The Gurobi Python Module

GUROBI
OPTIMIZATION

ANACONDA®

Inside the Gurobi Optimizer

- Algorithms for continuous optimization (LP, QP, QCP)
  - Simplex, Barrier
- Algorithms for discrete optimization (MIP, MIQP, MIQCP)
  - Parallel branch-and-bound
- Other algorithms
  - Presolve, parameter tuning, irreducible inconsistent subsystems…
- Command-line interface, full-featured interactive shell
- Programming interfaces
  - C, C++, Java, .NET, Python, MATLAB, R

CONTINUUM ANALYTICS

GUROBI OPTIMIZATION

Gurobi Model Components

- Decision variables
- Objective function
  - minimize $\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \mathbf{c}^{\mathrm{T}}\mathbf{x} + \alpha$
- Constraints
  - $\mathbf{A}\mathbf{x} = \mathbf{b}$ (linear constraints)
  - $\mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$ (bound constraints)
  - some x_h integral (integrality constraints)
  - some x_i lie within second order cones (cone constraints)
  - $\mathbf{x}^{\mathrm{T}}\mathbf{Q}_j\mathbf{x} + \mathbf{q}_j^{\mathrm{T}}\mathbf{x} \leq \beta_j$ (quadratic constraints)
  - some x_k in SOS (special ordered set constraints)

Example: Mixed Integer Program

- Decision variables
- Objective function
  - minimize $\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{c}^T\mathbf{x} + \alpha$
- Constraints
  - $\mathbf{A}\mathbf{x} = \mathbf{b}$ (linear constraints)
  - $\mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$ (bound constraints)
  - some x_h integral (integrality constraints)
  - some x_i lie within second order cones (cone constraints)
  - $\mathbf{x}^T\mathbf{Q}_j\mathbf{x} + \mathbf{q}_j^T\mathbf{x} \leq \beta_j$ (quadratic constraints)
  - some x_k in SOS (special ordered set constraints)

Benchmarks – Open Source MIP

- Mittelmann MIPLIB2010 tests, P=1 (>1X means Gurobi wins):
  - http://plato.la.asu.edu/bench.html

	January 2012	January 2013	January 2015	November 2015	% Solved
CBC	10X	13X	17X	26X	55%
SCIP	6X	7X	7X	9X	80%
GLPK	22X	27X	-	-	1%
LPSOLVE	19X	24X	-	-	6%

- GLPK and LPSOLVE are not currently tested in the public benchmarks

conda package management

Install	`conda install pandas numpy=1.9 gurobi`
Update	`conda update gurobi`
List	`conda list`
Search	`conda search gurobi`
Remove	`conda remove matplotlib`

CONTINUUM® ANALYTICS

GUROBI OPTIMIZATION

conda environment management

Create	`conda create -n lp_project anaconda gurobi`	

Select	`source activate lp_project`
	`activate lp_project`
		(Linux/Mac)
		(Windows)

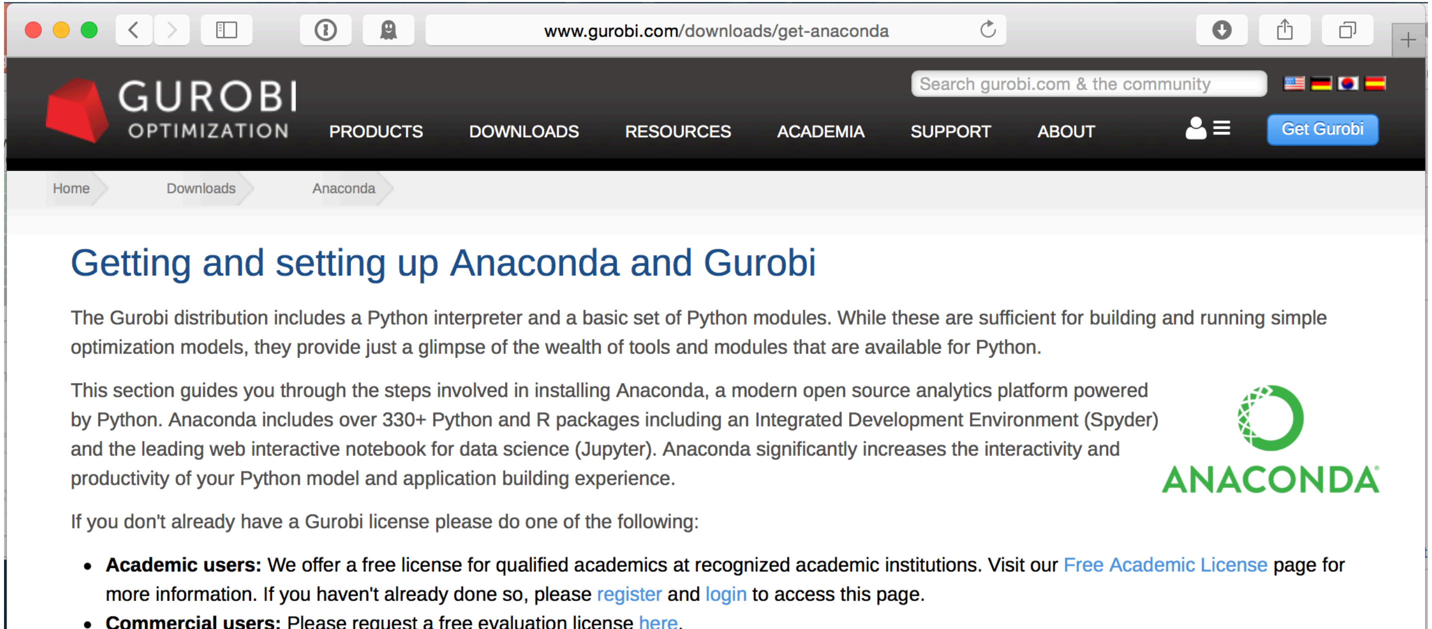
Return to default	`source deactivate`
	`deactivate`
		(Linux/Mac)
		(Windows)

Remove	`conda remove -n lp_project --all`	

Obtaining Anaconda

From Continuum: https://www.continuum.io/downloads

From Gurobi: http://www.gurobi.com/downloads/get-anaconda



Getting and setting up Anaconda and Gurobi

The Gurobi distribution includes a Python interpreter and a basic set of Python modules. While these are sufficient for building and running simple optimization models, they provide just a glimpse of the wealth of tools and modules that are available for Python.

This section guides you through the steps involved in installing Anaconda, a modern open source analytics platform powered by Python. Anaconda includes over 330+ Python and R packages including an Integrated Development Environment (Spyder) and the leading web interactive notebook for data science (Jupyter). Anaconda significantly increases the interactivity and productivity of your Python model and application building experience.

If you don't already have a Gurobi license please do one of the following:

- **Academic users:** We offer a free license for qualified academics at recognized academic institutions. Visit our Free Academic License page for more information. If you haven't already done so, please register and login to access this page.
- **Commercial users:** Please request a free evaluation license here.

Building Models with Gurobi

- Option 1: use a modeling language
  - Very easy to build models
    - Optimization modeling constructs built into language
  - Attractive choice for non-programmers
- Option 2: use a full programming language
  - Much more powerful and flexible development environment
    - Complete access to solver functionality
    - Richer set of language features
  - Natural choice when deploying and/or integrating with applications

Gurobi Python Environment

- High-level optimization modeling constructs embedded in Python
- Design goals:
  - Require minimal programming skills to get started
  - Bring "feel" of a modeling language to the Python interface
  - Allow for code that is easy to write and maintain
  - Maintain unified design across all of our interfaces
  - Remain lightweight and efficient compared to solver alone
  - Support all solver and programming needs

Essential Gurobi Python Constructs

- Objects represent model components (`Model`, `Var`, `Constr`)
- Overloaded operators
  - Arithmetic (+, -, ×, ÷), constraints (≤, =, ≥)
- Aggregate sum operator (`quicksum`)
- Python provides the rest for representing data, indices and subscripts
  - Lists, tuples, dictionaries, loops, generator expressions, …

Ex: $x_i + y_i \le 5, \forall i \in I$ ⇔

```
for i in I:
    m.addConstr(x[i] + y[i] <= 5)
```

Building a Gurobi Model: Six Steps

Create a model	`m = Model()`
Add variables	`x = m.addVar(...)`
Commit changes	`m.update()`
Set objective	`m.setObjective(...)`
Add constraints	`m.addConstr(...)`
Optimize	`m.optimize()`

Installing the Gurobi Python Module

Add the channel	`conda config --add channels gurobi`
Install Gurobi	`conda install gurobi`
Retrieve license	`grbgetkey xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx`
Test license	`>>> import gurobipy`
Test token server	`gurobi_cl --tokens`

Portfolio Optimization in Jupyter

GUROBI
OPTIMIZATION

ANACONDA®

Modern Portfolio Theory In One Slide

- First offered by Harry Markowitz in 1952
- Recognized by a Nobel Memorial Prize in Economics
- Associates with each stock an *expected return* and a risk or *volatility* measure, and a matrix of *correlations* between pairs of stocks
- The goal is to study the tradeoff between *risk* and *return*
  - Minimize risk for a given target (expected) return
  - Maximize return given an upper bound on risk
- The most basic models are *quadratic programs* (QPs)

(okay, two slides)

$$\begin{array}{ll} \text{minimize} & x^T \Sigma x \\ \text{subject to} & \vec{1}^T x = 1 \\ & \rho^T x = \gamma \\ & x \geq 0 \end{array}$$

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^n \sum_{j=1}^n \sigma_{ij} x_i x_j \\ \text{subject to} & \sum_{i=1}^n x_i = 1 \\ & \sum_{i=1}^n \rho_i x_i = \gamma \\ & x_i \geq 0 \quad i = 1, 2, \ldots, n \end{array}$$

x_i	amount of investment allocated to stock i
ρ_i	expected return for stock i
σ_{ii}	variance of the performance for stock i
$\sigma_{ij}/\sqrt{\sigma_{ii}\sigma_{jj}}$	correlation between stocks i and j

Jupyter (formerly IPython)

- A web application for interactive data analysis
- Create notebooks containing live code, text, equations, and visualizations
- A great way to prototype new algorithms, document experiments, share reproducible results
- Originally for Python, now there are kernels for over 50 different languages

http://www.jupyter.org

Pandas

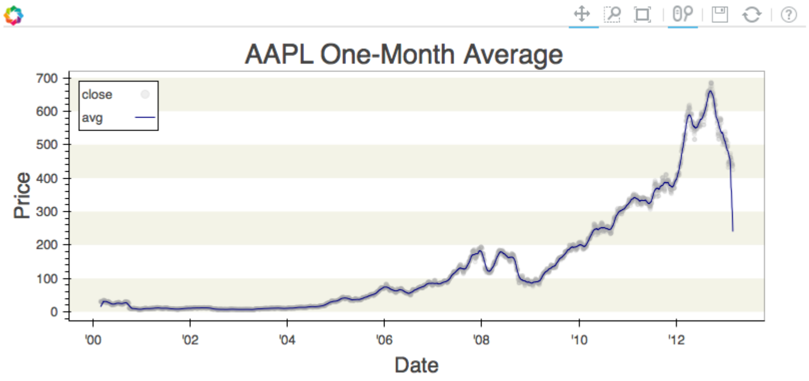
- Fast and efficient DataFrame and Series objects
- Read/write CSV, HDF5, Excel, SQL, plain text
- Missing data handling
- Slicing, fancy indexing, subsetting
- Merges, joins
- Split-apply-combine operations (groupby)

http://pandas.pydata.org

Bokeh

Interactive visualization for web browsers

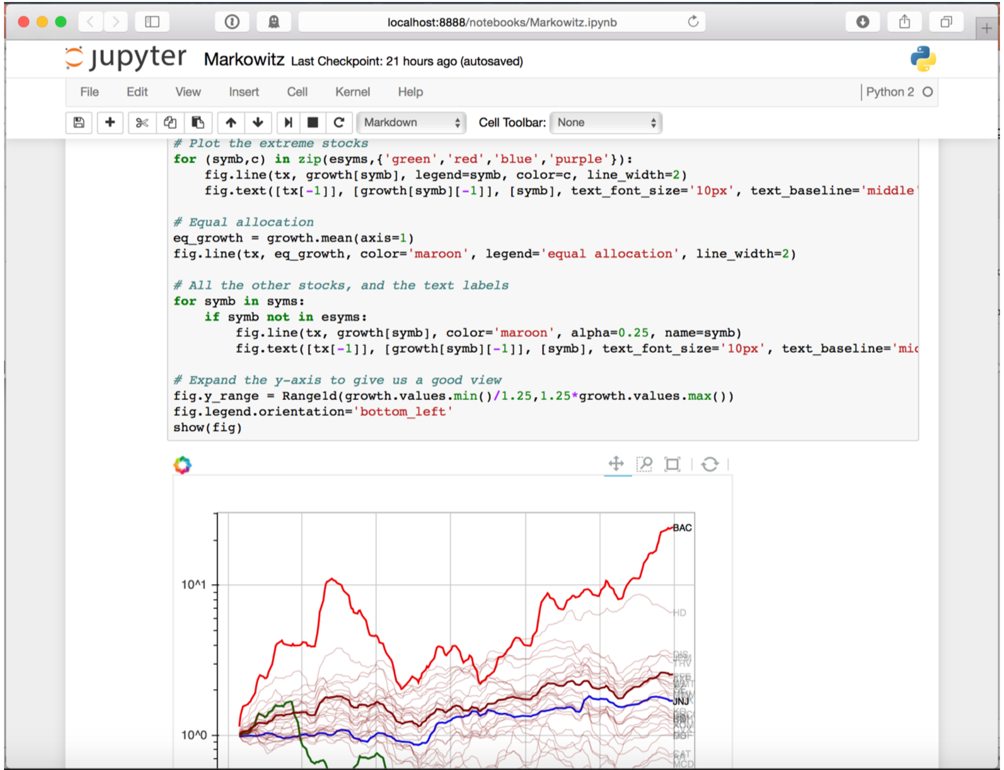
- High performance with streaming data and big data
- In-browser interactivity on otherwise *static* pages
- Bindings for Python, Scala, Julia, R



http://bokeh.pydata.org

Jupyter Demo

https://notebooks.anaconda.org/mcg/portfolio



```
# Plot the extreme stocks
for (symb,c) in zip(esyms,{'green','red','blue','purple'}):
    fig.line(tx, growth[symb], legend=symb, color=c, line_width=2)
    fig.text([tx[-1]], [growth[symb][-1]], [symb], text_font_size='10px', text_baseline='middle'

# Equal allocation
eq_growth = growth.mean(axis=1)
fig.line(tx, eq_growth, color='maroon', legend='equal allocation', line_width=2)

# All the other stocks, and the text labels
for symb in syms:
    if symb not in esyms:
        fig.line(tx, growth[symb], color='maroon', alpha=0.25, name=symb)
        fig.text([tx[-1]], [growth[symb][-1]], [symb], text_font_size='10px', text_baseline='mic

# Expand the y-axis to give us a good view
fig.y_range = Range1d(growth.values.min()/1.25,1.25*growth.values.max())
fig.legend.orientation='bottom_left'
show(fig)
```