Fun with Gurobi v6.5.2

A Simple LP in Excel Solver

	X1	X2	X3	X4			
Solution							
Lower Bound		-10					
Upper Bound	1	10		10	Max		
Objective				1			
	4.1	-1				=	0
	1		1			>=	0.4
	1		-1			<=	0.4
	-0.230438			1		<=	3.4695627

Excel: Solve (1)

	X1	X2	Х3	X4			
Solution	1	4.1	0.6	3.7000007			
Lower Bound		-10					
Upper Bound	1	10		10	Max		
Objective				1	3.7000007		
	4.1	-1			0	=	0
	1		1		1.6	>=	0.4
	1		-1		0.4	<=	0.4
	-0.230438			1	3.4695627	<=	3.4695627

Excel: Solve (2)

	X1	X2	X3	X4			
Solution	1	4.1	0.6	3.699997			
Lower Bound		-10					
Upper Bound	1	10		10	Max		
Objective				1	3.699997		
	4.1	-1			0	=	0
	1		1		1.6	>=	0.4
	1		-1		0.4	<=	0.4
	-0.230438			1	3.469559	<=	3.4695627
New Constr.	3.892116	-1		1	3.492113	<=	3.492113

Excel: Solve (3)

	X1	X2	<i>X3</i>	X4			
Solution	1	4.1	0.6	3.699997			
Lower Bound		-10					
Upper Bound	1	10		10	Max		
Objective				1	3.699997		
	4.1	-1			0	=	0
	1		1		1.6	>=	0.4
	1		-1		0.4	<=	0.4
	-0.230438			1	3.469559	<=	3.4695627
New Constr.	3.892116	-1		1	3.492113	<=	3.492113
New Constr.				1	3.699997	<=	20

GurobiPy: Solve (1)

```
from gurobipy import *
m = Model("Gurobi Bug")
                                                  Changed value of parameter Presolve to 0
m.setParam("Presolve", 0)
                                                    Prev: -1 Min: -1 Max: 2 Default: -1
                                                 Optimize a model with 4 rows, 4 columns and 8 nonzeros
X1 = m.addVar(lb=-GRB.INFINITY, ub=1)
                                                 Coefficient statistics:
X2 = m.addVar(lb=-10, ub=10)
                                                   Matrix range
                                                                  [2e-01, 4e+00]
X3 = m.addVar(lb=-GRB.INFINITY, ub=GRB.INFINITY)
X4 = m.addVar(lb=-GRB.INFINITY, ub=10, obj=1)
                                                   Objective range [1e+00, 1e+00]
                                                   Bounds range
                                                                  [1e+00, 1e+01]
                                                    RHS range [4e-01, 3e+00]
m.modelSense = GRB.MAXIMIZE
                                                 Iteration Objective
                                                                            Primal Inf.
                                                                                           Dual Inf.
                                                                                                           Time
                                                        0
                                                              handle free variables
                                                                                                             03
m.update()
                                                             3.7000007e+00 0.000000e+00 0.000000e+00
                                                                                                             03
C1 = m.addConstr(X2 == 4.1*X1)
                                                  Solved in 3 iterations and 0.01 seconds
C2 = m.addConstr(X1 >= 0.4 - X3)
C3 = m.addConstr(X3 + 0.4 >= X1)
                                                 Optimal objective 3.700000700e+00
C4 = m.addConstr(-0.230438*X1 + X4 <= 3.4695627)
m.update()
```

m.optimize()

GurobiPy: Solve (2)

```
from gurobipy import *
                                               Changed value of parameter Presolve to 0
                                                  Prev: -1 Min: -1 Max: 2 Default: -1
m = Model("Gurobi Bug")
                                               Optimize a model with 4 rows, 4 columns and 8 nonzeros
m.setParam("Presolve", 0)
                                               Coefficient statistics:
                                                 Matrix range
                                                                [2e-01, 4e+00]
X1 = m.addVar(lb=-GRB.INFINITY, ub=1)
                                                 Objective range [1e+00, 1e+00]
X2 = m.addVar(lb=-10, ub=10)
X3 = m.addVar(lb=-GRB.INFINITY, ub=GRB.INFINITY) Bounds range [1e+00, 1e+01]
X4 = m.addVar(lb=-GRB.INFINITY, ub=10, obj=1)
                                                 RHS range [4e-01, 3e+00]
                                               Iteration Objective
                                                                           Primal Inf. Dual Inf.
                                                                                                        Time
                                                           handle free variables
                                                                                                          0.3
m.modelSense = GRB.MAXIMIZE
                                                      3 3.7000007e+00 0.000000e+00 0.000000e+00
                                                                                                          0.3
m.update()
                                               Solved in 3 iterations and 0.01 seconds
                                               Optimal objective 3.700000700e+00
C1 = m.addConstr(X2 == 4.1*X1)
C2 = m.addConstr(X1 >= 0.4 - X3)
                                               Optimize a model with 5 rows, 4 columns and 11 nonzeros
                                               Coefficient statistics:
C3 = m.addConstr(X3 + 0.4 >= X1)
C4 = m.addConstr(-0.230438*X1 + X4 <= 3.4695627) Matrix range [2e-01, 4e+00]
                                                 Objective range [1e+00, 1e+00]
                                                 Bounds range [1e+00, 1e+01]
m.update()
                                                 RHS range [4e-01, 3e+00]
                                               Iteration Objective
                                                                           Primal Inf. Dual Inf.
                                                                                                        Time
m.optimize()
                                                           3.7000007e+00 0.000000e+00 0.000000e+00
                                                                                                          03
m.addConstr(3.892116*X1 - X2 + X4 <= 3.492113)
                                                      1 3.6999970e+00 0.000000e+00 0.000000e+00
                                                                                                          0.3
                                               Solved in 1 iterations and 0.01 seconds
m.update()
                                               Optimal objective 3.699997000e+00
m.optimize()
```

GurobiPy: Solve (3)

```
from gurobipy import *
m = Model("Gurobi Bug")
m.setParam("Presolve", 0)
X1 = m.addVar(lb=-GRB.INFINITY, ub=1)
X2 = m.addVar(lb=-10, ub=10)
X3 = m.addVar(lb=-GRB.INFINITY, ub=GRB.INFINITY)
X4 = m.addVar(lb=-GRB.INFINITY, ub=10, obj=1)
m.modelSense = GRB.MAXIMIZE
m.update()
C1 = m.addConstr(X2 == 4.1*X1)
C2 = m.addConstr(X1 >= 0.4 - X3)
C3 = m.addConstr(X3 + 0.4 >= X1)
C4 = m.addConstr(-0.230438*X1 + X4 <= 3.4695627)
m.update()
m.optimize()
m.addConstr(3.892116*X1 - X2 + X4 <= 3.492113)
m.update()
m.optimize()
                                 Obi = 3.699997
m.addConstr(X4 <= 20)
                                 X1 = 1.0
                                 X2 = 4.1
m.update()
                                 X3 = 0.6
                                 X4 = 3.699997
m.optimize()
                                 Dual on C1 = 1.0
```

```
Changed value of parameter Presolve to 0
   Prev: -1 Min: -1 Max: 2 Default: -1
Optimize a model with 4 rows, 4 columns and 8 nonzeros
Coefficient statistics:
 Matrix range
                [2e-01, 4e+00]
 Objective range [1e+00, 1e+00]
 Bounds range
                [1e+00, 1e+01]
 RHS range
                [4e-01, 3e+00]
Iteration Objective
                            Primal Inf.
                                          Dual Inf.
                                                         Time
           handle free variables
                                                          03
           3.7000007e+00 0.000000e+00
                                         0.000000e+00
                                                          0.3
Solved in 3 iterations and 0.01 seconds
Optimal objective 3.700000700e+00
Optimize a model with 5 rows, 4 columns and 11 nonzeros
Coefficient statistics:
 Matrix range [2e-01, 4e+00]
 Objective range [1e+00, 1e+00]
 Bounds range [1e+00, 1e+01]
 RHS range
                [4e-01, 3e+00]
Iteration Objective
                           Primal Inf.
                                          Dual Inf.
                                                        Time
           3.7000007e+00 0.000000e+00
      0
                                         0.000000e+00
                                                          0s
      1 3.6999970e+00 0.000000e+00
                                         0.000000e+00
                                                          0.3
Solved in 1 iterations and 0.01 seconds
Optimal objective 3.699997000e+00
Optimize a model with 6 rows, 4 columns and 12 nonzeros
Coefficient statistics:
 Matrix range [2e-01, 4e+00]
 Objective range [1e+00, 1e+00]
 Bounds range [1e+00, 1e+01]
 RHS range
               [4e-01, 2e+01]
Iteration Objective
                            Primal Inf.
                                          Dual Inf.
                                                         Time
           3.6999970e+00 0.000000e+00
                                         0.0000000e+00
                                                          0.8
Solved in 0 iterations and 0.01 seconds
Optimal objective 3.699997000e+00
```

Now lets do something redundant

```
Obj = m.getAttr("Obj", m.getVars())
m.setAttr("Obj", m.getVars(), obj)
```

Audience Participation Time

```
from gurobipy import *
m = Model("Gurobi Bug")
m.setParam("Presolve", 0)
X1 = m.addVar(lb=-GRB.INFINITY, ub=1)
X2 = m.addVar(lb=-10, ub=10)
X3 = m.addVar(lb=-GRB.INFINITY, ub=GRB.INFINITY)
X4 = m.addVar(lb=-GRB.INFINITY, ub=10, obj=1)
m.modelSense = GRB.MAXIMIZE
m.update()
C1 = m.addConstr(X2 == 4.1*X1)
C2 = m.addConstr(X1 >= 0.4 - X3)
C3 = m.addConstr(X3 + 0.4 >= X1)
C4 = m.addConstr(-0.230438*X1 + X4 <= 3.4695627)
m.update()
m.optimize()
m.addConstr(3.892116*X1 - X2 + X4 <= 3.492113)
m.update()
m.optimize()
obj = m.getAttr("Obj", m.getVars())
m.setAttr("Obj", m.getVars(), obj)
m.addConstr(X4 <= 20)
m.update()
m.optimize()
```

a) Unbounded

b) Infeasible

c) 3.699997 optimal solution

d) 3.7000007 first solution

e) 20 RHS of the last constraint

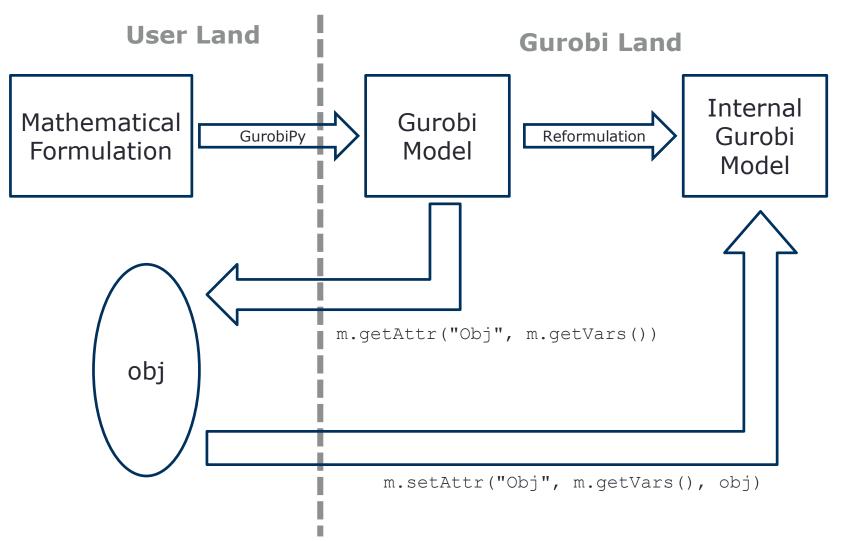
The answer is e) 20!

Changed value of parameter Presolve to 0

```
Prev: -1 Min: -1 Max: 2 Default: -1
from gurobipy import *
                                                    Optimize a model with 4 rows, 4 columns and 8 nonzeros
                                                    Coefficient statistics:
m = Model("Gurobi Bug")
                                                      Matrix range
                                                                       [2e-01, 4e+001
m.setParam("Presolve", 0)
                                                      Objective range [1e+00, 1e+00]
                                                      Bounds range
                                                                       [1e+00, 1e+01]
X1 = m.addVar(lb=-GRB.INFINITY, ub=1)
                                                      RHS range
                                                                       [4e-01, 3e+00]
X2 = m.addVar(lb=-10, ub=10)
                                                    Iteration
                                                                  Objective
                                                                                  Primal Inf.
                                                                                                  Dual Inf.
                                                                                                                 Time
X3 = m.addVar(lb=-GRB.INFINITY.
                               ab=GRB.INFIN
                                                                 handle free variables
                                                                                                                   0.8
X4 = m.addVar(lb=-GRB.INFINITY
                               ub=10, obj=1
                                                                 3.7000007e+00 0.000000e+00
                                                                                                 0.000000e+00
m.modelSense = GRB.MAXIMIZE
                                                    Solved in 3 iterations and 0.01 seconds
                                                    Optimal objective 3.700000700e+00
m.update()
                                                    Optimize a model with 5 rows, 4 columns and 11 nonzeros
                                                    Coefficient statistics:
C1 = m.addConstr(X2 == 4.1*X1)
                                                      Matrix range
                                                                       [2e-01, 4e+00]
C2 = m.addConstr(X1 >= 0.4 - X3)
                                                      Objective range [1e+00, 1e+00]
C3 = m.addConstr(X3 + 0.4 >= X1)
                                                      Bounds range
                                                                       [1e+00, 1e+01]
C4 = m.addConstr(-0.230438*X1 + X4 <= 3.4695627)
                                                      RHS range
                                                                       [4e-01, 3e+00]
m.update()
                                                    Iteration
                                                                Objective
                                                                                  Primal Inf.
                                                                                                 Dual Inf.
                                                                                                                 Time
                                                                 3.7000007e+00
                                                                                 0.000000e+00
                                                                                                 0.000000e+00
                                                                                                                   0s
m.optimize()
                                                                 3.6999970e+00 0.000000e+00
                                                                                                 0.000000e+00
                                                                                                                   0.8
m.addConstr(3.892116*X1 - X2 + X4 \le 3.492113)
                                                    Solved in 1 iterations and 0.01 seconds
                                                    Optimal objective 3.699997000e+00
m.update()
                                                    Optimize a model with 6 rows, 4 columns and 12 nonzeros
                                                    Coefficient statistics:
m.optimize()
                                                      Matrix range
                                                                       [2e-01, 4e+00]
                                                      Objective range [1e+00, 1e+00]
obj = m.getAttr("Obj", m.getVars())
                                                      Bounds range
                                                                       [1e+00, 1e+01]
m.setAttr("Obj", m.getVars(), obj)
                                                      RHS range
                                                                       [4e-01, 2e+01]
                                    20.0
                                                    Iteration
                                                                                                 Dual Inf.
                                                                 Objective
                                                                                  Primal Inf.
                                                                                                                 Time
m.addConstr(X4 <= 20)
                                   -2.43902439024
                                                                 4.2078840e+30
                                                                                7.232884e+30
                                                                                                 4.207884e+00
                                                                                                                   0s
                                                                 2.0000000e+01 0.000000e+00
                                                                                                 0.000000e+00
                                   -10.0
                                                                                                                   0s
m.update()
                                   2.83902439024
                                                    Solved in 3 iterations and 0.01 seconds
m.optimize()
                                  -20.0
                                                    Optimal objective 2.000000000e+01
                            Dual on C1 = 0.0
```

Obj = 3.699997 X1 = 1.0 X2 = 4.1 X3 = 0.6 X4 = 3.699997 Dual on C1 = 1.0

So what's going on?



So what's going on?

```
max - \mathbf{x4} s.t.

4.1\mathbf{x}1 - \mathbf{x}2 == 0

\mathbf{x}1 + \mathbf{x}3 >= 0.4

\mathbf{x}1 - \mathbf{x}3 <= 0.4

-0.230438\mathbf{x}1 - \mathbf{x}4 <= 3.4695627

3.892116\mathbf{x}1 - \mathbf{x}2 - \mathbf{x}4 <= 3.492113

\mathbf{x}1 \in (-\infty, 1] \ \mathbf{x}2 \in [-10, 10]

\mathbf{x}3 \in (-\infty, \infty) \ \mathbf{x}4 \in [-10, \infty)
```

What we know from Gurobi

You need

- 1. A variable with a finite upper bound and an infinite lower bound

 Value to go variable in SDDP...
- 2. A model that Gurobi chooses not to rescale Hard to control...
- 3. To set some coefficients in memory and resolve

Which is what SDDP does...