

# Python for Computational Chemistry

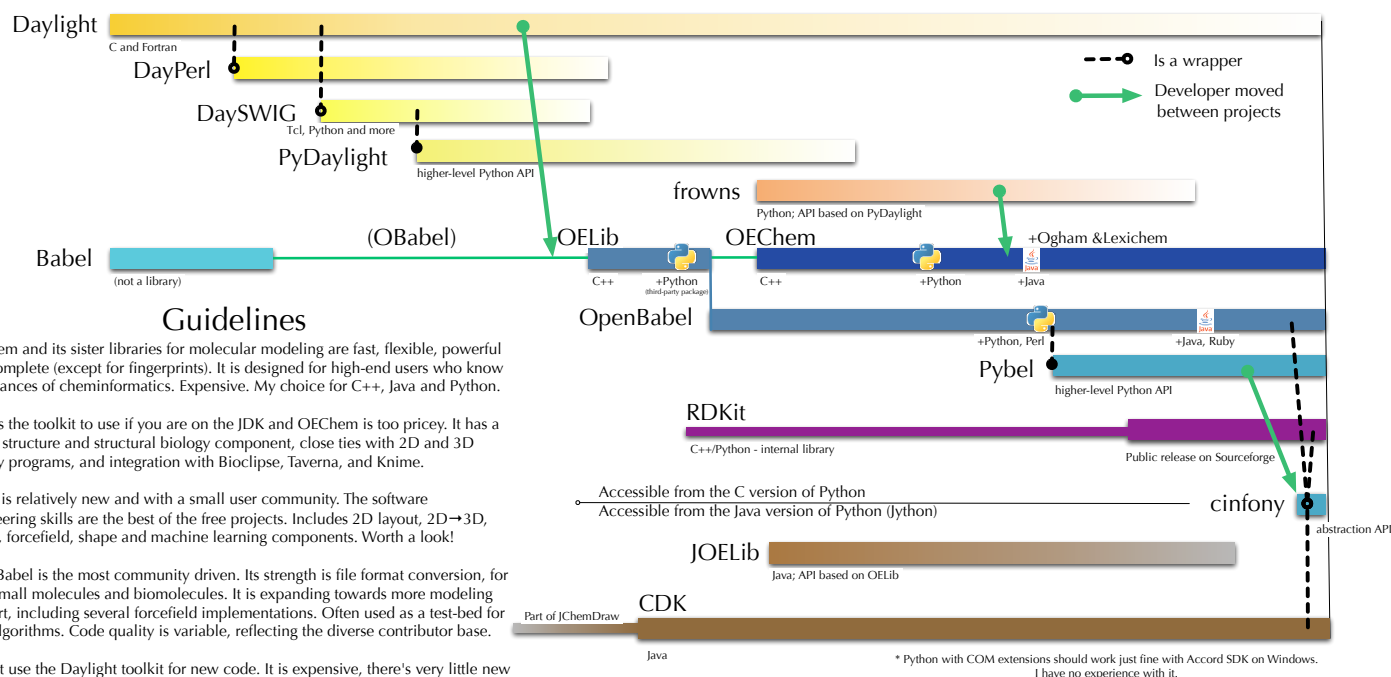
Andrew Dalke <dalke@dalkescientific.com>, Andrew Dalke Scientific AB, Göteborg, Sweden

(Wherein I describe how Python is the best choice of high-level programming language for computational chemistry.)

## Timeline of cheminformatics toolkits\*

\*runs on Unix and supports SMILES and SMARTS)

1995 and earlier 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008



## Answers to the question: "How do I do \_\_\_\_\_ in Python?"

### Plotting

Use matplotlib. It produces great plots, is easy to use, and has a big support community.

```
from pylab import *
from data_helper import get_daily_data
intc, mstf = get_daily_data()
delta1 = diff(intc.open)/intc.open[0]

# size in points **2
volume = [15*intc.volume[-2]/intc.volume[0]]**2
close = 0.003*intc.close[-2]/0.003*intc.open[-2]
scatter(delta1[-1], delta1[1:], c=close, s=volume, alpha=0.75)

ticks = arange(-0.06, 0.061, 0.02)
xticks(ticks)
yticks(ticks)

xlabel(r'$\Delta_{15}$', fontsize=20)
ylabel(r'$\Delta_{1+1}$', fontsize=20)
title('Volume and percent change')
grid(True)
show()
```

Scatterplot example from matplotlib

### Web applications

Use Django.

There are other options, like Zope and TurboGears, but if you don't already know about them then the answer is Django.

If you're doing Javascript programming, use jQuery. I also like MochiKit because it makes Javascript programming feel like Python.

### Databases

If you're developing a Django web application then use its ORM.

If you like writing SQL statements, use the DB-API library for your database of choice. (Got Oracle? Install cx\_oracle to connect to it from Python.)

If you like object interfaces, use SQLAlchemy.

Recent versions of Python include a in-process relational DBMS called SQLite. It's easy to use and it might make more sense to have a SQLite database in your project instead of a set of flat files.

### Training

I teach Python courses for cheminformatics, designed for working scientists who want to get better training in that aspect of their research. If you are interested, email me or see:

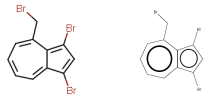
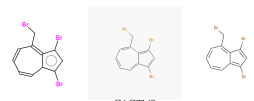
<http://dalkescientific.com/training/>

### 2D Depiction

Your chemists want ChemDraw.

But if you have to depict a structure you can use OEChem, RDKit or CDK. There's even BKChem which is a 2D editor for Python, but I've never used it. If you're only interested in depictions then also consider the command-line depicitors from Molinspiration and CACTVS.

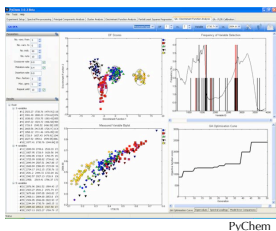
Thanks to Noel O'Boyle for the pointers and the images, from his blog article "Cheminformatics toolkit face-off - Depiction Part 2"



### GUI Programming

It seems that all my clients these days want web applications so I don't have much experience here. There's two main choices: wxPython and Qt. I like the Qt API and functionality better. I've known people who used wxPython and complained about how the APIs are always in flux, especially for the spreadsheet table.

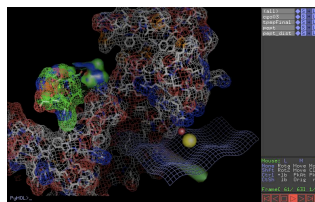
On the other hand, the PyChem developers (a multivariate analysis package, shown below) used wxPython and found it very useful.



### 3D Structure visualization

Probably PyMol.

People have diverse personal preferences about their choice of structure viewer. There are so many, even restricting myself to those programmable in Python. The best general purpose choice is PyMol, but Chimera and VMD (for trajectory visualization) are also good.



### Excel

If you're on Windows, try the win32com interface to control Excel. You can use it to open and read a spreadsheet for you, modify data in an existing spreadsheet, make charts, and more.

If you want to read an Excel "csv" file, use the "csv" module from Python's standard library.

If you're on unix and want to read an Excel file, try xlrd and then let me know. I've only read about it.

### Command-line wrappers

Use the subprocess module.

And likely a bunch of parsing, guess work, error-handling, and perhaps a dash of evil genius. But start with subprocess.

### Interfaces to compiled libraries

My favorite is ctypes, which lets Python call C libraries directly without compiling glue code. It needs some hand-written code but at least it's Python code.

SWIG is best if you have a lot of code and want to automate building the interface, or if you want bindings to multiple languages. OEChem and OpenBabel use SWIG.

Boost.Python helps make interfaces to C++ code. It's hand-written code, in C++, but it does a very good job converting between C++ and Python expectations. RDKit uses Boost.Python.

### MD and QM

In Python? Don't.

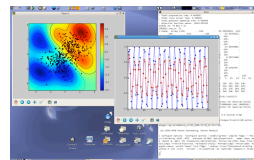
That's the realm of FORTRAN and C/C++. You'll not find much Python there. There's nMOLDYN, BALLView and PyQuante but you'll probably want one of the more well-known and used programs.

Matrix math, Fourier transforms, ODEs, and other deep math topics

Use numpy and scipy.

Clustering, SVM, and other machine learning. And R.

Some popular packages are Shogun, libsvm and PyCluster. Plenty are available, it's mostly a matter of finding a good quality one. But it seems the best code is in R (a programming language for doing math) and not Python. Solution? Use RPy to exchange data between Python and R.



### .Net or Silverlight

You're cutting edge, aren't you?

IronPython is an implementation of Python for .Net, and it's being developed by Microsoft. However, the chemistry toolkits haven't caught up with you. I haven't heard of any .Net libraries with cheminformatics support. Word on the street is that OpenBabel is working towards it.

You might be lucky and get IronClad to work. It's a way for IronPython code to call CPython extensions.