Comparing Runtimes of Python and MATLAB for Computational Electromagnetic Problems

Alec Weiss and Atef Elsherbeni
aweiss@mines.edu, aelsherb@mines.edu
Introduction

- MATLAB and Python are two commonly used modern programming languages.
- Compare the speed of MATLAB and Python for Computational Electromagnetic (CEM) Problems.
- CEM problems commonly deal with complex numbers:
  - These are also used as both single and double precision.
  - Many times also deal with dense and sparse matrix operations.
- Multiple operations run for different sizes of problems:
  - Basic operations (e.g. addition, multiplication).
  - Extended operations (e.g. exponentiation, summation).
  - Matrix Operations (e.g. matrix factorization, linear system solving).
- Real Problems tested.
Basic Operations

Addition \((c = a + b)\)

- MATLAB Single
- Numpy Single
- Numba Single
- MATLAB Double
- Numpy Double
- Numba Double

Division \((c = a/b)\)

- MATLAB Single
- Numpy Single
- Numba Single
- MATLAB Double
- Numpy Double
- Numba Double
Extended Operations

**Combination** ($c = a + b \cdot \exp(a)$)

- **MATLAB Single**
- **Numpy Single**
- **Numba Single**
- **MATLAB Double**
- **Numpy Double**
- **Numba Double**

**Summation** ($c = \text{Sum}(a)$)

- **MATLAB Single**
- **Numpy Single**
- **Numba Single**
- **MATLAB Double**
- **Numpy Double**
Matrix Operations

Dense Linear System solving

Sparse Linear System solving

- MATLAB Single
- Numpy Single
- MATLAB Double
- Numpy Double

Runtime (s)

Number of Elements $\times 10^7$
Finite Difference Frequency Domain Example

- Simulating 2D scattering from a cylinder using the Finite Difference Frequency Domain (FDFD) method
- The main computation from an FDFD simulation is dominated by the solving of a sparse system of linear equations

![Total Field Magnitude](image)

![Runtime vs Number of Elements](image)
Angle of Arrival Example

- Angle of arrival estimation using conventional beamforming (CBF)
- The received value $V$ at each steering angle $\theta, \phi$ of an antenna array is calculated as

\[
V(\theta, \phi) = \sum_{n=1}^{N} S_n e^{-jkr}
\]

where $S_n$ is the received value at each antenna element $n$. 

![Graph showing runtime and number of angles for different computational methods]
Conclusions

- MATLAB and python have similar performance in many situations
- For sparse matrices, MATLAB will drastically outperform Python
- For many CEM applications, Python can provide a competitive free alternative to MATLAB