Scripting at the Speed of Compiled Code

Alexander Pletzer, Wolfgang Hayek and Chris Scott
Science Coding Conference, Christchurch 2019
5 September
alexander.pletzer@nesi.org.nz

New Zealand eScience Infrastructure

Overview

- The problem
- The solution
- How well the solution works in practice

Everyone loves scripting

- Python, R, Julia, Matlab are examples of popular scripting languages used on NeSI platforms
- No need to compile (or compilation happens under the hood)
- Generally more portable than compiled code (C, C++ or Fortran)
- Faster turn around between development and deployment
- Easier to learn than C, C++ or Fortran

We'll focus here on Python…

But performance sometimes sucks

- It's possible to approach compiled code performance but you'll have to work hard
- Avoid loops in scripting languages
  - Same instruction executed many, many times
  - Each instruction needs to be parsed, interpreted, checked at runtime (slow)
  - Compiled languages shift the above overhead from run to compile time
  - Some optimisations (loop fusion, unrolling, …) are only available in C/C++, Fortran

Example: add elements of array in Python

```
import numpy
n = 100000000 # 100 million
a = numpy.arange(0, n)
s = 0
for i in range(n):
    s += a[i]
print('sum is {}'.format(s))
```

real 0m21.589s

1x

Solution 1: Use functools.reduce

```
import numpy, functools, operator
n = 100000000
a = numpy.arange(0, n)
s = functools.reduce(operator.add, a)
print('sum is {}'.format(s))
```

real 0m10.180s
2x faster

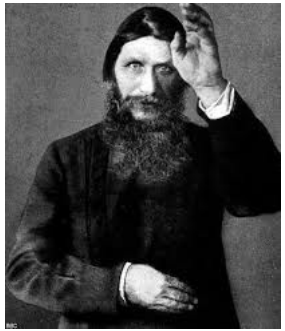
Solution 2: Use numpy.sum

```
import numpy, functools, operator
n = 100000000
a = numpy.arange(0, n)
s = numpy.sum(a)
print('sum is {}'.format(s))
```

real 0m0.576s

20x

Two words of wisdom



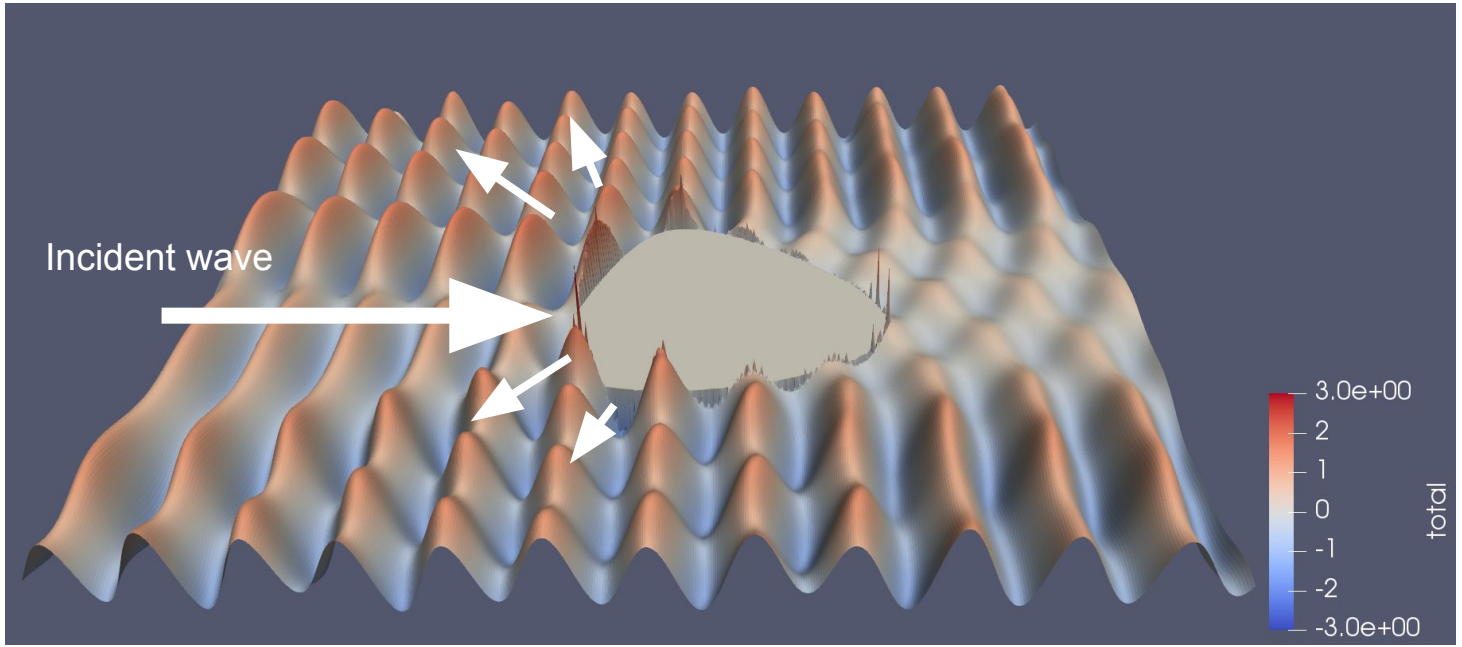
- You don’t need to know C/C++ or Fortran to accelerate your code
- But it helps if you know numpy well

If numpy vectorization is not enough then consider:

- numba
  - Add decorator to Python code then C code will be generated automatically
- Cython
  - Write code in a Python-like dialect
- Writing a C extension
  - Expose C code to Python via ctypes, SWIG, BoostPython, ….

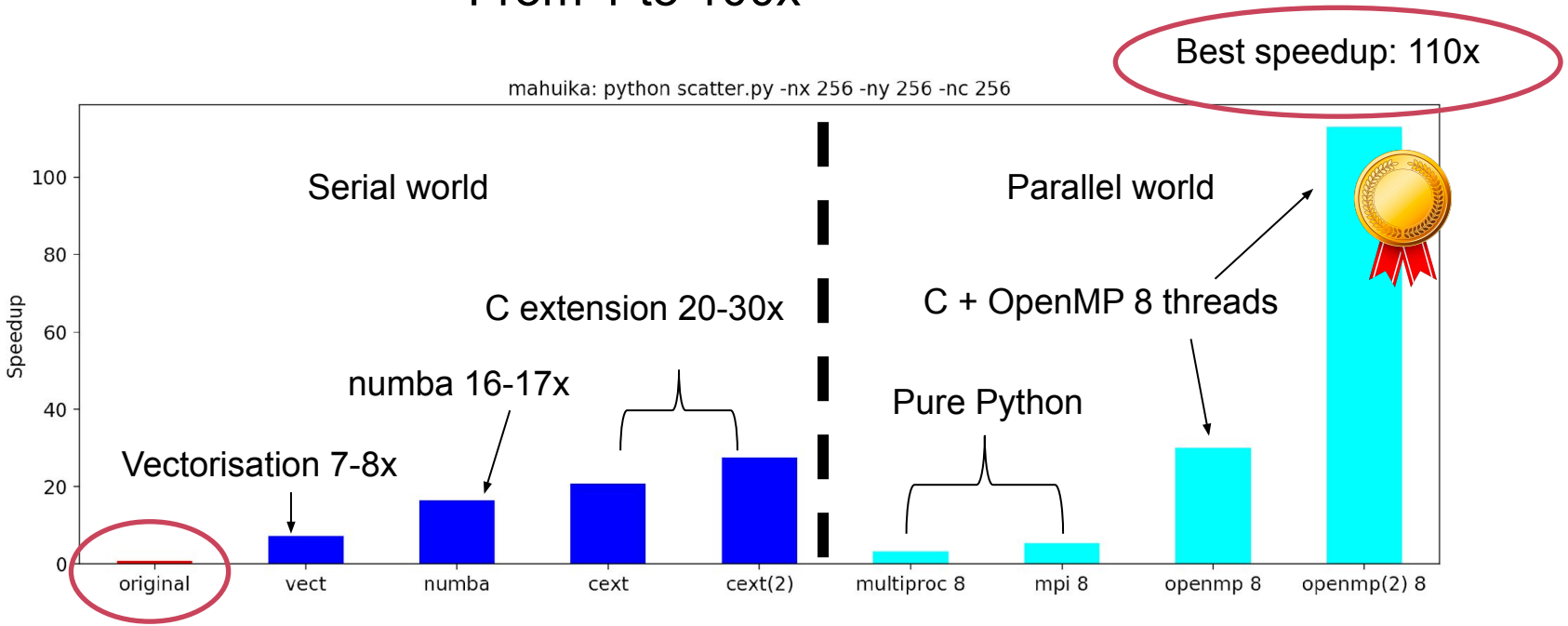
Case study: scattering of waves from an object

https://nesi.github.io/perf-training/python-scatter



Getting more bang for your buck

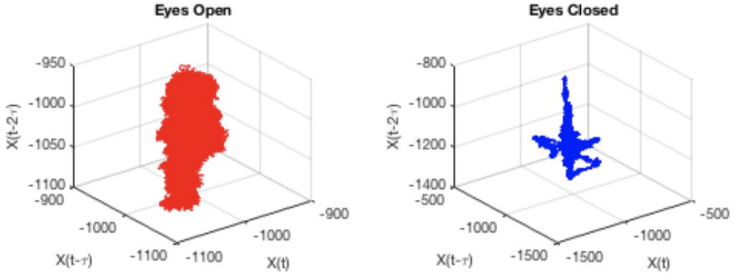
From 1 to 100x



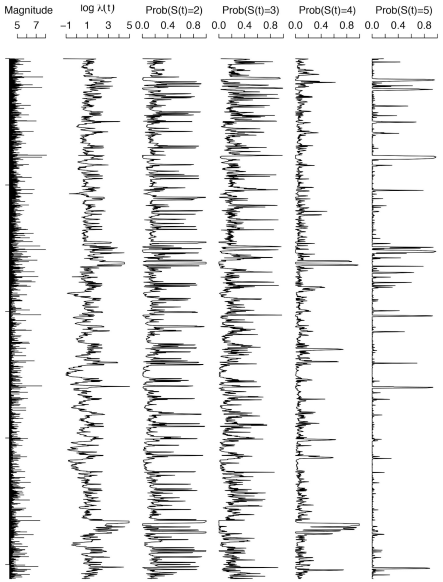
Summary

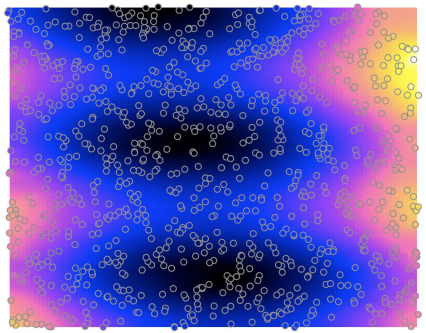
- Some projects known to have benefited from the above

Diagnosing autism from ECG signals (MATLAB): 8x with mex'ing



Hidden Markov Chains (R)





Sibson nearest neighbor interpolation (Python) 100x with refactoring + vectorisation + multiprocessing

Talk to Chris, Wolfgang or me if you need help.
More info about consultancies at
https://www.nesi.org.nz/services/consultancy

Chris Scott: Improving NeSI's researchers' productivity with a consultancy service (Fri 11:00)

Thank you.