencia$
    $x_{i+1} \leftarrow x_i - \alpha \frac{f'(x_i)}{f''(x_i)}$
    $x_i \leftarrow x_{i+1}$
end while
$\hat{x} \leftarrow x_i$
return $\hat{x}$

Herramientas para aplicar NR en MATLAB

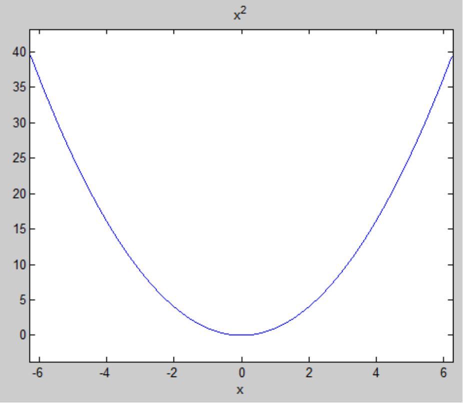
- Definición teórica de una función
  - Método habitual:

```
clc, clear all, close all

x=-10:0.01:10;

y=x.^2;

figure
plot(x,y)
```



  - Método teórico:

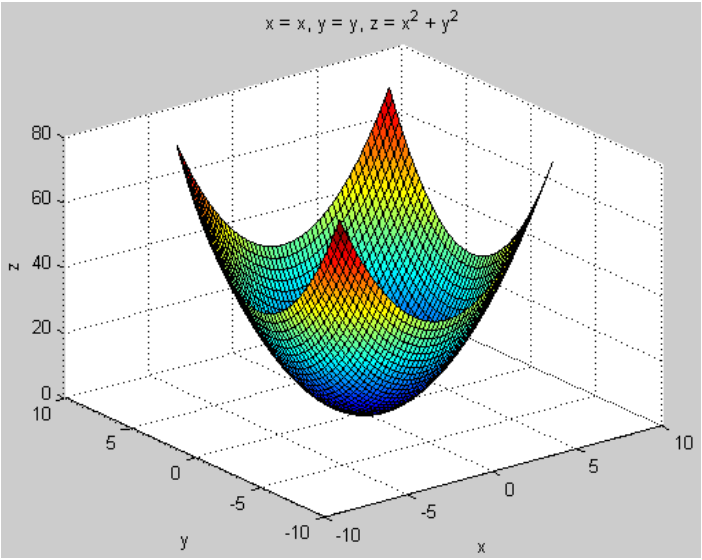
```
syms x y

y = x^2;

figure
ezplot(y)
```



Herramientas para aplicar NR en MATLAB

- Definición teórica de una función
  - Función en 3D o superficie:

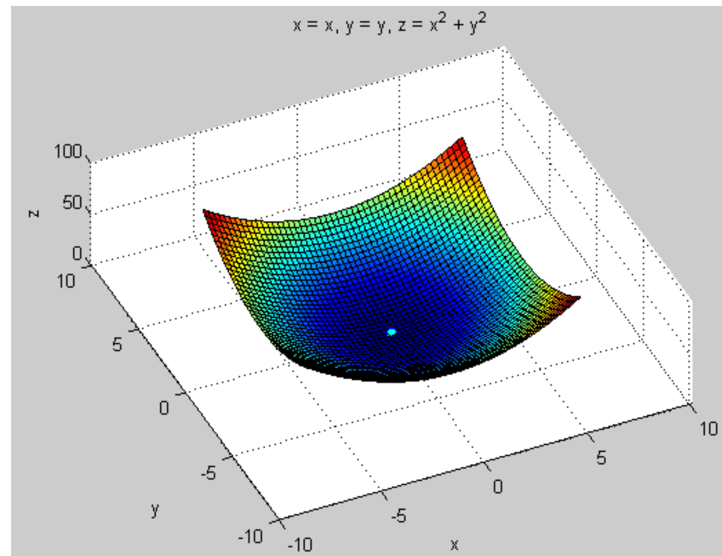
```
syms z x y

z = x^2 + y^2;

figure
ezsurf(x,y,z)
```



Herramientas para aplicar NR en MATLAB

- Definición teórica de una función
  - Función en 3D o superficie:

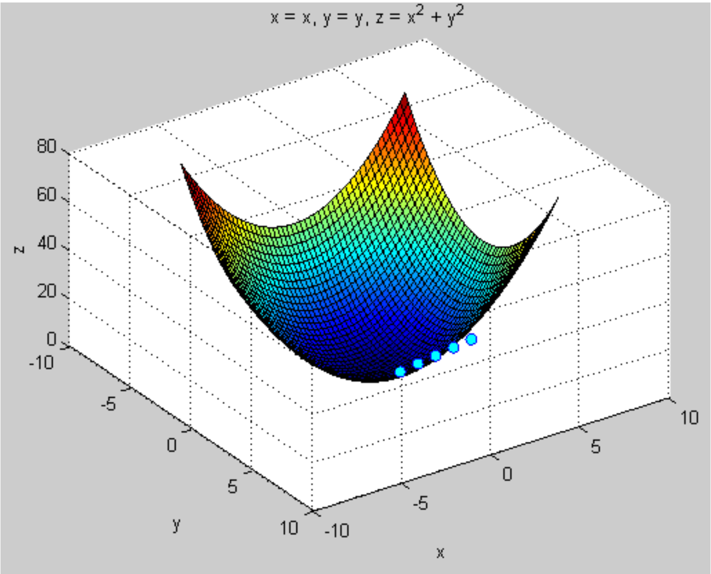
```
syms z x y

z = x^2 + y^2;

figure
ezsurf(x,y,z)
hold on;

plot3(0,0,0,'o', 'MarkerFaceColor', 'c')
```



Herramientas para aplicar NR en MATLAB

- Definición teórica de una función
  - Función en 3D o superficie:

```
syms z x y

z = x^2 + y^2;

figure
ezsurf(x,y,z)
hold on;

zz=[1 2 3 4 5];
yy=[1 1 1 1 1 ];
xx=[1 2 3 4 5];

plot3(xx,yy,zz,'o', 'MarkerFaceColor', 'c')
```



Evaluación de funciones

- Evaluación de funciones
  - Evaluar una función 2D:

```
syms x y

y = x^2;

y_eval=subs(y,[x],[0]);
y_eval_double=double(y_eval)
```

```
Command Window
New to MATLAB? Watch th

y_eval_double =

     0

fx >>
```

  - Evaluar una función 3D:

```
syms z x y

z = x^2 + y^2;

z_eval=subs(z,[x y],[0 0]);
z_eval_double=double(z_eval)
```

```
Command Window
New to MATLAB? Watch thi

z_eval_double =

     0

fx >>
```

Herramientas para aplicar NR en MATLAB

- Derivada de una función

```
syms x y

y = x^2;

d1_y=diff(y)

figure
ezplot(x,y)
```

```
Command Window
New to MATLAB? Watch th

d1_y =

2*x

>>
```

- Doble derivada de una función:

```
syms x y

y = x^2;

d1_y=diff(y);
d2_y=diff(d1_y)
```

```
Command Window
New to MATLAB? Watch thi

d2_y =

2

>>
```

Herramientas para aplicar NR en MATLAB

- Gradiente y Hessiano de una función
  - Gradiente de una función:

$$\nabla f(\mathbf{r}) = \left(\frac{\partial f(\mathbf{r})}{\partial x_1}, \ldots, \frac{\partial f(\mathbf{r})}{\partial x_n}\right)$$

```
syms z x y

z = x^2 + y^2;

grad_z=gradient(z)
```

```
Command Window
New to MATLAB? Watch thi

grad_z =

 2*x
 2*y

>>
```

  - Hessiano de una función:

$$H(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

```
syms z x y

z = x^2*y^2;

hess_z=hessian(z)
```

```
Command Window
New to MATLAB? Watch this V

hess_z =

[ 2*y^2, 4*x*y]
[ 4*x*y, 2*x^2]

>>
```

Herramientas para aplicar NR en MATLAB

- Gradiente y Hessiano de una función
  - Evaluar el Gradiente de una función:

```
syms z x y

z = x^2 + y^2;

grad_z=gradient(z)

grad_z_eval=subs(grad_z, [x y], [1 2]);
grad_z_eval_double=double(grad_z_eval)
```

```
Command Window
New to MATLAB? Watch

grad_z =

2*x
2*y


grad_z_eval_double =

  2
  4

fx >>
```

  - Evaluar el Hessiano de una función:

```
syms z x y

z = x^2*y^2;

hess_z=hessian(z)

hess_z_eval=subs(hess_z, [x y], [1 2]);
hess_z_eval_double=double(hess_z_eval)
```

```
Command Window  Vari
New to MATLAB? Watch this V

hess_z =

[ 2*y^2, 4*x*y]
[ 4*x*y, 2*x^2]


hess_z_eval_double =

  8   8
  8   2

fx >>
```

Herramientas para aplicar NR en MATLAB

- Inversa y Norma de una función
  - Inversa de una función: inv(Matriz_A)
  - Norma de un vector:
    - Norm(vector_A) $\|\mathbf{v}\| = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2}$

Algoritmo de NR para encontrar máximos/mínimos de una función

- Caso 2D:

Algorithm Pseudocodigo de Newton Raphson para 2 dimensiones

$i \leftarrow 1$
$Inicializar\ x_i$
$\alpha \leftarrow 1$
$convergencia \leftarrow 0.001$
while do $\mid f'(x_i) \mid > convergencia$
  $x_{i+1} \leftarrow x_i - \alpha \frac{f'(x_i)}{f''(x_i)}$
  $x_i \leftarrow x_{i+1}$
end while
$\hat{x} \leftarrow x_i$
return $\hat{x}$

- Caso 3D:

Algorithm Pseudocodigo de Newton Raphson para **3** dimensiones

$i \leftarrow 1$
$Inicializar\ x_i$
$\alpha \leftarrow 1$
$convergencia \leftarrow 0.001$
while do $\| \nabla f(x_i) \| > convergencia$
  $x_{i+1} \leftarrow x_i - \alpha (H(f(x_i))^{-1} \nabla f(x_i)$
  $x_i \leftarrow x_{i+1}$
end while
$\hat{x} \leftarrow x_i$
return $\hat{x}$

Algoritmo de NR para encontrar máximos/mínimos de una función

- Actividades en casa:
  - Probar los casos particulares de las herramientas para implementar Newton Raphson en 2D y 3D.

Algoritmo del Gradiente Descendente para encontrar mínimos locales de una función

- Pseudocodigo del GD:

Algorithm Pseudocodigo del algoritmo del Gradiente Descendente

---

1: $Inicializar\ x_0$
2: $Inicializar\ \alpha$
3: **for** 1 to N **do**
4: $\quad x_0 = x_0 - \alpha * \nabla f(x_0)$
5: **end for**
6: $x^* \longleftarrow x_0$

  - Caracteristicas del algoritmo:
    - Se usa para encontrar mínimos locales.
    - X_0 : valor de arranque.
    - Alfa: define el tamaño del paso.
    - N: número de pasos para encontrar un mínimo local.
    - Caso 2D: primera derivada.
    - Caso 3D: gradiente.

Universidad de los Andes

Algoritmo del Gradiente Descendente para encontrar mínimos locales de una función

- Pseudocodigo del GD:

Algorithm Pseudocodigo del algoritmo del Gradiente Descendente

1: $Inicializar\ x_0$
2: $Inicializar\ \alpha$
3: **for** 1 to N **do**
4: $\quad x_0 = x_0 - \alpha * \nabla f(x_0)$
5: **end for**
6: $x^* \longleftarrow x_0$

  – Caracteristicas del algoritmo:
    - También tenemos un alfa como en NR.
      – Alfa grande: alfa=0.9, alfa=1, etc
      – Alfa pequeño: alfa=0.01, alfa=0.02, etc
    - Si alfa es grande, son necesarios menos pasos para encontrar el mínimo o máximo, pero el riesgo de no convergencia es mayor.
    - Si alfa es pequeño, son necesarios mas pasos para encontrar el mínimo o máximo, pero el riesgo de no convergencia es menor.

Algoritmo del Gradiente Ascendente para encontrar máximos locales de una función

- Pseudocodigo del GA:

Algorithm Pseudocodigo del algoritmo del Gradiente Ascendente

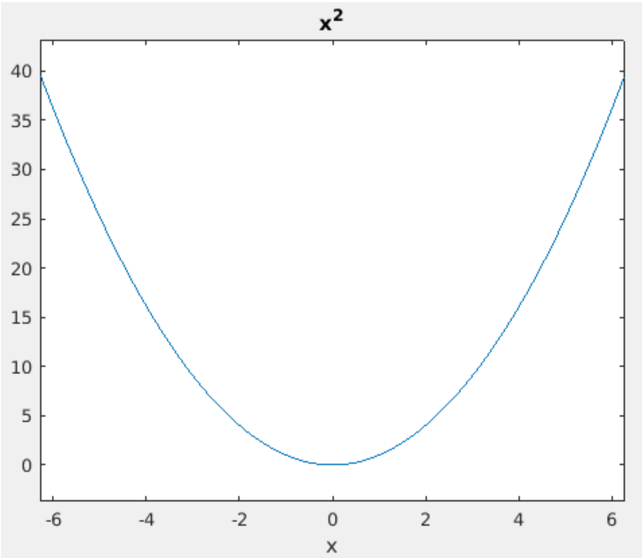
1: $Inicializar\ x_0$
2: $Inicializar\ \alpha$
3: **for** 1 *to* N **do**
4: $\quad x_0 = x_0 + \alpha * \nabla f(x_0)$
5: **end for**
6: $x^* \longleftarrow x_0$

  - Caracteristicas del algoritmo:
    - Se usa para encontrar máximos locales.
    - X_0 : valor de arranque.
    - Alfa: define el tamaño del paso.
    - N: número de pasos para encontrar un máximo local.
    - Caso 2D: primera derivada.
    - Caso 3D: gradiente.

Algoritmo del Gradiente Descendente para encontrar mínimos de una función

- Implementación en MATLAB del algoritmo de GD:
  - Encontrar el mínimo de $f(x)=x^2$

```
clc, clear all, close all

syms y x;

y=x^2;

figure

ezplot(y)
hold on;
```



Algoritmo del Gradiente Descendente para encontrar mínimos de una función

- Implementación en MATLAB del algoritmo de GD:
  - Encontrar el mínimo de $f(x)=x^2$

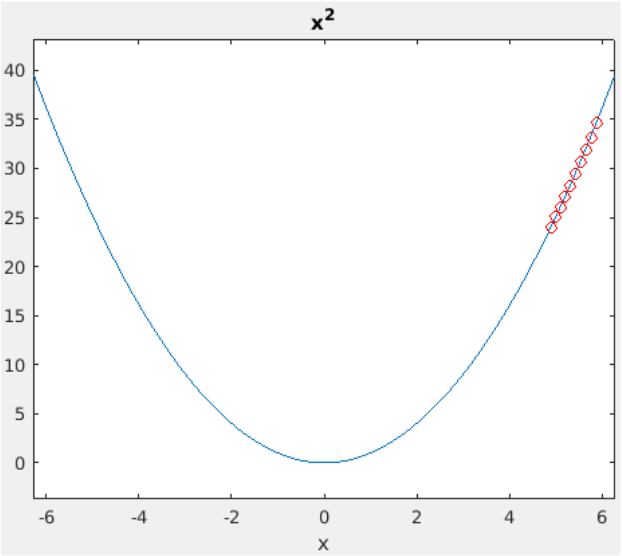
```
x0=6;  %valor inicial

alfa=0.01;
N=10;

d_y=diff(y);
x0Buffer=[];

for i=1:N

    d_y=diff(y);
    d_y_eval=double(subs(d_y,x,x0));

    x0=x0-alfa*d_y_eval;

    x0Buffer=[x0Buffer x0];

    y_eval=double(subs(y,x,x0));
    plot(x0, y_eval, 'or')

end
```

Inicializar x_0
Inicializar α
for 1 *to* N **do**
  $x_0 = x_0 - \alpha * \nabla f(x_0)$
end for

Algoritmo del Gradiente Descendente para encontrar mínimos de una función

- Implementación en MATLAB del algoritmo de GD:
  - Encontrar el mínimo de $f(x)=x^2$

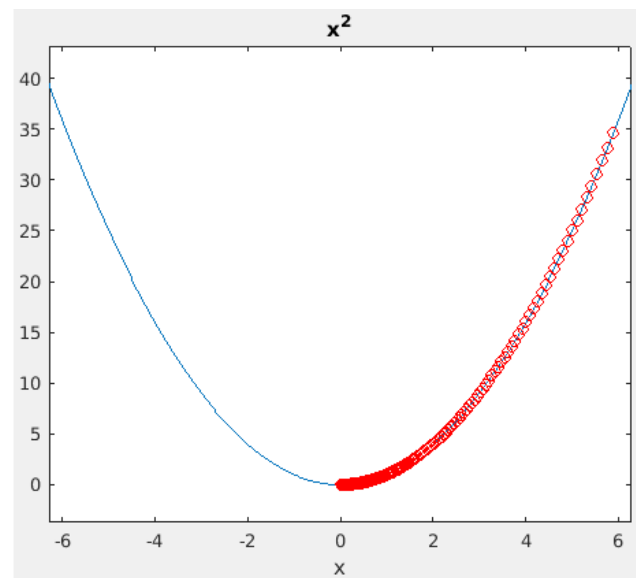
```
x0=6;  %valor inicial

alfa=0.01;
N=10;

d_y=diff(y);
x0Buffer=[];

for i=1:N

    d_y=diff(y);
    d_y_eval=double(subs(d_y,x,x0));

    x0=x0-alfa*d_y_eval;

    x0Buffer=[x0Buffer x0];

    y_eval=double(subs(y,x,x0));
    plot(x0, y_eval, 'or')

end
```



Algoritmo del Gradiente Descendente para encontrar mínimos de una función

- Implementación en MATLAB del algoritmo de GD:
  - Encontrar el mínimo de $f(x)=x^2$
    - Actividad en casa: Ajustar los valores de alfa y/o N para encontrar el mínimo de la función (archivo ejGradDescendente.m).

```
x0=6;  %valor inicial

alfa=0.01;
N=10;

d_y=diff(y);
x0Buffer=[];

for i=1:N

    d_y=diff(y);
    d_y_eval=double(subs(d_y,x,x0));

    x0=x0-alfa*d_y_eval;

    x0Buffer=[x0Buffer x0];

    y_eval=double(subs(y,x,x0));
    plot(x0, y_eval, 'or')

end
```



Algoritmo del Gradiente Descendente para encontrar mínimos de una función

- Pseudocodigo de GD Modificado:

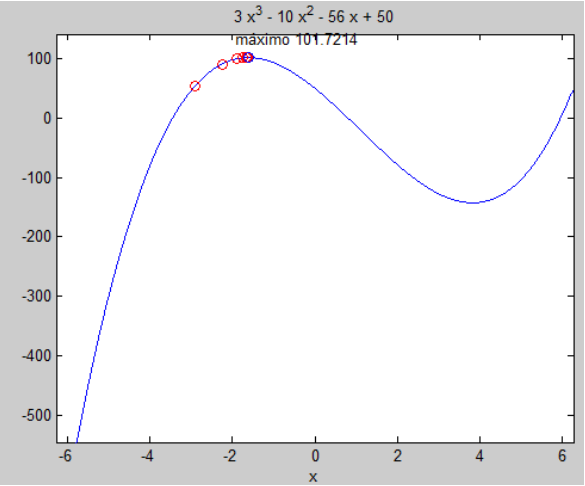
Algorithm Pseudocodigo del algoritmo del GD Modificado

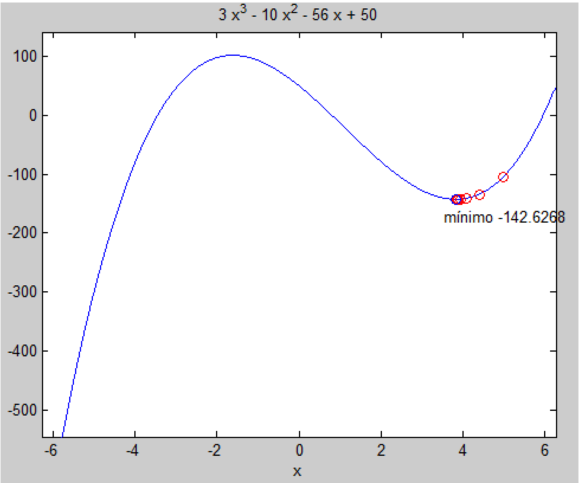
1: $Inicializar\ x_0$
2: $Inicializar\ \alpha$
3: $Inicializar\ convergencia$
4: **while** $\|\nabla f(x_0)\| > convergencia$ **do**
5: $\quad x_0 = x_0 - \alpha * \nabla f(x_0)$
6: **end while**
7: $x^* \longleftarrow x_0$

  – Caracteristicas del algoritmo:
    - X_0 : valor de arranque.
    - Alfa: define el tamaño del paso.
    - ~~N: número de pasos para encontrar un mínimo local.~~
    - Caso 2D: primera derivada.
    - Caso 3D: gradiente.
  – Para el algoritmo del GA modificado, simplemente cambiamos a positivo el signo del término: $\alpha * \nabla f(x_0)$

Algoritmo del Gradiente Descendente para encontrar mínimos de una función

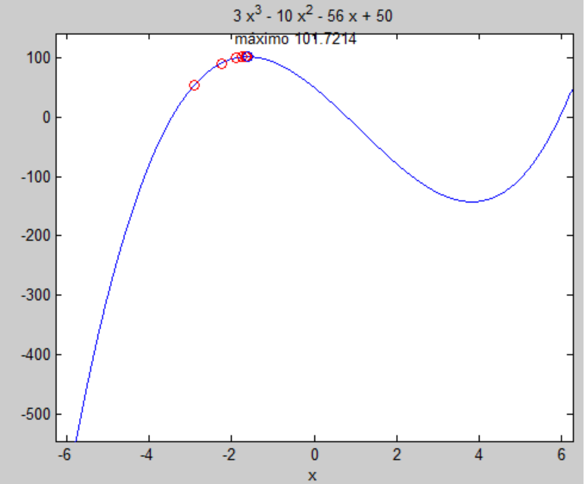
- Implementación en MATLAB del algoritmo del GD y GA:
  - Actividad en casa: Encontrar el mínimo y el máximo de la siguiente función $f(x)=3x^3 -10x^2 -56x + 50$

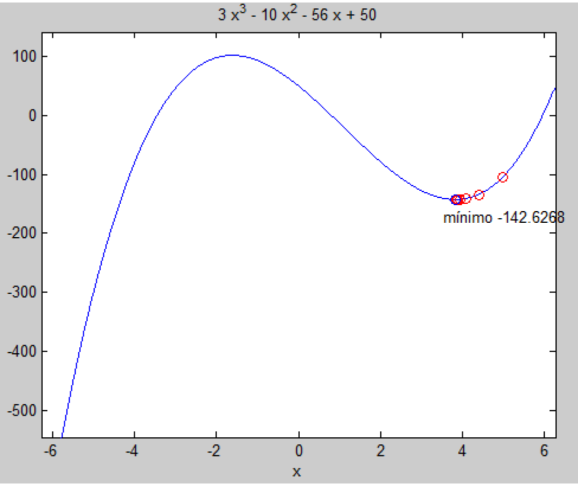




Actividad en casa

- Resultado esperado:
  - Matlab:





  - GAMS:

```
41 VARIABLE x.L                    =        -1.620


42 VARIABLE z.L                    =       101.721
```

```
41 VARIABLE x.L                    =         3.842


42 VARIABLE z.L                    =      -142.627
```