Analysis of Matrix-Vector Multiplication Performance

Chuyang Wang

October 12, 2023

1 Analysis of Timings

1.1 Differences Observed:

1. **Execution Time:**
    - **Matlab/Python Timings:** Typically, Matlab/Python implementations might take longer to execute because both Matlab and Python are higher-level languages compared to C. They have additional layers of abstraction, which can introduce overhead.
    - **C Timings:** C, being a lower-level language, can execute operations faster, especially with optimizations enabled.
    - **C with -O3 Flag:** The execution time is generally reduced further with the '-O3' optimization flag, making the C implementation significantly faster.
2. **Optimization:**
    - **C with -O3 Flag:** The '-O3' optimization flag enables the compiler to perform extensive optimizations, enhancing the execution speed. It can inline functions, perform loop unrolling, and apply other advanced optimization techniques, reducing the execution time.
    - **Matlab/Python:** These languages are not as amenable to the same level of optimization as C during compilation. However, built-in functions and libraries in Python and Matlab are often optimized and compiled in C or Fortran, which can sometimes offer competitive performance.

1.2 Speculations on Reasons for Differences:

1. **Interpreted vs. Compiled Languages:**

- **Matlab/Python:** They are interpreted languages. Although some JIT compilation occurs behind the scenes, especially in Matlab, they generally have a slower execution speed due to the overhead of interpretation.
- **C:** It's a compiled language, translating code directly into machine language, ensuring faster execution.

2. **Optimization Level:**
   - The optimization provided by the '-O3' flag in C makes a significant difference. It performs multiple passes over the code to optimize loops, inline functions, and apply other optimizations, resulting in highly efficient machine code.
3. **Data Types and Memory Management:**
   - **Matlab/Python:** They handle memory management, data type assignments, and other low-level tasks automatically, introducing some overhead.
   - **C:** Programmers have direct control over memory allocation, data types, and storage, which can lead to more efficient code when optimized properly.
4. **Library and Built-in Function Efficiency:**
   - **Matlab/Python:** These languages often rely on built-in functions and libraries, which are typically optimized and can sometimes perform on par with optimized C code for specific tasks.
   - **C:** Writing custom implementations in C requires careful optimization to outperform the highly optimized libraries available in Python/Matlab.

1.3 Conclusion:

The differences in timings can be attributed to the inherent characteristics of the programming languages and the optimization levels applied during compilation. C, especially with the '-O3' flag, is expected to provide faster execution due to its low-level nature and the extensive optimizations applied by the compiler. In contrast, Python and Matlab, while offering the advantage of rapid development and ease of use, may exhibit slower execution times for custom-implemented algorithms due to the overhead associated with higher-level abstractions and interpreted execution.