GUROBI & PYTHON: THE DAWN OF A NEW ARE IN NUMERICAL OPTIMIZATION?

Mizzou Engineering
University of Missouri
engineering.missouri.edu

Mustafa Y. Sir
(sirm@missouri.edu)
University of Missouri

Background

- My area of research is applied operations research
- *Operations Research:* the discipline of applying advanced analytical methods to help make better decisions (INFORMS).
- Numerical Optimization?! Isn't optimization by definition numerical?

Background

- My research interest are mathematical modeling and (mixed) integer / convex / dynamic / stochastic programming.
- I work in many application domains but mainly in healthcare/medicine and logistics.
- I can spend several hours writing computer code without being distracted (Ask Esra!).
- I use C++, Matlab, AMPL, Visual Basic, Excel, SAS, CPLEX, Gurobi, and Python for my research.

Outline

- Review of Optimization Software Packages
- C++
- Pyhton
- Gurobi
- CPLEX

General Structure of an Optimization Package

INTERFACE: This is where you define your math model

Your model gets converted into a format which a solver can read (LP or MPS)

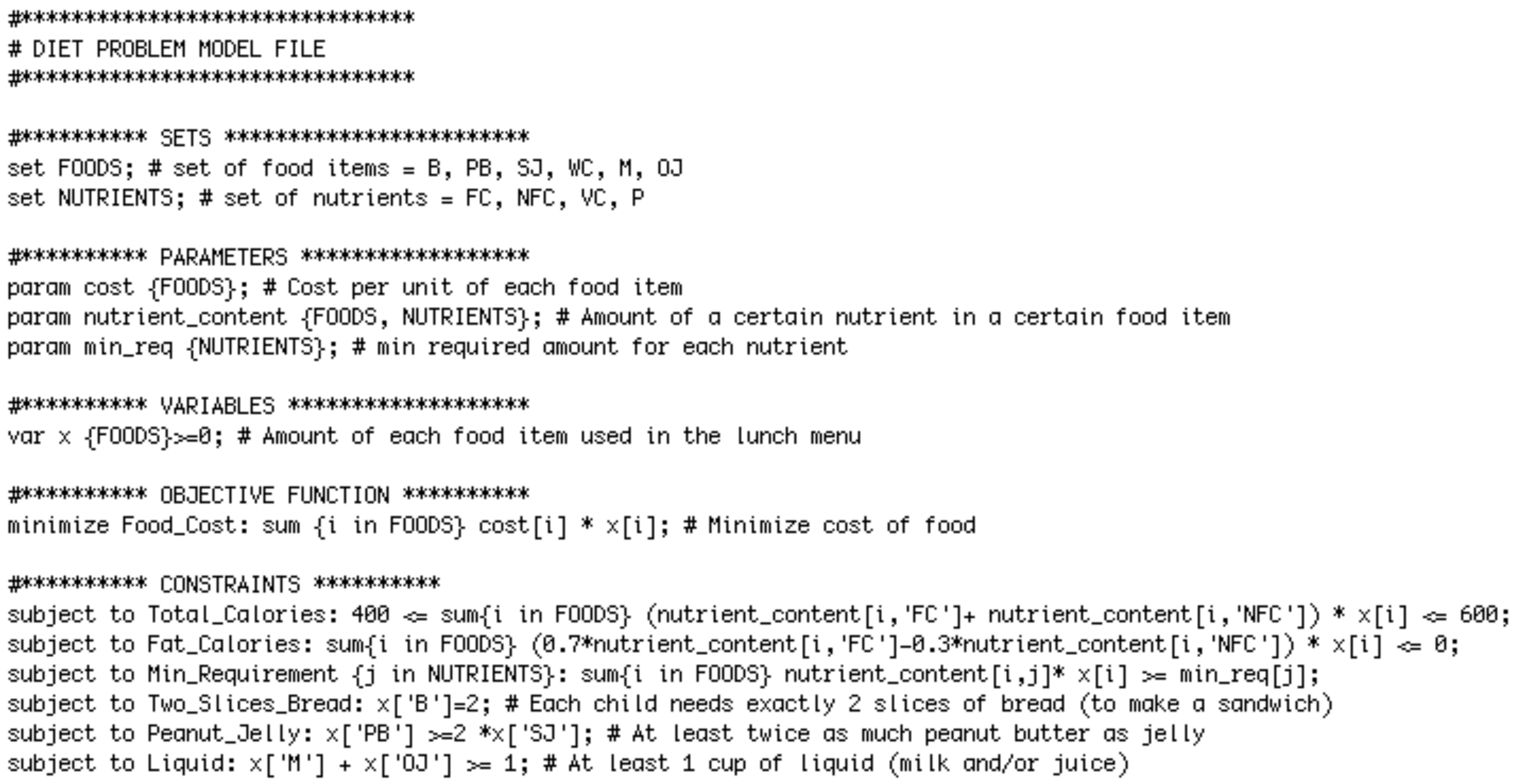
SOLVER: This is where your model gets solved

Solver returns the solution in a format you specify

Types of Interface

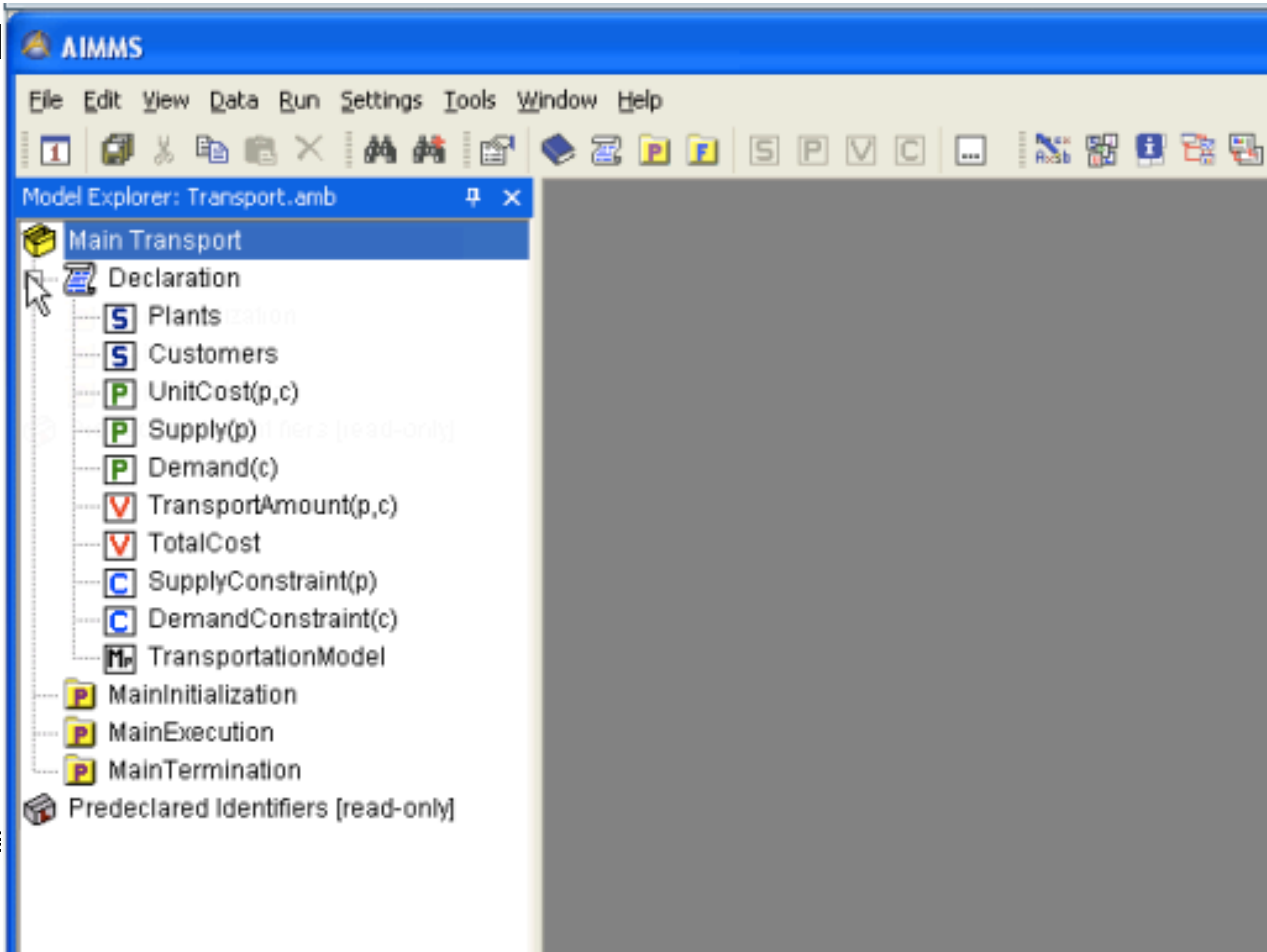
- There are two types of interface.
- The first one is that of a modeling system.
- Modeling systems have their own syntax (e.g., AMPL, GAMMS) to define a model and sometimes a graphical user interface (e.g., AIMMS).
- When you are using a modeling system, you do not directly interact with the solver.

Types of Interface



```
#*******************************
# DIET PROBLEM MODEL FILE
#*******************************

#********** SETS ************************
set FOODS; # set of food items = B, PB, SJ, WC, M, OJ
set NUTRIENTS; # set of nutrients = FC, NFC, VC, P

#********** PARAMETERS ******************
param cost {FOODS}; # Cost per unit of each food item
param nutrient_content {FOODS, NUTRIENTS}; # Amount of a certain nutrient in a certain food item
param min_req {NUTRIENTS}; # min required amount for each nutrient

#********** VARIABLES *******************
var x {FOODS}>=0; # Amount of each food item used in the lunch menu

#********** OBJECTIVE FUNCTION **********
minimize Food_Cost: sum {i in FOODS} cost[i] * x[i]; # Minimize cost of food

#********** CONSTRAINTS **********
subject to Total_Calories: 400 <= sum{i in FOODS} (nutrient_content[i,'FC']+ nutrient_content[i,'NFC']) * x[i] <= 600;
subject to Fat_Calories: sum{i in FOODS} (0.7*nutrient_content[i,'FC']-0.3*nutrient_content[i,'NFC']) * x[i] <= 0;
subject to Min_Requirement {j in NUTRIENTS}: sum{i in FOODS} nutrient_content[i,j]* x[i] >= min_req[j];
subject to Two_Slices_Bread: x['B']=2; # Each child needs exactly 2 slices of bread (to make a sandwich)
subject to Peanut_Jelly: x['PB'] >=2 *x['SJ']; # At least twice as much peanut butter as jelly
subject to Liquid: x['M'] + x['OJ'] >= 1; # At least 1 cup of liquid (milk and/or juice)
```

Types of Interface


AIMMS
File Edit View Data Run Settings Tools Window Help
Model Explorer: Transport.amb

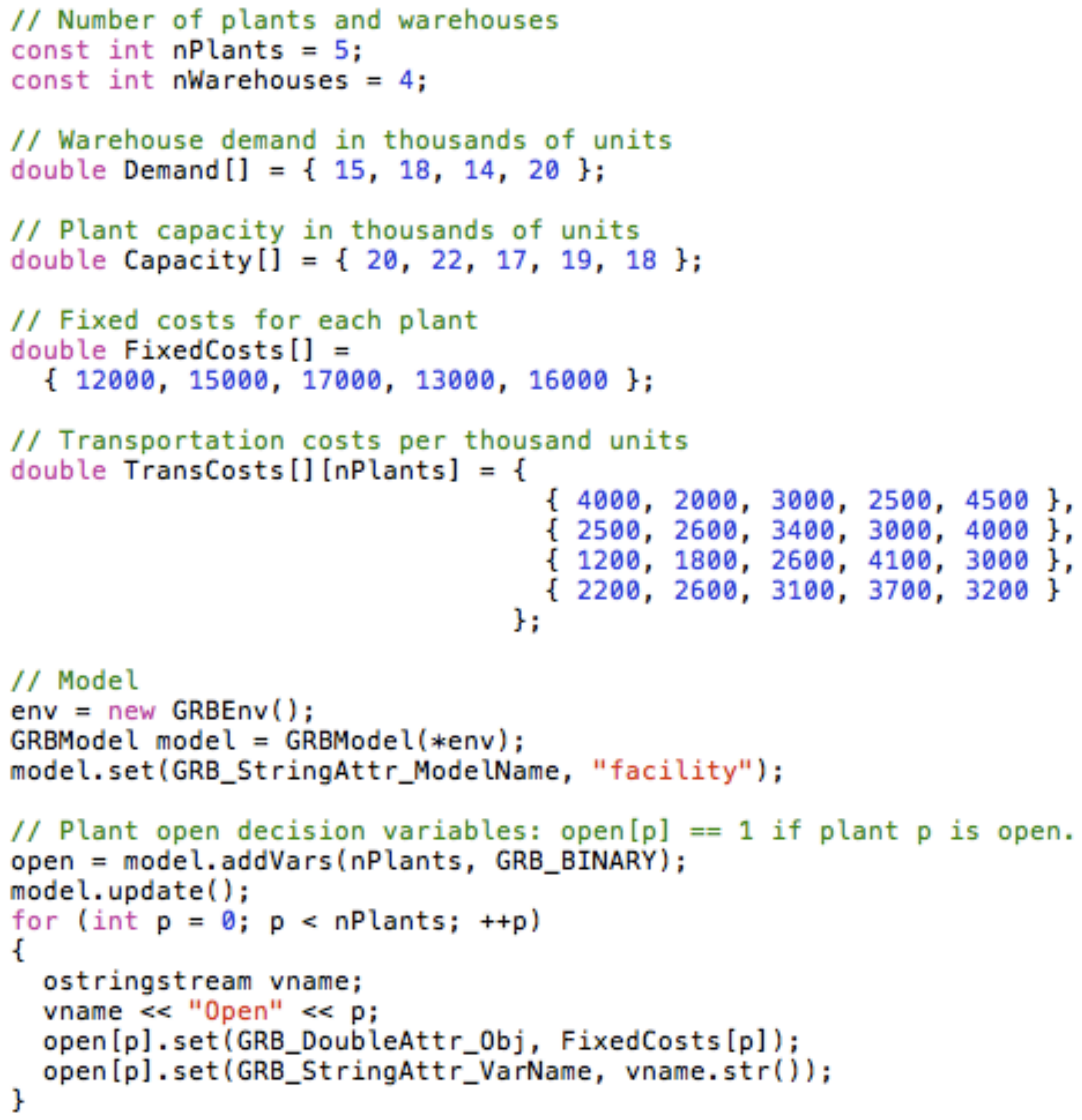
Main Transport
- Declaration
  - Plants
  - Customers
  - UnitCost(p,c)
  - Supply(p)
  - Demand(c)
  - TransportAmount(p,c)
  - TotalCost
  - SupplyConstraint(p)
  - DemandConstraint(c)
  - TransportationModel
- MainInitialization
- MainExecution
- MainTermination

Predeclared Identifiers [read-only]

Types of Interface

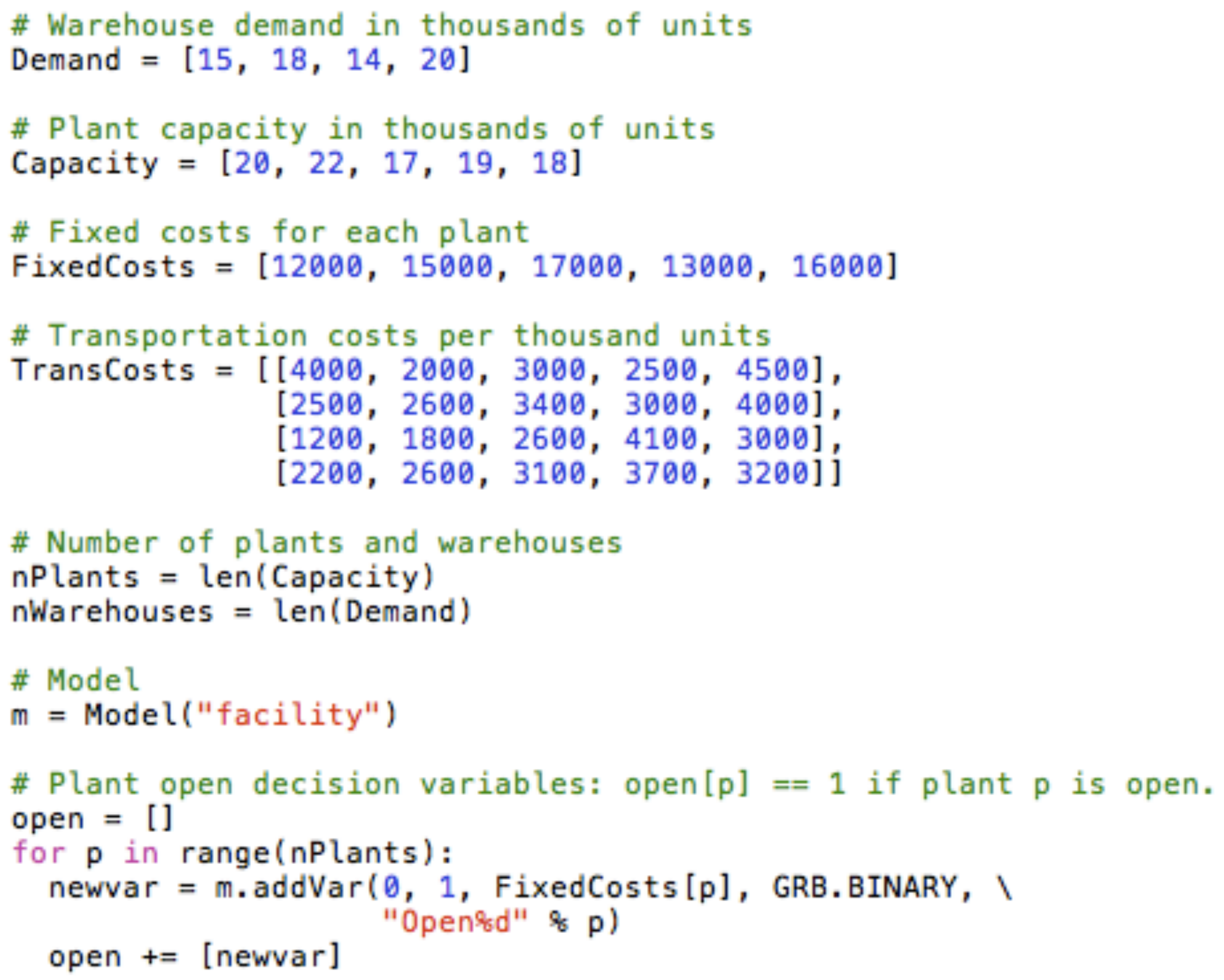
- Most solvers provide libraries in various programming languages which allows users to interact with the solver via several functions and objects.
- This type of interface is much more flexible:
  - Ability to control of what solver does
  - Ability to design heuristics/decomposition algorithms
  - Ability to control memory allocations
  - Flexibility in data format
- Downside is the time it takes to write the code.

Types o



```
// Number of plants and warehouses
const int nPlants = 5;
const int nWarehouses = 4;

// Warehouse demand in thousands of units
double Demand[] = { 15, 18, 14, 20 };

// Plant capacity in thousands of units
double Capacity[] = { 20, 22, 17, 19, 18 };

// Fixed costs for each plant
double FixedCosts[] =
  { 12000, 15000, 17000, 13000, 16000 };

// Transportation costs per thousand units
double TransCosts[][nPlants] = {
                                { 4000, 2000, 3000, 2500, 4500 },
                                { 2500, 2600, 3400, 3000, 4000 },
                                { 1200, 1800, 2600, 4100, 3000 },
                                { 2200, 2600, 3100, 3700, 3200 }
                              };

// Model
env = new GRBEnv();
GRBModel model = GRBModel(*env);
model.set(GRB_StringAttr_ModelName, "facility");

// Plant open decision variables: open[p] == 1 if plant p is open.
open = model.addVars(nPlants, GRB_BINARY);
model.update();
for (int p = 0; p < nPlants; ++p)
{
  ostringstream vname;
  vname << "Open" << p;
  open[p].set(GRB_DoubleAttr_Obj, FixedCosts[p]);
  open[p].set(GRB_StringAttr_VarName, vname.str());
}
```

Types of Interface



```
# Warehouse demand in thousands of units
Demand = [15, 18, 14, 20]

# Plant capacity in thousands of units
Capacity = [20, 22, 17, 19, 18]

# Fixed costs for each plant
FixedCosts = [12000, 15000, 17000, 13000, 16000]

# Transportation costs per thousand units
TransCosts = [[4000, 2000, 3000, 2500, 4500],
              [2500, 2600, 3400, 3000, 4000],
              [1200, 1800, 2600, 4100, 3000],
              [2200, 2600, 3100, 3700, 3200]]

# Number of plants and warehouses
nPlants = len(Capacity)
nWarehouses = len(Demand)

# Model
m = Model("facility")

# Plant open decision variables: open[p] == 1 if plant p is open.
open = []
for p in range(nPlants):
  newvar = m.addVar(0, 1, FixedCosts[p], GRB.BINARY, \
                    "Open%d" % p)
  open += [newvar]
```

C++

- C++ is one of the most popular programming languages ever created
- It is the object-oriented version of C.
- C++ is widely used in the software industry.
- It is robust, very fast, flexible....
- BUT...

C++

- First a few technical notes…
- Standard C++ provides no language features to create multi-threaded software.
- It is bloated and its pieces do not fit together well.
- Pointers and dynamic arrays are very confusing.
- Strongly-typed nature (usually viewed as an advantage by software engineers) makes it hard to write clear code.

C++

- What others say about C++…
- "Writing in C or C++ is like running a chain saw with all the safety guards removed." (Bob Gray)
- "C++: Hard to learn and built to stay that way."
- "Fifty years of programming language research, and we end up with C++ ???" (Richard A. O'Keefe*)*
- "When your hammer is C++, everything begins to look like a thumb." (Steve Haflich)
- "C++ sucks!!!" (Esra Sisikoglu)

Alternative to C++: A scripting language

- Dynamic typing: little or no distinction between integer, floating- point, or string variables.
- Arrays can mix elements of different “types,” such as integers and strings.
- Functions can return nonscalars, e.g., arrays.
- Nonscalars can be used as loop indexes.
- Lots of high-level operations intrinsic to the language, e.g. string concatenation and stack push/ pop.
- Interpreted, rather than being compiled.

Taken from Norman Marloff’s Python Tutorial

Python? What is it?

According to Guido van Rossum, the creator of the Python language, Python is

- Object-oriented rapid prototyping language
- Easy to learn, read, use
- Extensible (add new modules)
  - C/C++/Fortran/Java/whatever
- Embeddable in applications
- Open source (This is indeed the best part!)

Taken from python.org

Python vs. C++ (or Java)

- Code is up to 5 times shorter and more readable
- Dynamic typing
- Quicker development
  - no compilation phase
  - less typing
- It may run a bit slower
  - but development is much faster
  - and Python uses less memory, which is important for optimization

Taken from python.org

Still, what is so cool about Python?

Sequences in Python

```
>>> x = [5, 12, 13, 200]
>>> x
[5, 12, 13, 200]
```

Still, what is so cool about Python?

Sequences in Python

```
>>> x = [5, 12, 13, 200]
>>> x
[5, 12, 13, 200]
>>> x.append(-10)
>>> x
[5, 12, 13, 200, -10]
```

Still, what is so cool about Python?

Sequences in Python

```
>>> x = [5, 12, 13, 200]
>>> x
[5, 12, 13, 200]
>>> x.append(-10)
>>> x
[5, 12, 13, 200, -10]
>>> del x[2]
>>> x
[5, 12, 200, -10]
>>>
```

Still, what is so cool about Python?

Sequences in Python

```
>>> x = [5, 12, 13, 200]
>>> x
[5, 12, 13, 200]
>>> x.append(-10)
>>> x
[5, 12, 13, 200, -10]
>>> del x[2]
>>> x
[5, 12, 200, -10]
>>> z = x[1:3]
>>> z
[12, 200]
```

Still, what is so cool about Python?

Many more built-in functions that makes programming enjoyable

```
>>> x
[5, 12, 200, -10]
>>> x.insert(2,37)
>>> x
[5, 12, 37, 200, -10]
>>> x.remove(200)
>>> x
[5, 12, 37, -10]
>>> x.index(37)
2
```

Try to do these in C ++ and see if you are able to keep your sanity

Still, what is so cool about Python?

Easy concatenation of sequences

```
>>> x
[5, 12, 200, -10]
>>> y = ['mustafa', 'sir']
>>> y
['mustafa', 'sir']
>>> z = x + y
>>> z
[5, 12, 37, -10, 'mustafa', 'sir']
>>>
```

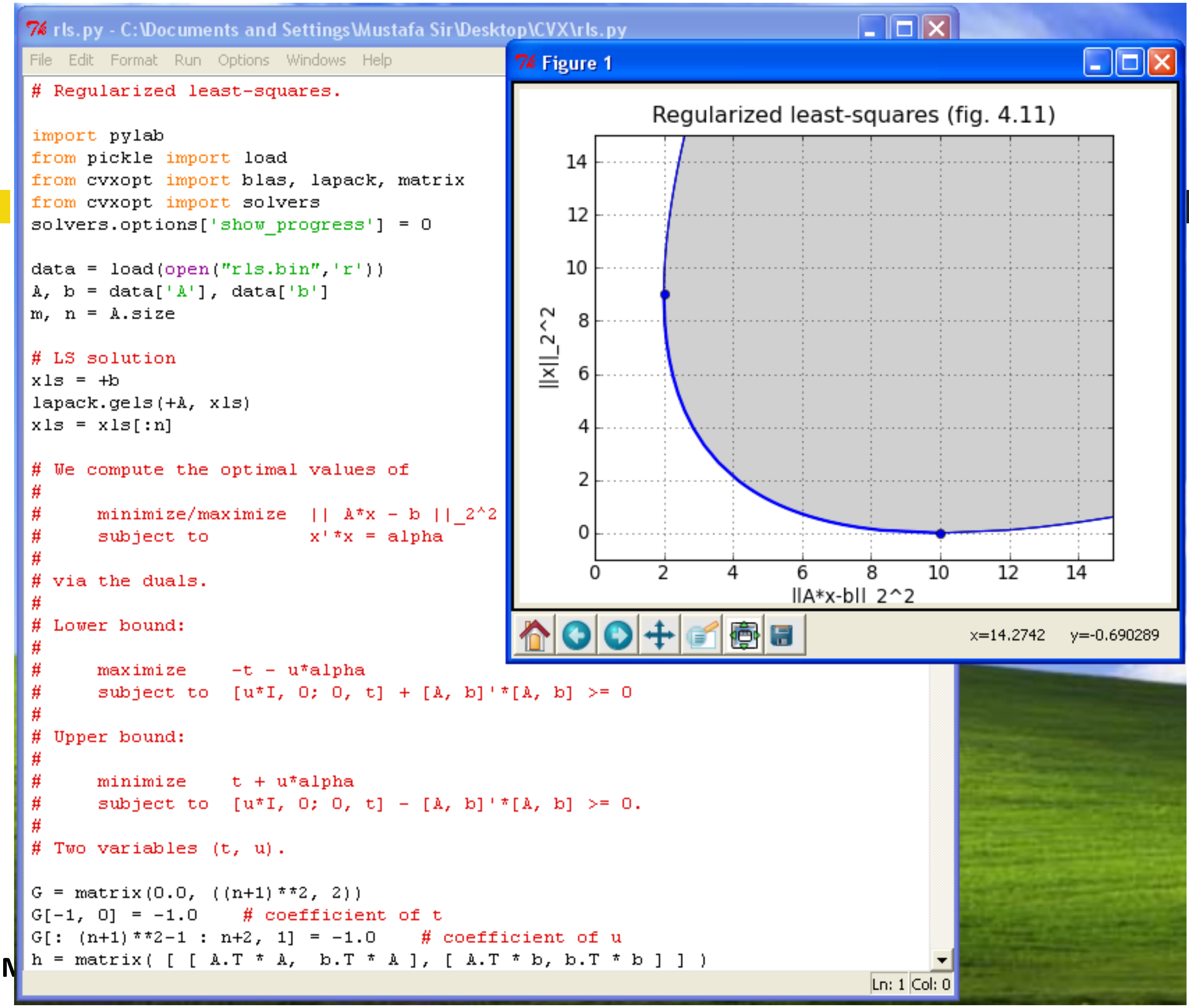
Still, what is so cool about Python?

Sequences can mix elements of different “types”

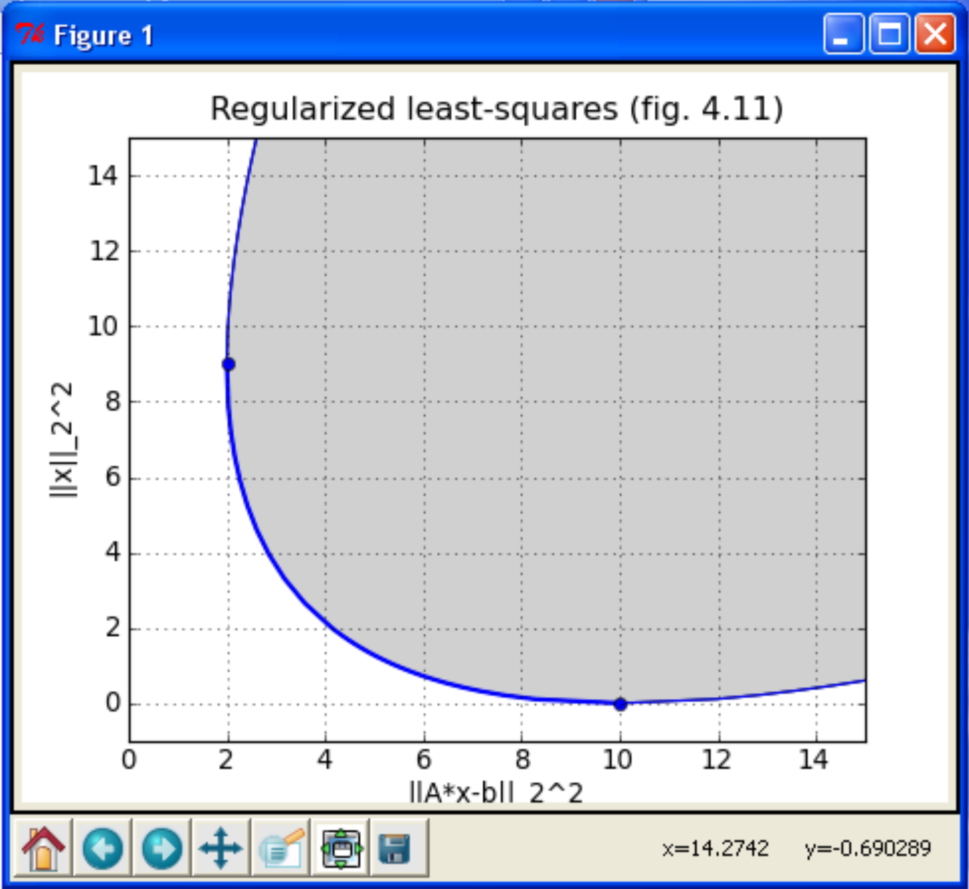
```
>>> Nurse1 = ['Laura', 'RN', 35, [[1, 3],[4,2], [5,3],[6,3],[7,3]],'Full Time']
>>> Nurse2 = ['Jane', 'NA', 20, [[6,1],[7,1]],'Part Time']
>>> Nurses = [Nurse1, Nurse2]
>>> Nurses[0][1]
'RN'
>>> Nurses[1][3]
[[6, 1], [7, 1]]
>>>
```

Many more features

- Functions, objects, multithreading, error handling
- Very active community
- A lot of great open-source libraries
- NumPy -- scientific computing with Python (http://numpy.scipy.org/)
- Matplotlib -- a python 2D plotting library (http://matplotlib.sourceforge.net/)
- CVXOPT -- convex optimization (http://abel.ee.ucla.edu/cvxopt/)


rls.py - C:\Documents and Settings\Mustafa Sir\Desktop\CVX\rls.py
File Edit Format Run Options Windows Help
Ln: 1 Col: 0

```
# Regularized least-squares.

import pylab
from pickle import load
from cvxopt import blas, lapack, matrix
from cvxopt import solvers
solvers.options['show_progress'] = 0

data = load(open("rls.bin",'r'))
A, b = data['A'], data['b']
m, n = A.size

# LS solution
xls = +b
lapack.gels(+A, xls)
xls = xls[:n]

# We compute the optimal values of
#
#     minimize/maximize  || A*x - b ||_2^2
#     subject to          x'*x = alpha
#
# via the duals.
#
# Lower bound:
#
#     maximize    -t - u*alpha
#     subject to  [u*I, 0; 0, t] + [A, b]'*[A, b] >= 0
#
# Upper bound:
#
#     minimize    t + u*alpha
#     subject to  [u*I, 0; 0, t] - [A, b]'*[A, b] >= 0.
#
# Two variables (t, u).

G = matrix(0.0, ((n+1)**2, 2))
G[-1, 0] = -1.0    # coefficient of t
G[: (n+1)**2-1 : n+2, 1] = -1.0    # coefficient of u
h = matrix( [ [ A.T * A,  b.T * A ], [ A.T * b, b.T * b ] ] )
```



Recommended Tutorial

A Quick, Painless Tutorial on the Python Language

Norman Matloff
University of California, Davis

http://heather.cs.ucdavis.edu/~matloff/Python/PythonIntro.pdf

Download Python

Python Programming Language -- Official Website

http://python.org/

Gurobi Solver

Gurobi
Optimization

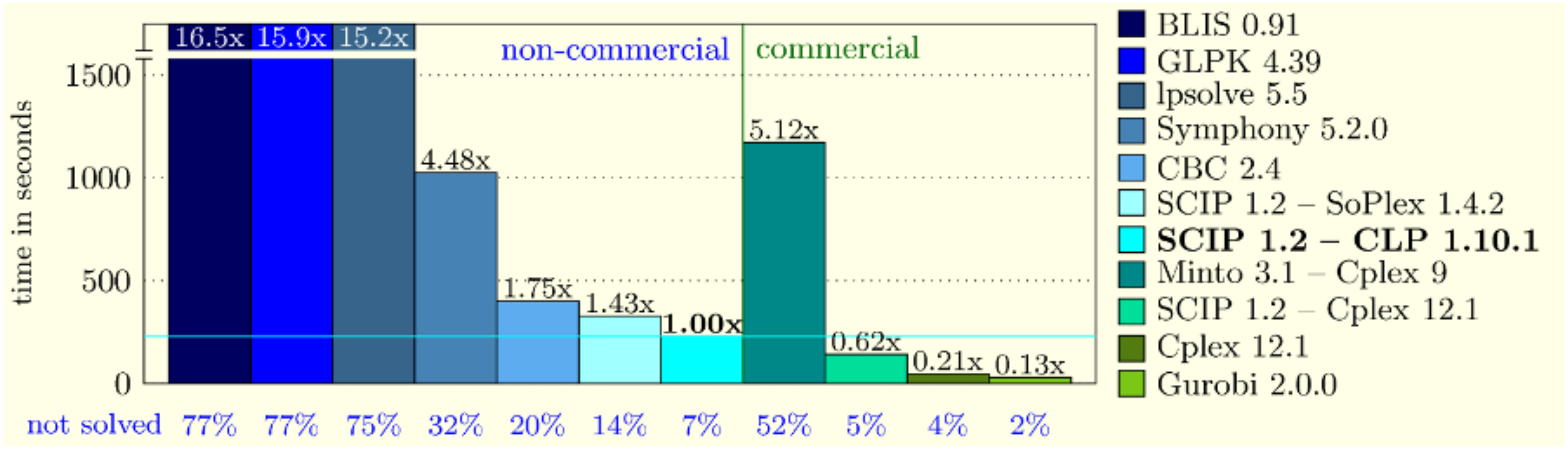
Free academic licenses to all faculty and students affiliated with a higher ed institution!

http://www.gurobi.com/

What is Gurobi?

- Gurobi is a new state-of-the-art simplex based linear programming (LP) and mixed-integer programming (MIP) solver
- Built from the ground up to exploit modern multi-core processors.
- It is developed by Zonghao **Gu**, Edward **Ro**thberg, and Robert **Bi**xby, who were the brains behind CPLEX.
- Its performance, in my limited experience, is better than CPLEX.

Gurobi Solver



Source: http://scip.zib.de/

Interactive Shell Based on Python

```
> gurobi.bat (or gurobi.sh for Linux or Mac OS)

Gurobi Interactive Shell, Version 2.0.0
Copyright (c) 2009, Gurobi Optimization, Inc.
Type "help()" for help

gurobi> m = read('c:/gurobi200/win32/examples/data/p0033.mps')
Read MPS format model from file c:/gurobi200/win32/examples/
data/p0033.mps
P0033: 16 Rows, 33 Columns, 98 NonZeros.
gurobi> m.optimize()
Optimize a model with 16 Rows, 33 Columns and 98 NonZeros
```

Interactive Shell Based on Python

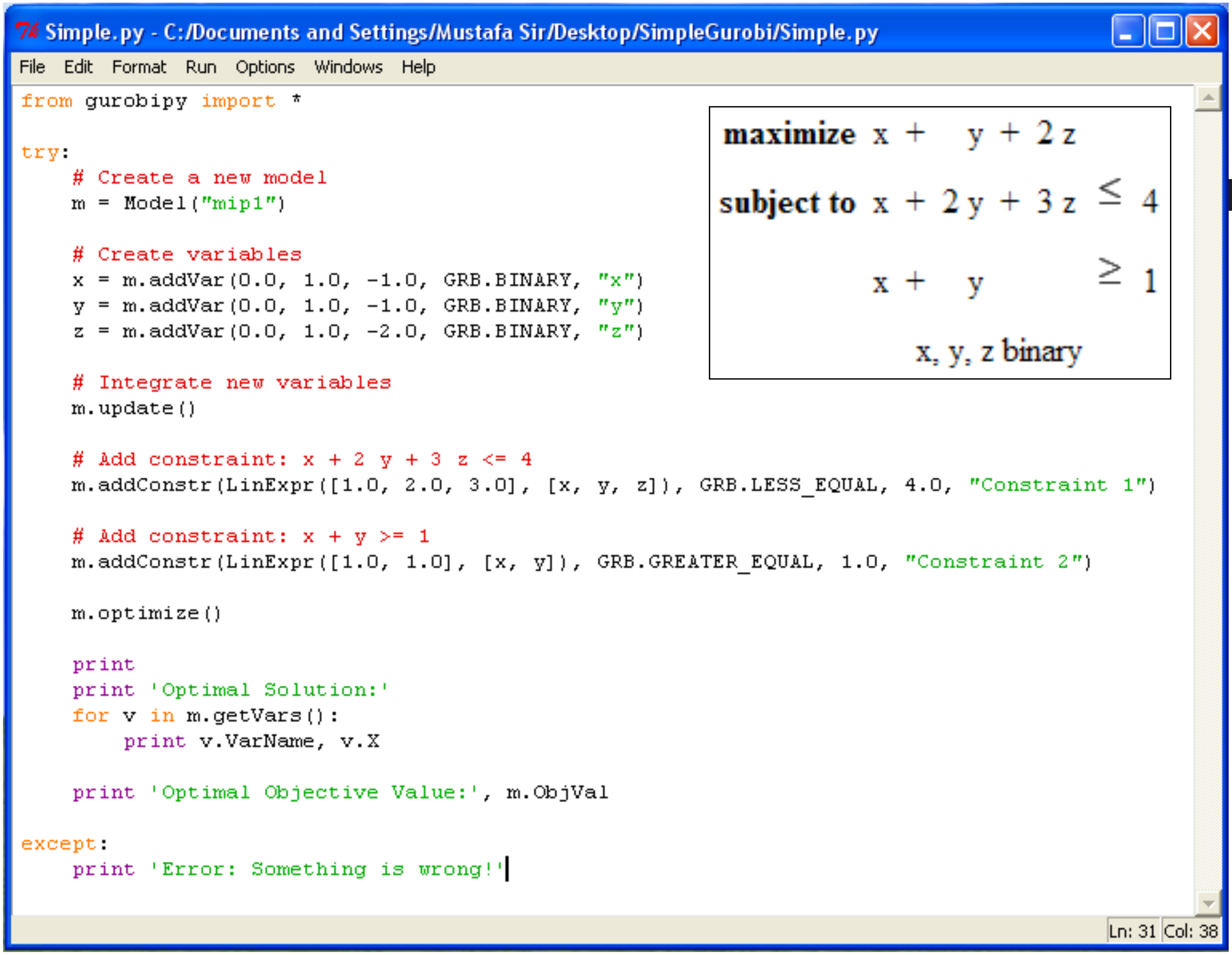
```
Explored 34 nodes (116 simplex iterations) in 0.01 seconds
Thread count was 4 (of 4 available processors)

Optimal solution found (tolerance 1.00e-04)
Best objective 3.0890000000e+03, best bound 3.0890000000e+03,
gap 0.0000%
```

Interactive Shell Based on Python

```
Explored 34 nodes (116 simplex iterations) in 0.01 seconds
Thread count was 4 (of 4 available processors)

Optimal solution found (tolerance 1.00e-04)
Best objective 3.0890000000e+03, best bound 3.0890000000e+03,
gap 0.0000%

gurobi> v = m.getVars()
gurobi> print len(v)
33

gurobi> print v[0].VarName, v[0].X
C157 1.0
```

Gurobi Python Interface

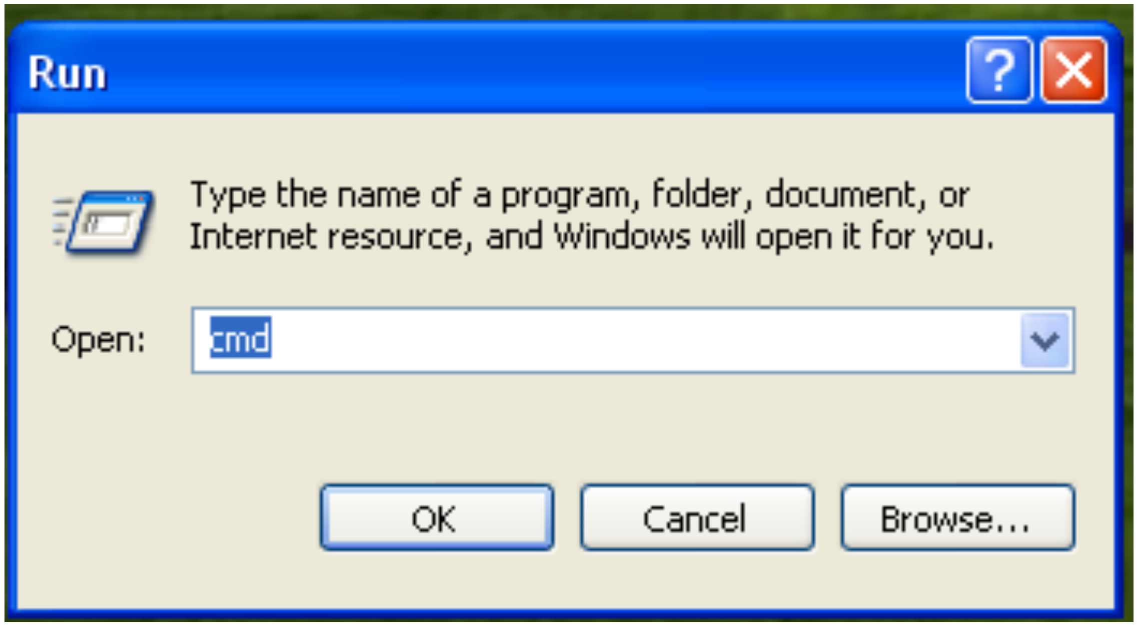
$$\begin{aligned} \textbf{maximize} \quad & x + y + 2z \\ \textbf{subject to} \quad & x + 2y + 3z \leq 4 \\ & x + y \geq 1 \\ & x, y, z \text{ binary} \end{aligned}$$


Simple.py - C:/Documents and Settings/Mustafa Sir/Desktop/SimpleGurobi/Simple.py
File
Edit
Format
Run
Options
Windows
Help
Ln: 31
Col: 38

```python
from gurobipy import *

try:
    # Create a new model
    m = Model("mip1")

    # Create variables
    x = m.addVar(0.0, 1.0, -1.0, GRB.BINARY, "x")
    y = m.addVar(0.0, 1.0, -1.0, GRB.BINARY, "y")
    z = m.addVar(0.0, 1.0, -2.0, GRB.BINARY, "z")

    # Integrate new variables
    m.update()

    # Add constraint: x + 2 y + 3 z <= 4
    m.addConstr(LinExpr([1.0, 2.0, 3.0], [x, y, z]), GRB.LESS_EQUAL, 4.0, "Constraint 1")

    # Add constraint: x + y >= 1
    m.addConstr(LinExpr([1.0, 1.0], [x, y]), GRB.GREATER_EQUAL, 1.0, "Constraint 2")

    m.optimize()

    print
    print 'Optimal Solution:'
    for v in m.getVars():
        print v.VarName, v.X

    print 'Optimal Objective Value:', m.ObjVal

except:
    print 'Error: Something is wrong!'
```

$$\begin{array}{lrcl} \textbf{maximize} & x + y + 2z & & \\ \textbf{subject to} & x + 2y + 3z & \leq & 4 \\ & x + y & \geq & 1 \\ & x, y, z \text{ binary} & & \end{array}$$

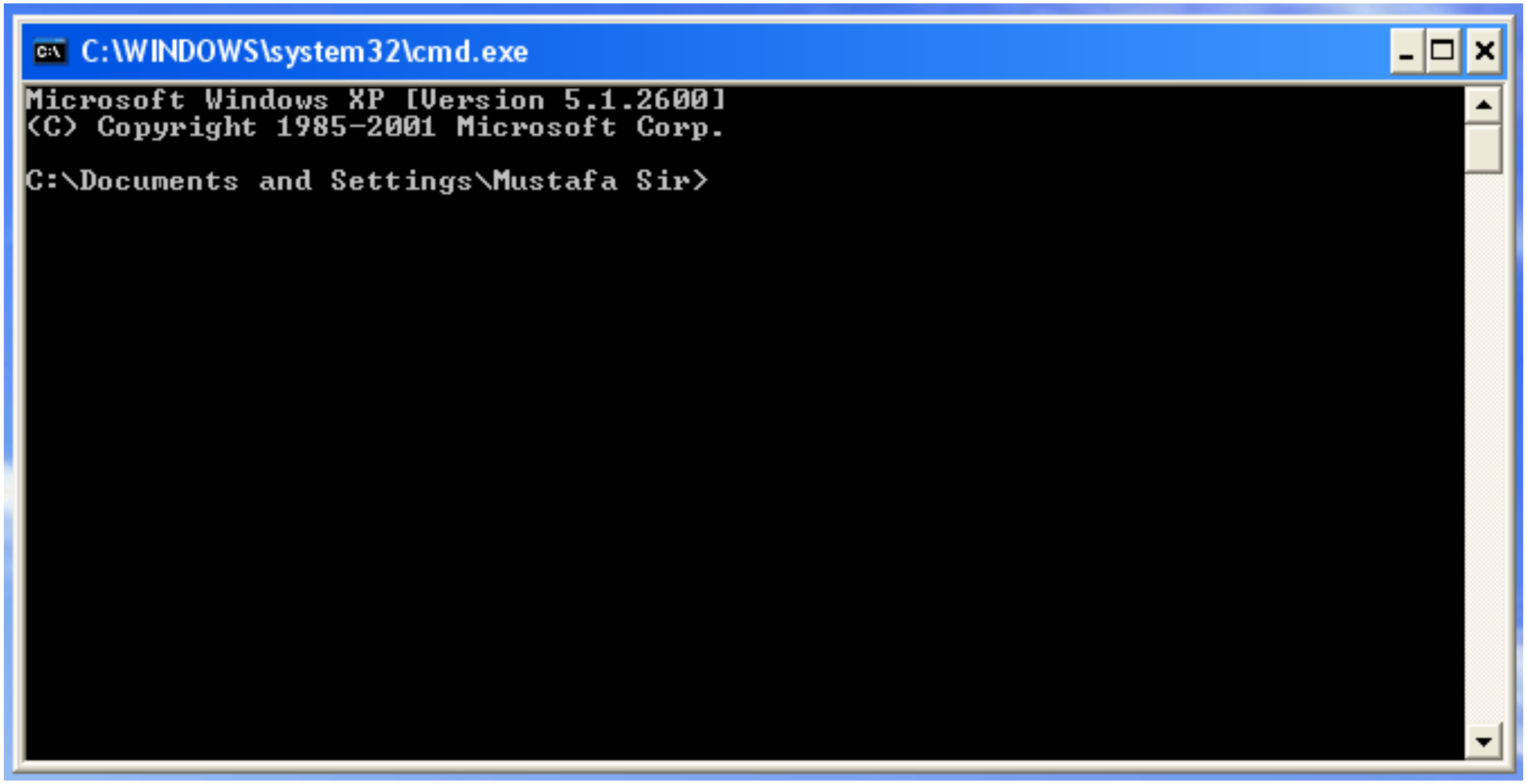
How to run a Python script using Gurobi?



Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

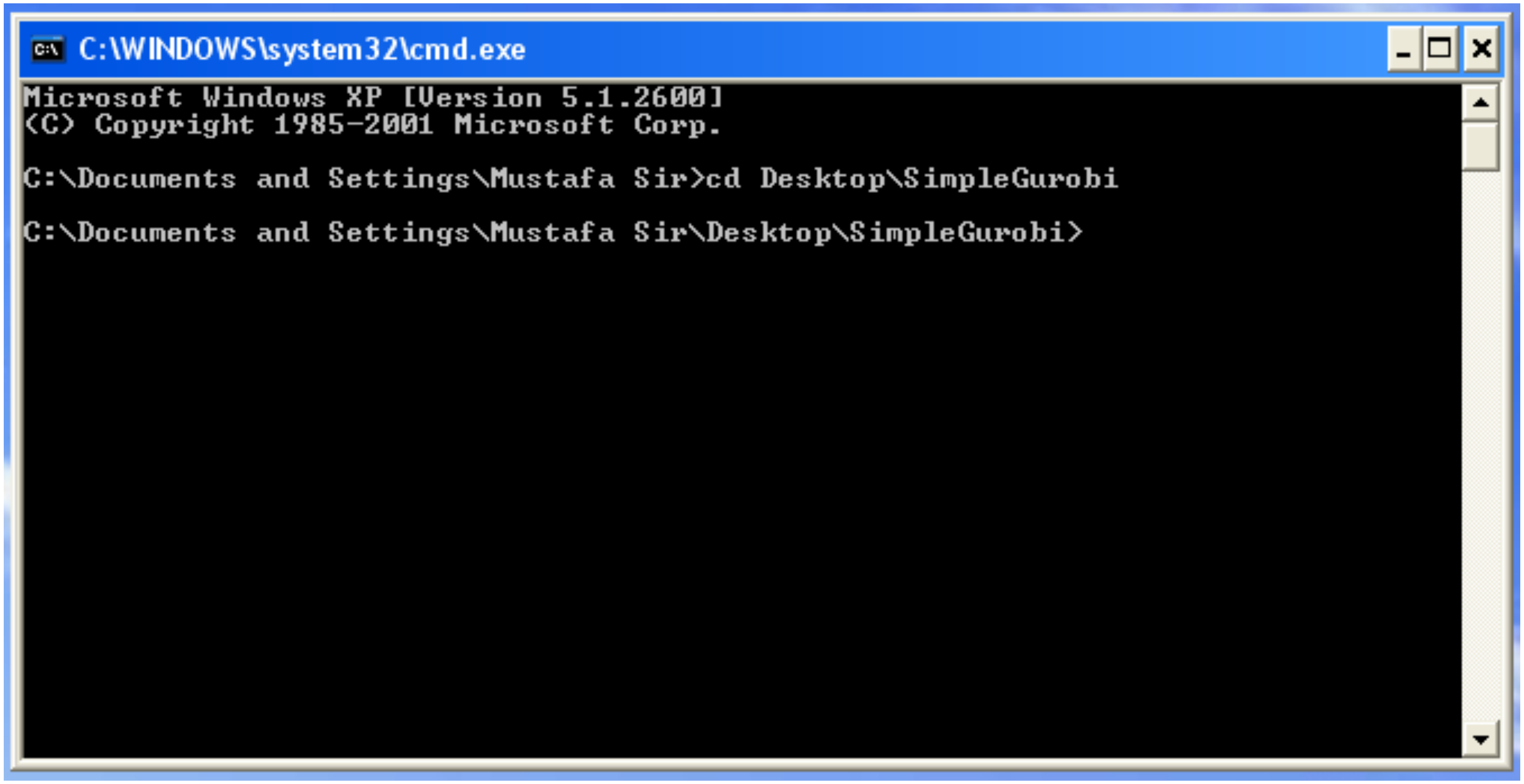
Open: cmd

How to run a Python script using Gurobi?



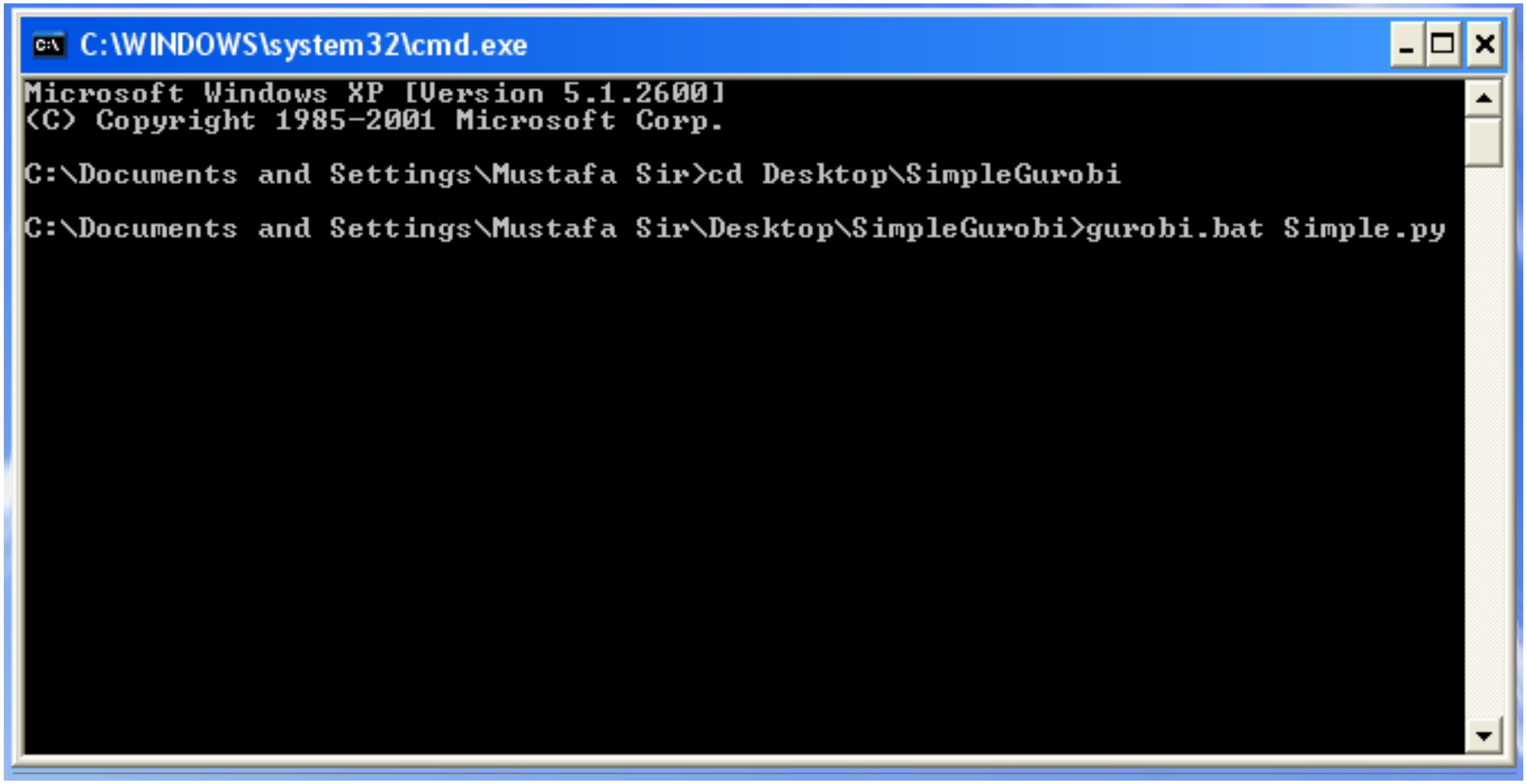
```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Mustafa Sir>
```

How to run a Python script using Gurobi?



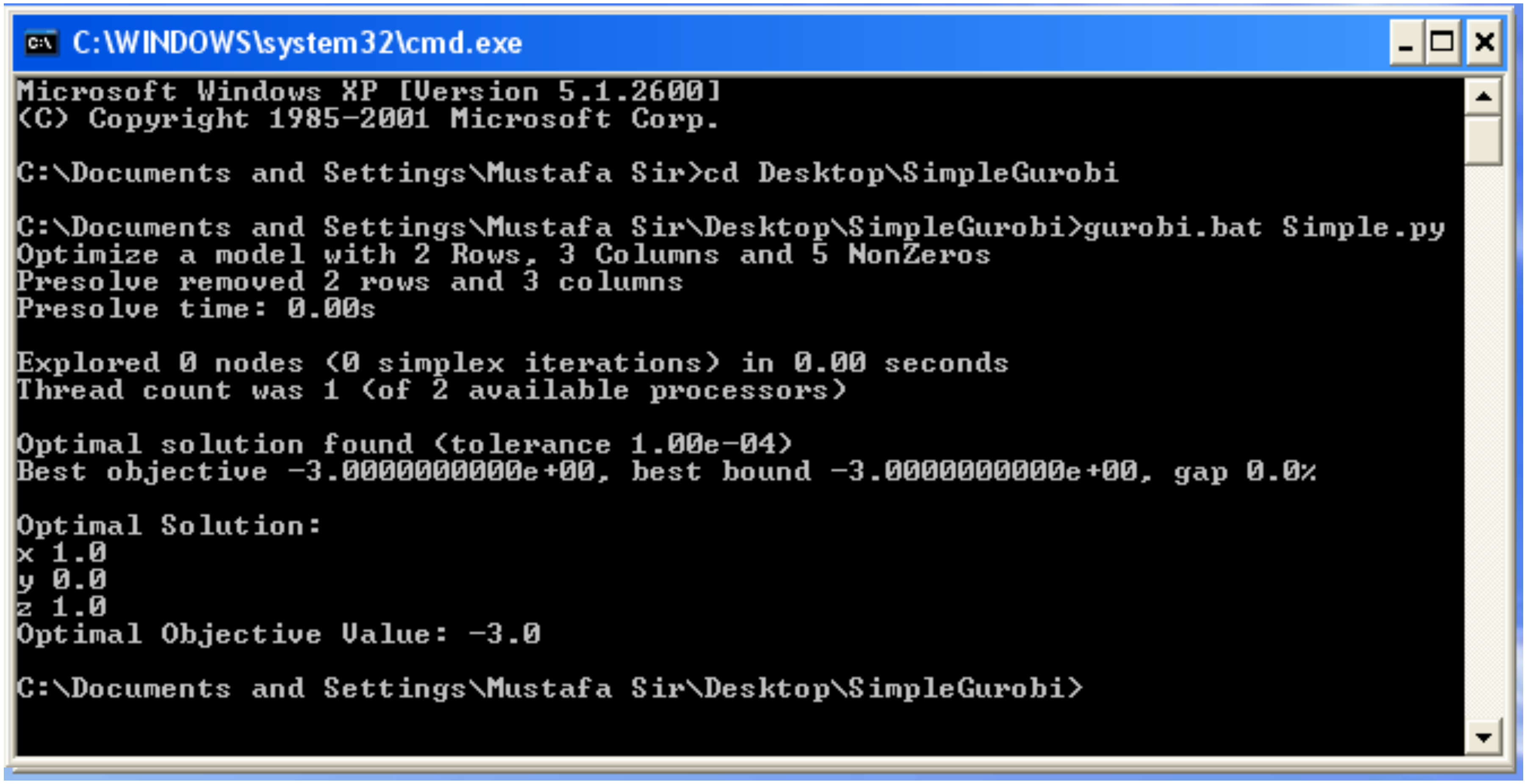
```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Mustafa Sir>cd Desktop\SimpleGurobi

C:\Documents and Settings\Mustafa Sir\Desktop\SimpleGurobi>
```

How to run a Python script using Gurobi?



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Mustafa Sir>cd Desktop\SimpleGurobi

C:\Documents and Settings\Mustafa Sir\Desktop\SimpleGurobi>gurobi.bat Simple.py
```

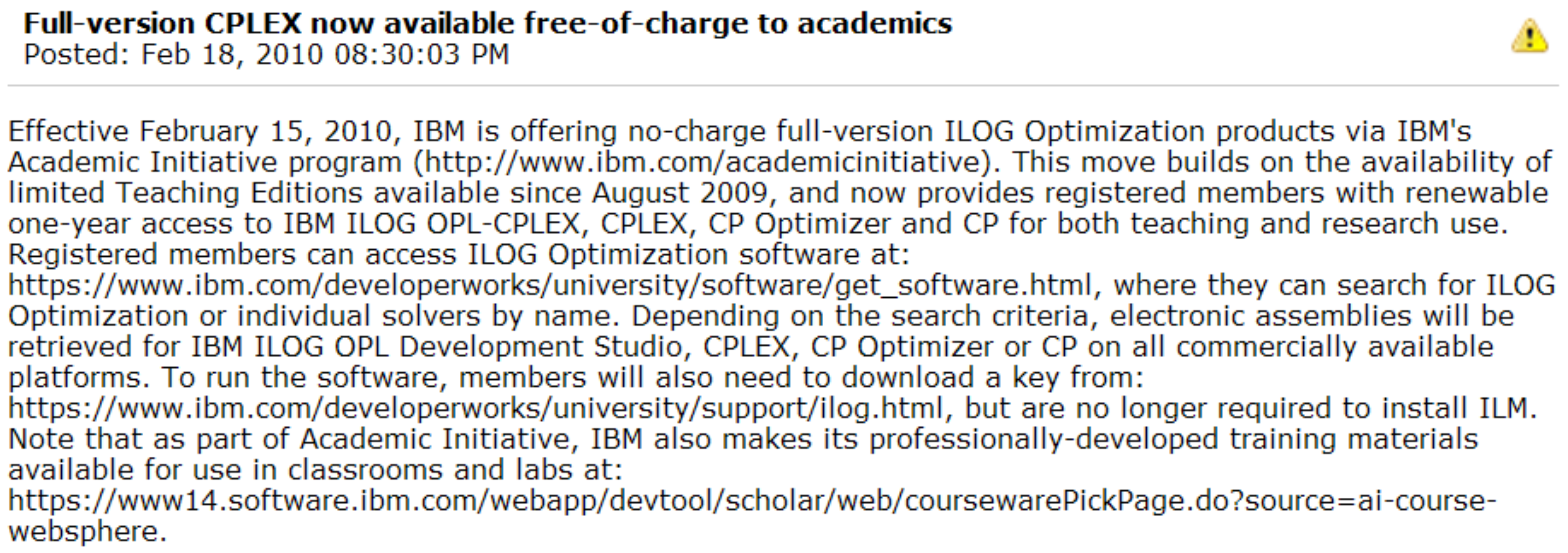
How to run a Python script using Gurobi?



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Mustafa Sir>cd Desktop\SimpleGurobi

C:\Documents and Settings\Mustafa Sir\Desktop\SimpleGurobi>gurobi.bat Simple.py
Optimize a model with 2 Rows, 3 Columns and 5 NonZeros
Presolve removed 2 rows and 3 columns
Presolve time: 0.00s

Explored 0 nodes (0 simplex iterations) in 0.00 seconds
Thread count was 1 (of 2 available processors)

Optimal solution found (tolerance 1.00e-04)
Best objective -3.0000000000e+00, best bound -3.0000000000e+00, gap 0.0%

Optimal Solution:
x 1.0
y 0.0
z 1.0
Optimal Objective Value: -3.0

C:\Documents and Settings\Mustafa Sir\Desktop\SimpleGurobi>
```

What is CPLEX doing to hold its place?

http://www.ibm.com/developerworks/forums/thread.jspa?threadID=319342&tstart=0



Full-version CPLEX now available free-of-charge to academics
Posted: Feb 18, 2010 08:30:03 PM

Effective February 15, 2010, IBM is offering no-charge full-version ILOG Optimization products via IBM's Academic Initiative program (http://www.ibm.com/academicinitiative). This move builds on the availability of limited Teaching Editions available since August 2009, and now provides registered members with renewable one-year access to IBM ILOG OPL-CPLEX, CPLEX, CP Optimizer and CP for both teaching and research use. Registered members can access ILOG Optimization software at: https://www.ibm.com/developerworks/university/software/get_software.html, where they can search for ILOG Optimization or individual solvers by name. Depending on the search criteria, electronic assemblies will be retrieved for IBM ILOG OPL Development Studio, CPLEX, CP Optimizer or CP on all commercially available platforms. To run the software, members will also need to download a key from: https://www.ibm.com/developerworks/university/support/ilog.html, but are no longer required to install ILM. Note that as part of Academic Initiative, IBM also makes its professionally-developed training materials available for use in classrooms and labs at: https://www14.software.ibm.com/webapp/devtool/scholar/web/coursewarePickPage.do?source=ai-course-websphere.

QUESTIONS?