C++ and python

• We’ve now looked at how to do C++ and python separately
  – (Technically we also looked at them together, since numpy is written in C++, but more on that later)

• How can we put them together?

• Lots of options out there
  – We’ll use SWIG for a concrete example
  – http://www.swig.org

• We’re also going to use Jupyter for this for an additional bit of fun!
C++ and python... with JUPYTER!

- Then click through to “CompPhys/SwigExamples” and you get something like this:
Example of using C++ code in python and Jupyter with SWIG

It is also possible to use our C++ code from python and Jupyter. This involves using the SWIG package. You can download it [here](#) and then install following instructions [here](#). If you are successful, you should be able to open a new terminal and type `which swig` to obtain the path of swig.

The idea is then to use SWIG to automatically generate python-readable code from our C++/C libraries. There is a lot to learn in this regard, so we will try first with a simple example that illustrates some concepts we will need, such as using STL libraries and C++11 compilation.

**Step 1 : Look at C++ files**

You should be able to see these two simple C++ files:

```
In [ ]: ! cat swig_example/example.hpp swig_example/example.cpp
```

**Step 2 : Look at SWIG interface file**

The magic of SWIG is to create wrapper C++ functions that use the "cython" interface. You will see an "interface" file for SWIG:

```
In [ ]: ! cat swig_example/example.i
```