Exam Assignments 11

What is Cython?

Cython is a compiled superset of python, used to make python code run faster and let it interface with C or C++ libraries. A cython module compiles to an extension module that can be called from normal python code.

Accelerating Python Programs with Cython

Compiling a pure python program with cython might already speed it up, but a significant speed up can be expected when type annotations are added to the program, since they allow for more automatic optimizations.

Compiling a `.pyx` Cython Module

First Way

A cython module can be compiled into a native extension module using the `cythonize` command. It first compiles to C/C++ and then to the extension module.

Second Way

One can add `import pyximport` and `pyximport.install()` to the top of the python file using the cython module. This lets the developer import the cython module as if it was a normal python module. It is automatically compiled when imported.

Compiler Directives in Cython

`boundscheck`

If set to false, Cython will not check that an indexing operation stays within its bounds. This allows for some performance gains but also takes away a bit of safety from the code. Default is true.

`cdivision`

If set to false, Cython will adjust the division and remainder operations to match those of Python (slightly different from C) and will also check for division by zero, this leads to a code

slowdown. If set to true (as is default), no such checks are performed.

`def`, `cdef` and `cpdef` for Functions

Functions declared with `def` are standard python functions that can be called from both Cython and Cython. Data types can only be specified for the arguments, not for the return value.

Functions declared with `cdef` can only be called from Cython, let the developer specify types for arguments and return value and are typically much faster than other functions.

Functions declared with `cpdef` can be called both from Cython and Python, let the developer specify types for arguments and return value and are typically faster than `def` functions, but not as fast as `cdef` functions. Internally, two versions of the function are created, one for calling from Python and one for Cython.

Typed Memoryviews and Their Usefulness

Typed memoryviews allow efficient access to memory buffers (for example those underlying numpy arrays).

They can be used to efficiently pass arrays from Python to Cython code, without any Python overhead.