Introducción a Software de Optimización

Rodrigo Maranzana

Componentes

INTERFAZ / WRAPPER

- Python – scipy.optimize.linprog / PuLP
- MATLAB – LinProg / IntLinProg
- Julia – Jump.jl
- GAMS
- IBM ILOG
- Gusek
- Excel
- .....

SOLVER

- Gurobi
- CPLEX
- GLPK
- Xpress
- COIN-OR Solvers(Cbc, Clp, Ipopt, etc.)
- Mosek
- .....

Solver de MS Excel

Ejemplo:

$$max\ z = 5x + 3y$$
$$st:$$
$$3x + 2y \leq 2400$$
$$y \leq 800$$
$$2x \leq 1200$$

1) Ingresamos los datos del problema y armamos el diseño de la planilla:

	A	B	C	D	E	F
1		x	0	y	0	
2	Coef. Func.	c_x	5	c_y	3	
3						
4	Coef. Tec.	a_1x	3	a_1y	2	
5		a_2x	0	a_2y	1	
6		a_3x	2	a_3y	0	

2) Armamos las funciones de excel para las restricciones:

MULTIPLO.S... | = C4*C1+E4*E1

	A	B	C	D	E	F	G	H	I	J
1		x	0	y	0					
2	Coef. Func.	c_x	5	c_y	3		Objetivo	0		
3										bi
4	Coef. Tec.	a_1x	3	a_1y	2		Restr. 1	1	<=	2400
5		a_2x	0	a_2y	1		Restr. 2	0	<=	800
6		a_3x	2	a_3y	0		Restr. 3	0	<=	1200

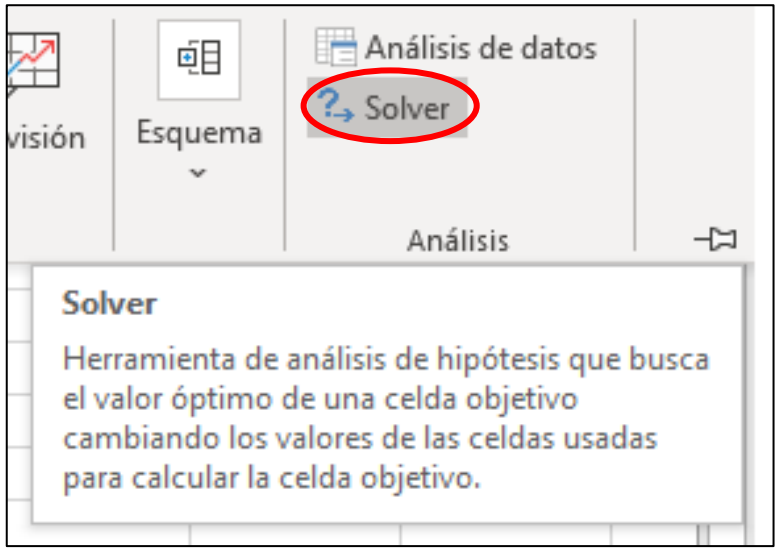
3) Armamos la función de excel para el funcional del modelo:

MULTIPLO.S... | =C2*C1+E2*E1

	A	B	C	D	E	F	G	H	I	J
1		x	0	y	0					
2	Coef. Func.	c_x	5	c_y	3		Objetivo	*E1		
3										bi
4	Coef. Tec.	a_1x	3	a_1y	2		Restr. 1	0	<=	2400
5		a_2x	0	a_2y	1		Restr. 2	0	<=	800
6		a_3x	2	a_3y	0		Restr. 3	0	<=	1200

Solver de MS Excel

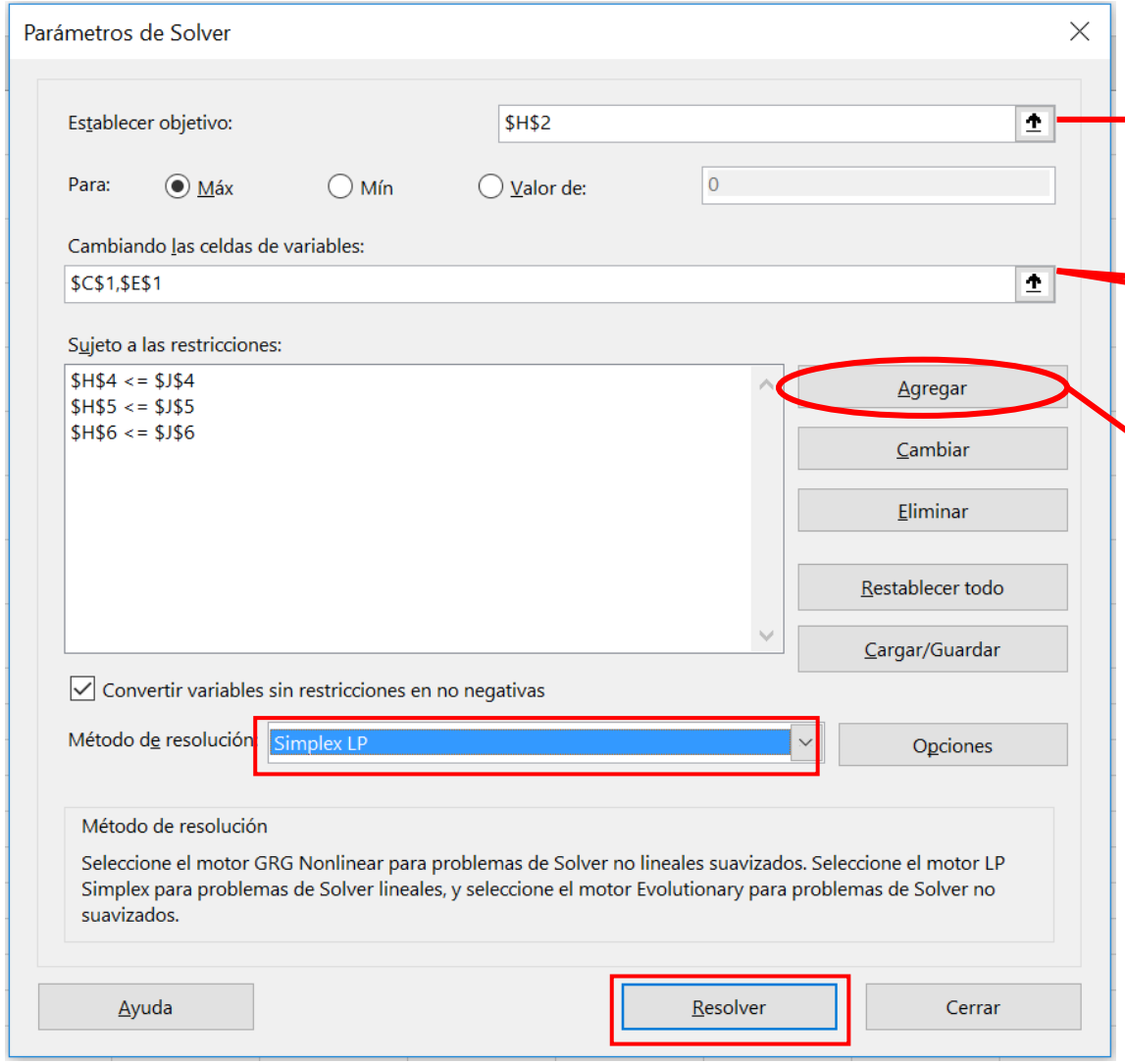
4) Acceso al solver:

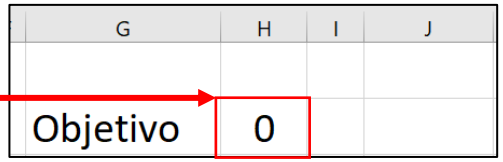


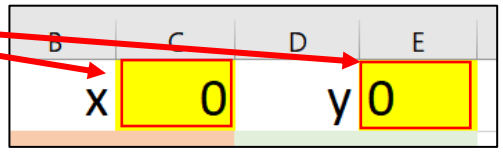
Tutorial de Microsoft para agregar add-in Solver de Excel:
https://support.office.com/es-es/article/carga-del-complemento-solver-en-excel-2016-612926fc-d53b-46b4-872c-e24772f078ca

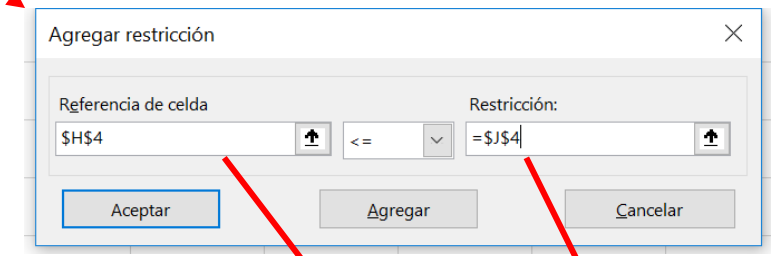
Solver de MS Excel

5) Configuración del problema:





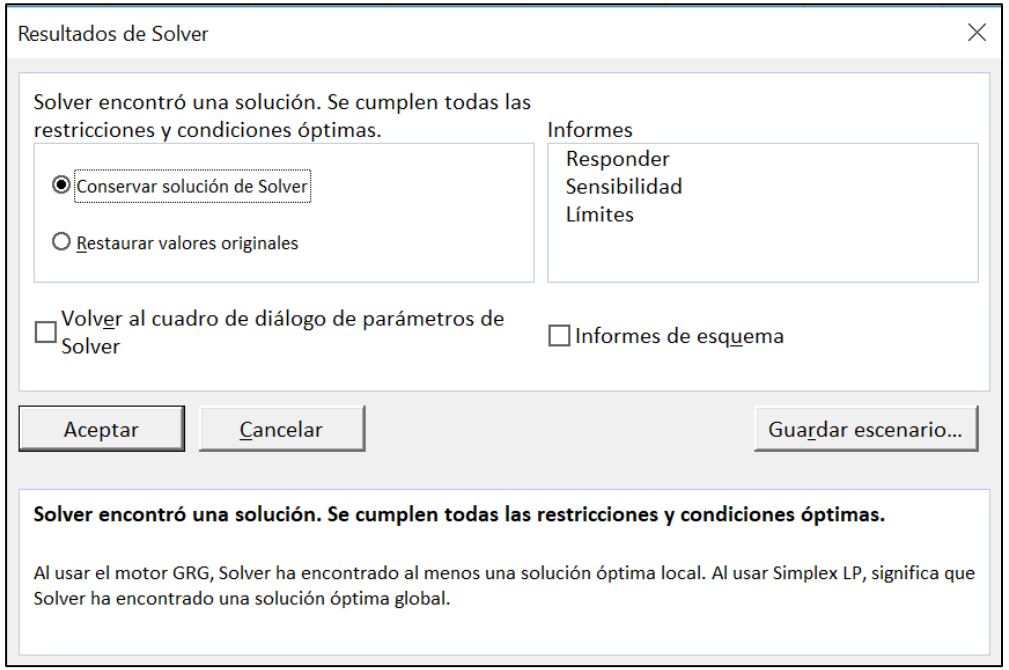




			bi
Restr. 1	0	<=	2400
Restr. 2	0	<=	800
Restr. 3	0	<=	1200

Solver de MS Excel

6) Configuración de resultados:



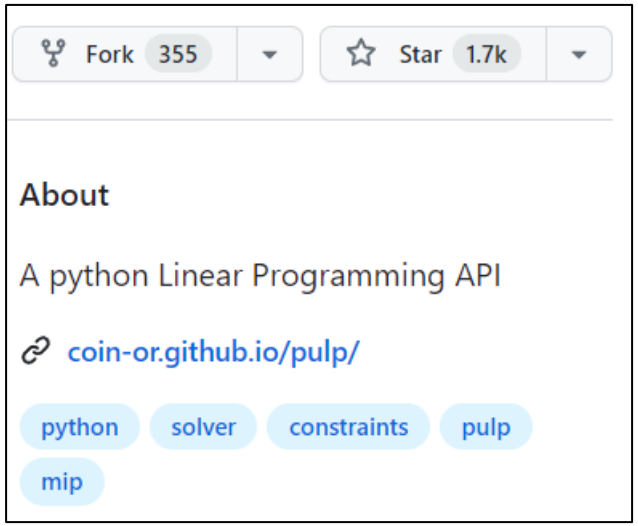
7) El solver pega como valores los resultados:

	A	B	C	D	E	F	G	H	I	J
1			x	600	y	300				
2	Coef. Func.	c_x	5	c_y	3		Objetivo	3900		
3										bi
4	Coef. Tec.	a_1x	3	a_1y	2		Restr. 1	2400	<=	2400
5		a_2x	0	a_2y	1		Restr. 2	300	<=	800
6		a_3x	2	a_3y	0		Restr. 3	1200	<=	1200

Librería PuLP Python

PuLP:

- **Librería de Python** que permite modelizar problemas mediante **programación matemática**. https://github.com/coin-or/pulp



- Pertenece a la organización **COIN-OR** que se dedica a desarrollar soft de Investigación Operativa open-source.

COIN-OR Foundation

Computational Infrastructure for Operations Research.

297 followers | United States of America | https://www.coin-or.org | @coin_or | info@coin-or.org

Ejemplo librería PuLP Python

Ejemplo:

$$max\ z = 5x + 3y$$
$$st:$$
$$3x + 2y \leq 2400$$
$$y \leq 800$$
$$2x \leq 1200$$

1) Importamos el paquete y definimos un problema lineal de maximización:

```
import pulp

lp = pulp.LpProblem("ejercicio02", pulp.LpMaximize)
```

**Solver por defecto: COIN-OR LP (Clp)*

2) Creamos las variables de decisión:

```
# Variables:
x = pulp.LpVariable('x', lowBound=0, cat='Continuous')
y = pulp.LpVariable('y', lowBound=0, cat='Continuous')
```

Nombre Cota Categoría

Ejemplo librería PuLP Python

Ejemplo:

$$max\ z = 5x + 3y$$
$$st:$$
$$3x + 2y \leq 2400$$
$$y \leq 800$$
$$2x \leq 1200$$

3) Creamos la función objetivo:

```
# Función objetivo:
lp += 5*x + 3*y, "Z"
```

4) Creamos las restricciones:

```
# Restricciones
lp += 3*x + 2*y <= 2400
lp +=       y <=  800
lp += 2*x     <= 1200
```

Ejemplo librería PuLP Python

Ejemplo:

$$max\ z = 5x + 3y$$
$$st:$$
$$3x + 2y \leq 2400$$
$$y \leq 800$$
$$2x \leq 1200$$

```
Optimal
x = 600.00
y = 300.00
3900.0
```

5) Resolvemos el modelo:

```
lp.solve()
```

6) Imprimimos resultados:

```
print(pulp.LpStatus[lp.status])

for variable in lp.variables():
    print(f"{variable.name:s} = {variable.varValue:.2f}")

print(pulp.value(lp.objective))
```