An Open-Source MATLAB®-to-Python® Compiler

(One MATLAB® per Child)

by Peter Jurica, 2008 at *Perceptual Dynamics Lab., RIKEN Brain Science Institute*

Download | **Documentation** | **Developers** | **License** | **Timeline** | **Request for Features**

[⇩][?][#][©][◷][☞]

Try the online OMPC translator | Look at working examples of OMPC

NEWS

- ***Jun 14th, 2010:*** The import hooks do not work in Python version >= 2.6, I am looking into that. The examples still work.
- ***Jun 4th, 2010:*** A small change to get "import ompc" work in IPython 0.10.
- ***Jun 16th, 2009:*** I am working on OMPC one two days a week now, this will improve with time, at the moment I am playing with the future GUI and the multiprocessing library.
- ***Jun 16th, 2009:*** As with many projects my deadlines are moving, I need some time to publish my research.
- ***Jun 16th, 2009:*** I decided to try out Twitter (https://twitter.com/ompc)
- ***Feb 11th, 2009:*** A paper explaining the why and how of OMPC appeared in February 2009 in the journal Frontiers in Neuroinformatics. Peter Jurica, Cees van Leeuwen: ***OMPC: an Open-source MATLAB®-to-Python Compiler.*** The ***supplementary examples*** that are part of the article can be found here.

OMPC allows running MATLAB®'s m-files using Python interpreter. OMPC reads m-files and translates them into Python compatible code.

Download and Installation

OMPC aims to enable reuse of the huge open and free code base of MATLAB®

Navigation

An Open-Source MATLAB®-to-Python® Compiler

Request for features

Sitemap

Recent site activity

Top level
attachment from Peter Jurica

An Open-Source MATLAB®-to-Python® Compiler
edited by Peter Jurica

Examples (Supplemental data to the article OMPC: An Open-source MATLAB® -to-Python Compiler.)
edited by Peter Jurica

An Open-Source MATLAB®-to-Python® Compiler
edited by Peter Jurica

Request for features
edited by Peter Jurica

View All

on a free and faster growing Python platform. Running Python and MATLAB® in a single interpreter avoids issues with running two separate applications. Python adds general purpose programming libraries to the convenient syntax of the language of technical computing.

Dependencies: Python, OMPC uses PLY and Byteplay, but they are distributed with the source code.
Python dependencies: ctypes (for Python version < 2.5), for OMPClib: numpy, scipy, matplotlib (pylab) and possibly others
Supported operating systems: all platforms that support Python, ctypes, numpy, scipy, matplotlib(pylab)
Licensing: OMPC and all examples that are part of OMPC website are licensed under BSD style license.

How it works

OMPC allows running MATLAB®'s m-files using Python interpreter. OMPC reads m-files and translates them into Python compatible code. Just like in the following example

```
function [s,varargout] = mysize(x)
% size with varargout
nout = max(nargout,1)-1;
s = size(x);
for k=1:nout, varargout(k) = {s(k)}; end
```

```
from ompc import *

@mfunction("s", "varargout")
def mysize(x):
    """size with varargout"""
    nout = max(nargout,1) - 1
    s = size(x)
    for k in m_[1:nout]:
        varargout(k).lvalue = mcell[s(k)]
```

OMPC is not an interpreter, it lets Python do the work. This means that if Python gets faster OMPC gets faster too. OMPC translates m-files preserving the structure of the original programs as much as possible. Although OMPC comes with a library that emulates the features of MATLAB's® numerical array of there is nothing that will stop you from running the translated code the way you like it. This means that you could run the OMPC generated code on IronPython, Jython, PyPy or whatever else if you write your own numerical class.

All the magic is made possible by python decorators and introspection. Look at the examples section to see how it is possible to emulate implicit variables like nargin/nargout and other dynamic aspects of the MATLAB® engine that are not available in Python.

Developers, testers and requests for features

If you want to understand more about how OMPC works, or you want to contribute in any way to the development look at the developer documentation pages.
Contributions are welcome in any form. Anybody can contribute by testing. The development is driven by testing and demand. For library of functions, or

toolboxes if you like OMPC project does not use any sophisticated and possibly complicated bug-tracking tool. The reason for not using a common bug-tracking is that the development relies on contributions of non-programmers.