Try Blis - A Framework for Rapid BLAS Functionality

Justin M. Shea

2019-04-27

Introduction

Background

Testing and Benchmarking

Try BLIS! Thank You

About me: Justin M. Shea

- Visiting Professor of Finance, Roosevelt University
- TA/Co-instructor University of Chicago Master of Science in analytics Program, Capstone Project
- Google summer of code mentor 2019, R and Finance `PerformanceAnalytics` package.
- Organizer of The Chicago R User Group
- Committee member R/Finance 2019
- Organizer SatRday Chicago
- Author of the **`wooldridge`** R package, official R package for "Introductory Econometrics" by Jeffrey M. Wooldridge.
- Almost 2 decades in Finance, primarily Futures and Options trading

Data grows, hardware gets cheaper

- A Designer laptop that costs $3,000+ maybe a waste of your most valuable resources, **time & money**. It's slow and expensive.
- Some data is private/valuable and you may not want to (or be allowed to) outsource it to the cloud. Plus, that server time adds up $$$.
- You're already building your own Software for Data Analysis (`SoDa`), consider building your own Hardware for Data Analysis (`HoDa`) for similar reasons!
- **Consider BYOH: *Build Your Own HardWare!***

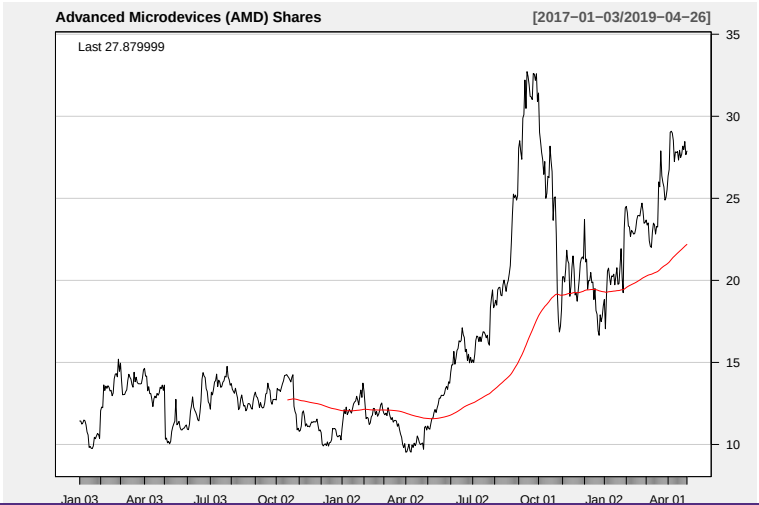
High-Performance Hardware Cheap?

- **August 2017**: Advanced Microdevices (`AMD`) x86 Core "Zen" architecture released
- `Ryzen` retail, `Threadripper` High-Performance, and `EPIC` Server product lines.
- Similar Performance as Intel, but at a deep discount.
- **Bottom Line for Data Peeps:**
  - `Twice the core counts for less money`
  - `Or more CPU for the same money`

`https://www.amd.com/en/technologies/zen-core`

Early adoption looks good

- **Market Response is very positive:** ZEN product lines are gaining acceptance among Graphics Designers, Gamers, and Data Professionals

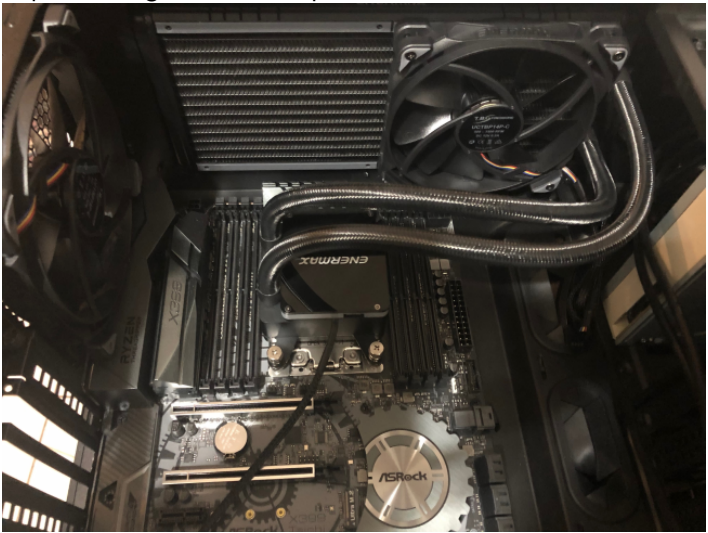


Taking a chance on AMD

- Bought a **Threadripper 1950X 3.4 GHz 16-Core** Processor

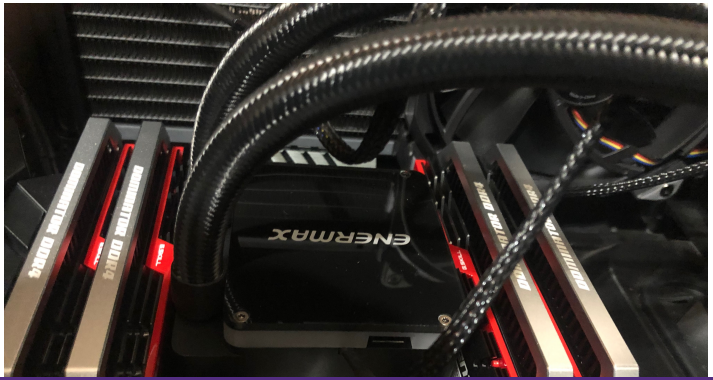
High-Performance CPUs get hot

Liquid cooling: Enermax Liqtech TR4 II 280 80.71 CFM CPU Cooler



RAM Installed: 64GB

- One *shouldn't* mix differing RAM. I did because I had extra and its **more RAM**
- When mixing `RAM`, `F2` into the `bios` on boot and adjust the MHZ accordingly.
- Mixing 3000 and 3200 kits, I was able to reach 2800 MHZ without overclocking.

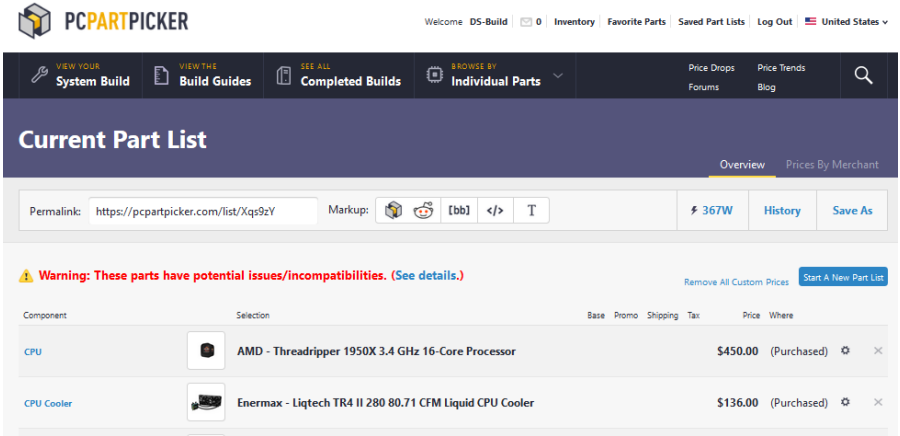


It's ALIVE

Complete List of parts

- I recycled parts from a previous build so cost $1500. $2500 from scratch.
- Time investment pays dividends via future upgrades scale and productivity.
- Cheaper than the average designer laptop, and more powerful!

Full Parts list available her: `https://pcpartpicker.com/list/Xqs9zY`



Benchmarking

Using the `bencharkme` package by Colin Gillespie.

```
library(benchmarkme)
```

- Contains functions which run `matrix algebra` computations on random data.
- Matrix algebra computations are the core of statistical/Machine Learning models.
- Contains `crowd-sourced` benchmarks from other useRs for comparison.

Benchmarks based heavily on the R script by **Simon Urbanek** & **Doug Bates**: `http://r.research.att.com/benchmarks/R-benchmark-25.R`

Benchmark: General Programming

- 3,500,000 Fibonacci numbers calculation (vector calc).
- Creation of a 3500x3500 Hilbert matrix (matrix calc).
- Grand common divisors of 1,000,000 pairs (recursion).
- Creation of a 1600x1600 Toeplitz matrix (loops).
- Escoufier's method on a 60x60 matrix (mixed)

Benchmark: Matrix Calculation

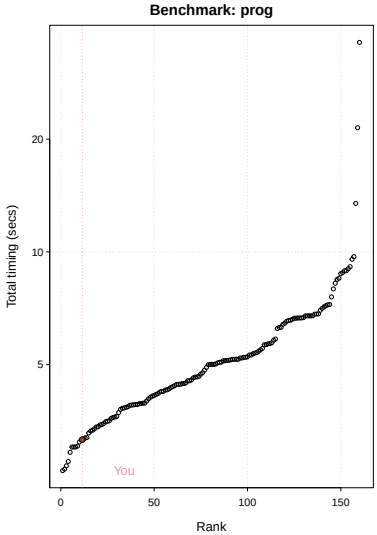
- Creation, transpose., deformation of a 2500x2500 matrix.
- 2500x2500 normal distributed random matrix ^1000.
- Sorting of 7,000,000 random values.
- 2500x2500 cross-product matrix (b = a' * a)
- Linear regression over a 3000x3000 matrix.

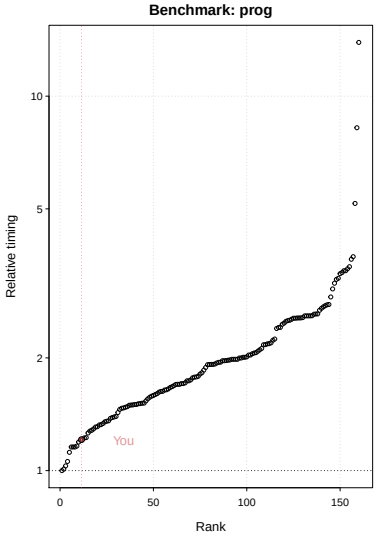
Benchmark: Matrix Functions

- FFT over 2,500,000 random values.
- Eigenvalues of a 640x640 random matrix.
- Determinant of a 2500x2500 random matrix.
- Cholesky decomposition of a 3000x3000 matrix.
- Inverse of a 1600x1600 random matrix.

base-R 1 core: General Programming

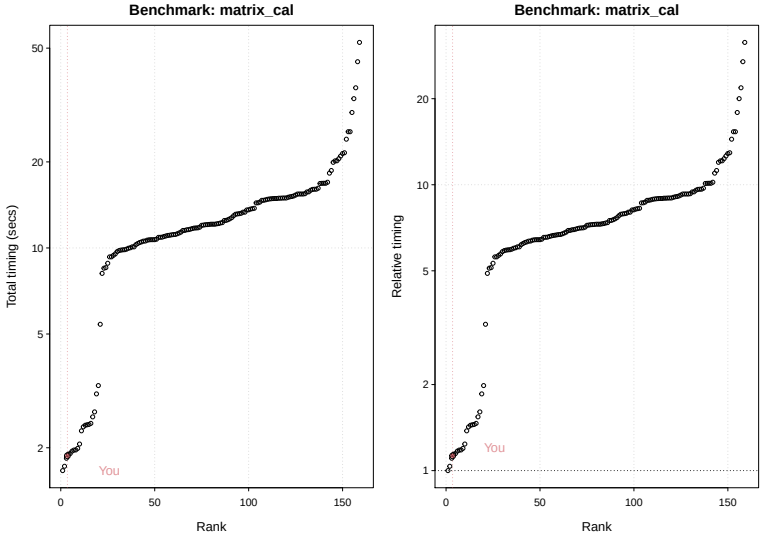
```
## You are ranked 12 out of 163 machines.
```





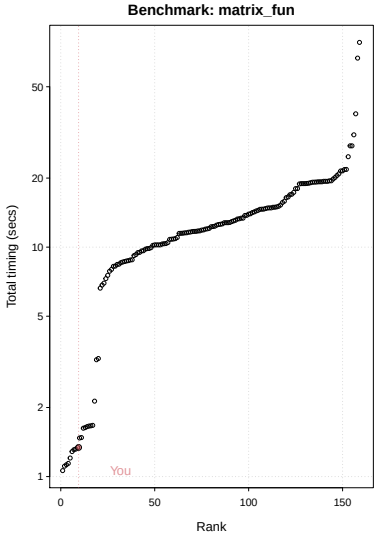
base-R 1 core: Matrix Calculation

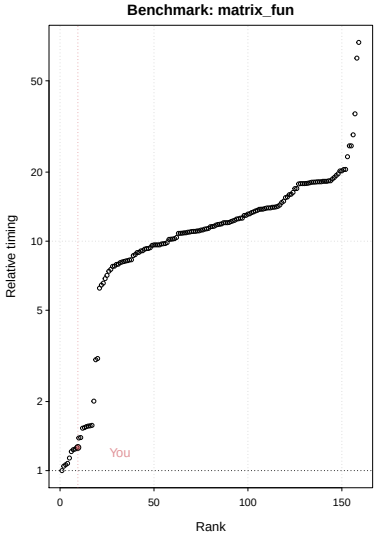
```
## You are ranked 4 out of 162 machines.
```



base-R 1 core: Matrix Functions

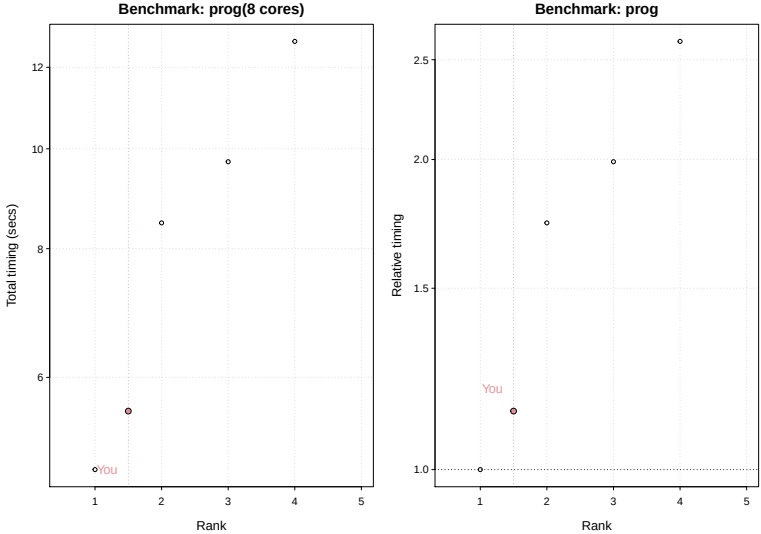
```
## You are ranked 10 out of 162 machines.
```





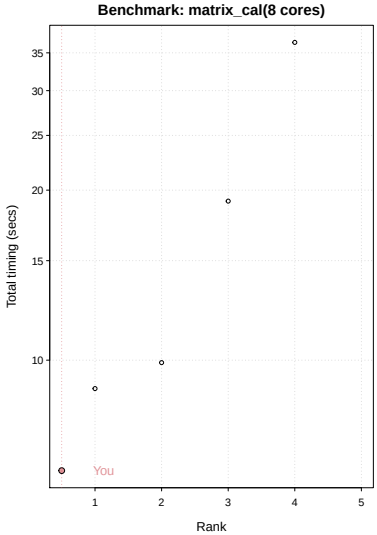
base-R 8 cores: General Programming

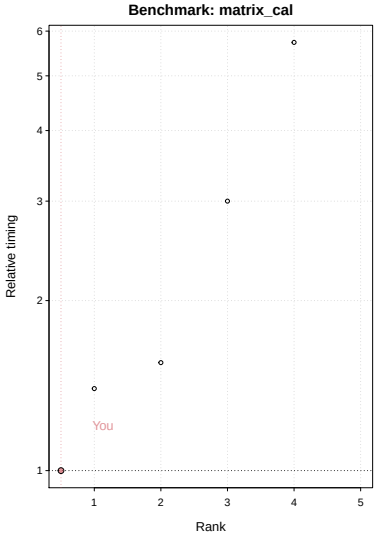
```
## You are ranked 2 out of 5 machines.
```



base-R 8 cores: Matrix Calculation

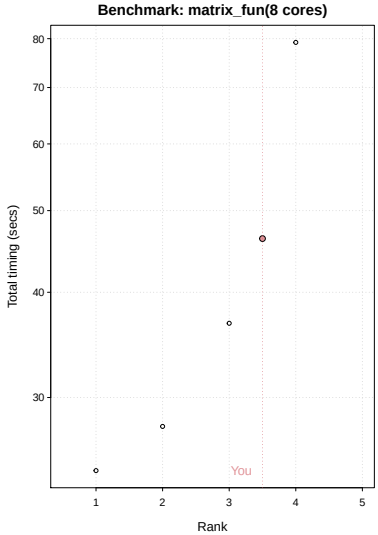
```
## You are ranked 1 out of 5 machines.
```

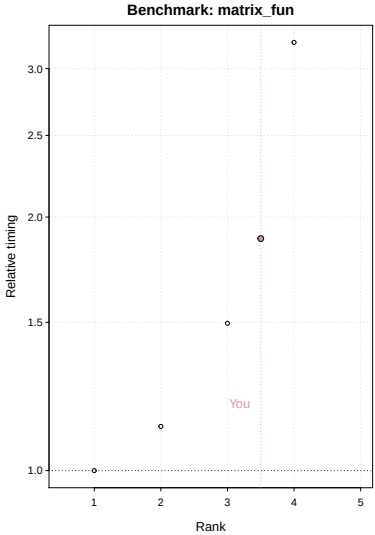




base-R 8 cores: Matrix Functions

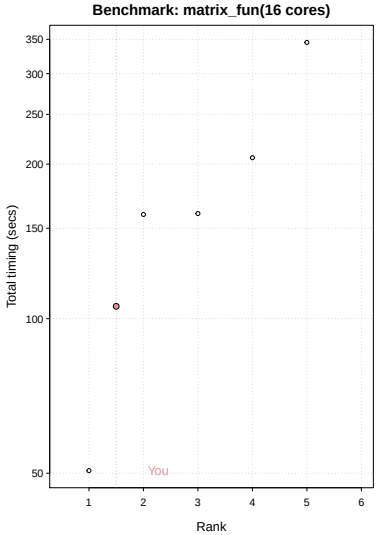
```
## You are ranked 4 out of 5 machines.
```

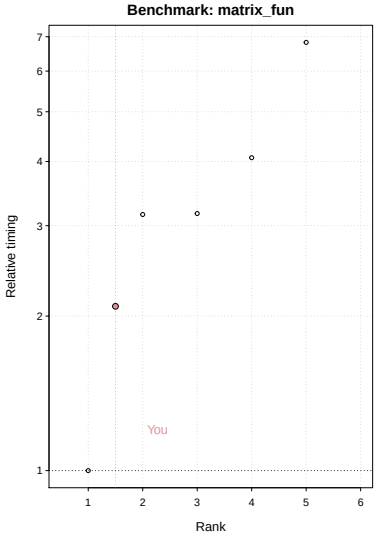




base-R 16 cores: Matrix Functions

```
## You are ranked 2 out of 7 machines.
```





base-R results summary

- General Programming Benchmarks are excellent!
- Linear Algebra Calculations are excellent!
- **Linear Algebra Functions lagging severely when run in Parallel!??**

The goal of this entire build was to run models in parallel, so this is no good!

Linear Algebra Functions in R

base-R interfaces with BLAS (Basic Linear Algebra Subprograms) routines that provide standard building blocks for performing linear algebra operations.

- scalar multiplication
- dot products
- linear combinations
- matrix operations
- Written in both C and FORTRAN

Papers and History here: http://www.netlib.org/blas/
Let's Try another BLAS!

From BLAS/LAPACK to BLIS/libFLAME

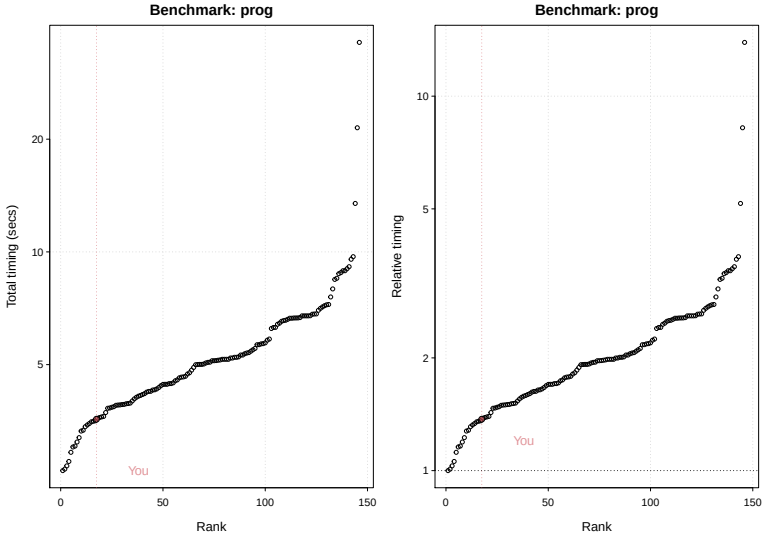
Poking around the internet and researching various BLAS libraries led me to BLIS and the FLAME project!
BLIS/libFlame are high performance dense linear algebra libraries, each addressing a layer in the linear algebra software stack.
Primarily developed and maintained by individuals in the Science of High-Performance Computing (SHPC) group in the Institute for Computational Engineering and Sciences at The University of Texas at Austin.
`https://www.cs.utexas.edu/~flame/web/`

BLIS/libFLAME

`BLIS` is a portable software framework for instantiating high-performance BLAS-like dense linear algebra libraries. The framework was designed to isolate essential kernels of computation that, when optimized, enable optimized implementations of most of its commonly used and computationally intensive operations. Build BLIS from source on Github here:
`https://github.com/flame/blis`
`libFLAME` is a high performance dense linear algebra library that is the result of the FLAME methodology for systematically developing dense linear algebra libraries. The FLAME methodology is radically different from the LINPACK/LAPACK approach that dates back to the 1970s, but is backwards compatible with them. Build libFLAME from source on Github here:
`https://github.com/flame/libflame/`
The best part is: They really ROCK!

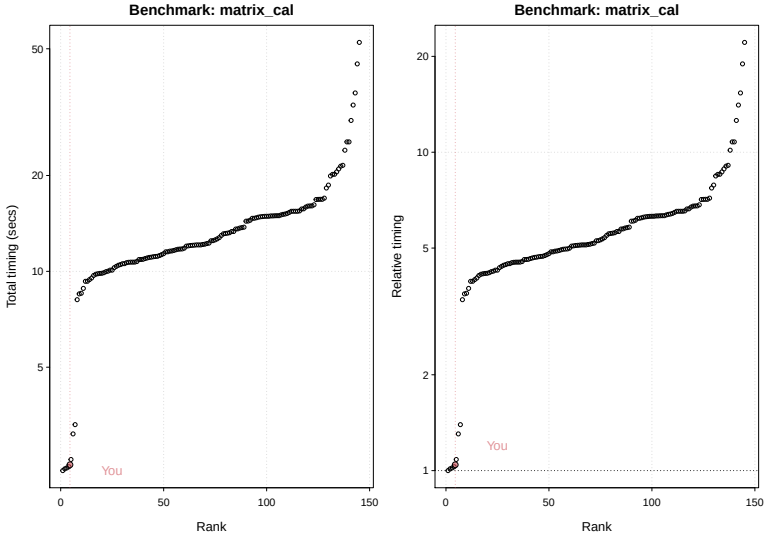
BLIS/libFLAME 1 core: Programming

```
## You are ranked 18 out of 149 machines.
```



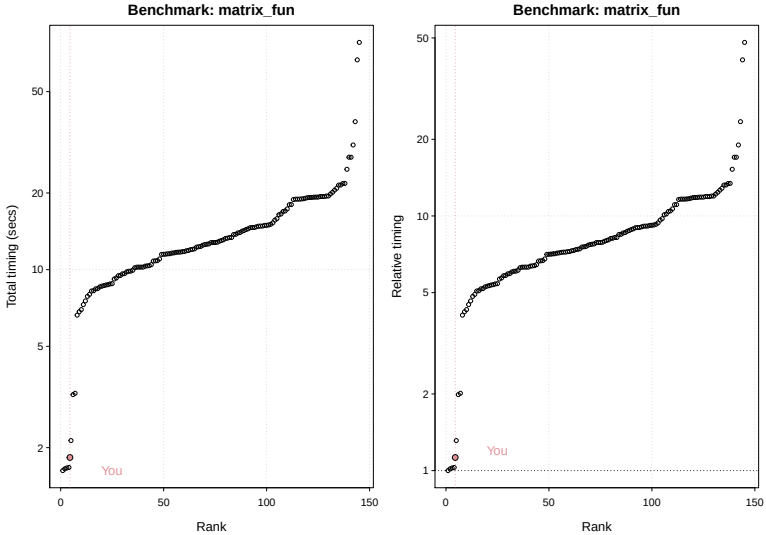
BLIS/libFLAME: Matrix Calculation

```
## You are ranked 5 out of 148 machines.
```



BLIS/libFLAME 1 core: Matrix Functions

```
## You are ranked 5 out of 148 machines.
```



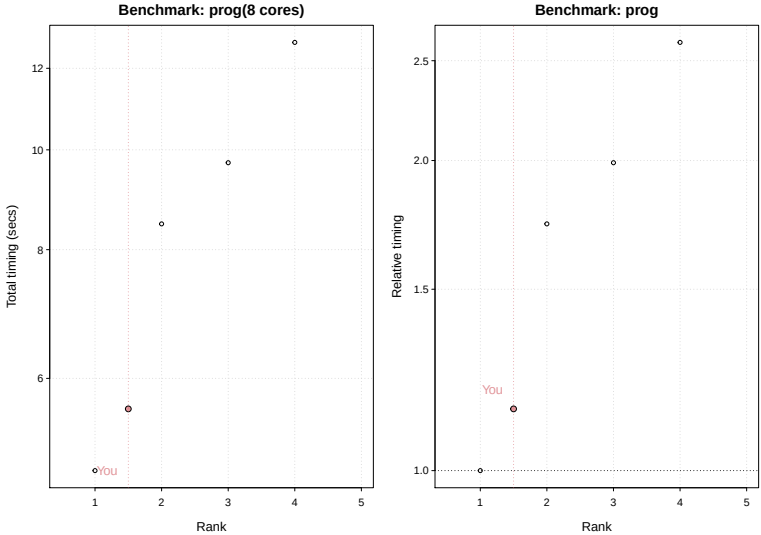
BLIS/libFLAME: Matrix Functions

Threadripper with BLIS places 5*th* of 162 with a time of 1.83 seconds.

time	cpu	ram	sysname	relea
1.625	Intel(R) Core(TM) i5-6300U CPU @ 2.40GHz	33.604	Linux	4.20
1.651	Intel(R) Core(TM) i5-6300U CPU @ 2.40GHz	33.604	Linux	4.20
1.664	Intel(R) Core(TM) i5-6300U CPU @ 2.40GHz	16.692	Linux	4.20
1.672	Intel(R) Core(TM) i5-6300U CPU @ 2.40GHz	16.692	Linux	4.20
1.830	AMD Ryzen Threadripper 1950X 16-Core Processor	NA	Linux	4.19
2.132	Intel(R) Core(TM) i5-3360M CPU @ 2.80GHz	16.678	Linux	4.20
3.230	Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz	33.482	Linux	4.15
3.270	Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz	33.482	Linux	4.15
6.626	Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz	NA	Windows	10 x
6.833	Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz	NA	Windows	>=

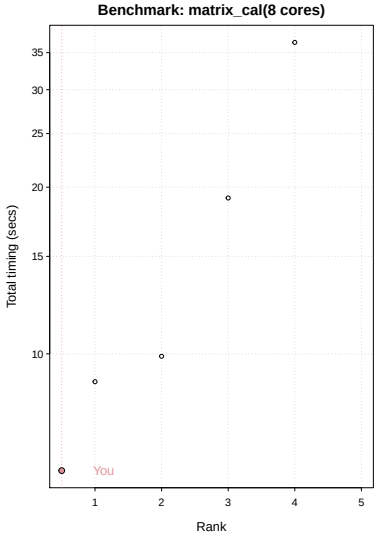
BLIS/libFLAME 8 core: Programming

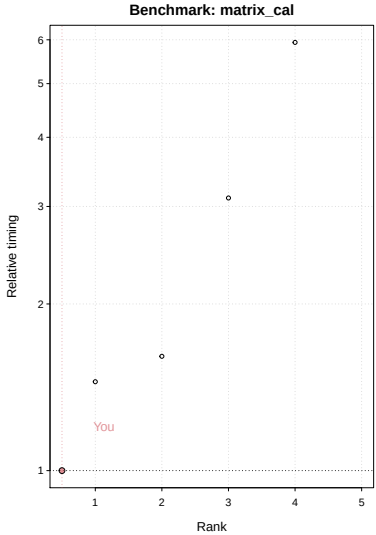
```
## You are ranked 2 out of 5 machines.
```



BLIS/libFLAME 8 core: M Calculation

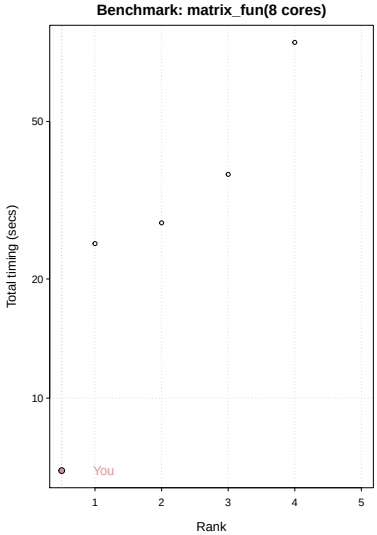
```
## You are ranked 1 out of 5 machines.
```

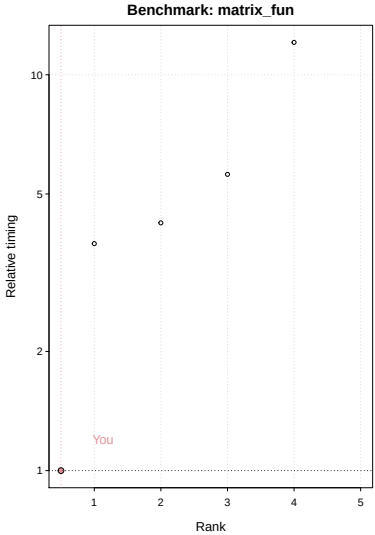




BLIS/libFLAME 8 core: M Functions

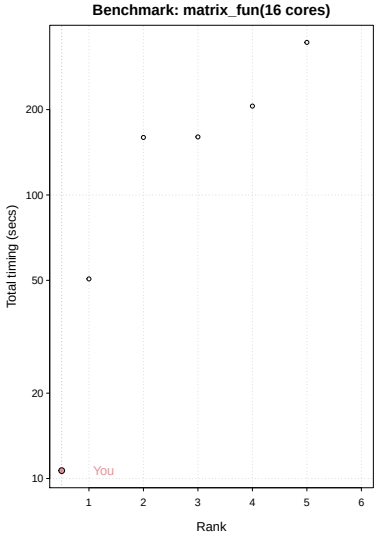
```
## You are ranked 1 out of 5 machines.
```

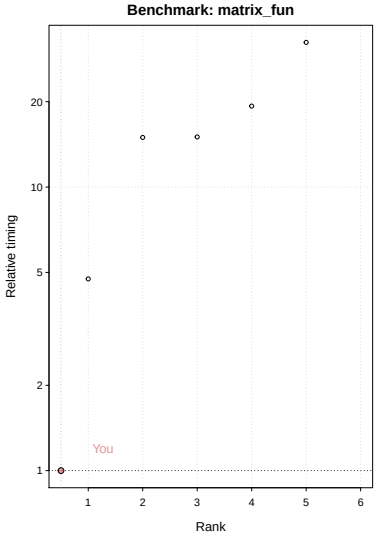




BLIS/libFLAME 16 cores: M Functions

```
## You are ranked 1 out of 7 machines.
```





BLIS/libFLAME 16 cores: M Functions

BLIS/libFLAME places 1*st* of 7 with a time of 10.66 seconds, beats the next submission by a factor of 5, and the last by a factor of 18.
No Intel submissions with this many cores to benchmark against.

time	cpu	ram	sysname	re
10.663	AMD Ryzen Threadripper 1950X 16-Core Processor	NA	Linux	4.
50.583	AMD Ryzen Threadripper 1950X 16-Core Processor	NA	Windows	>
159.536	AMD Ryzen Threadripper 1950X 16-Core Processor	33.664	Linux	4.
160.229	AMD Ryzen Threadripper 1950X 16-Core Processor	33.664	Linux	4.
164.089	AMD Ryzen Threadripper 1950X 16-Core Processor	33.664	Linux	4.
181.100	AMD Ryzen Threadripper 1950X 16-Core Processor	33.664	Linux	4.
205.805	AMD Ryzen Threadripper 1950X 16-Core Processor	33.664	Linux	4.

Fast Linear Algebra functions achieved!

On a single core, base-R **BLAS** is somewhat faster by 0.49 seconds, a factor of 1.37.

- BLAS Single 1.34 seconds.
- BLIS Single 1.83 seconds.

But on 8 cores, ***BLIS*** becomes much faster by 39.76 seconds, a factor of 7.06.

- BLAS 8-core 46.32 seconds.
- BLIS 8-core 6.56 seconds.

On 16 cores, **BLIS** is even faster yet by 95.02 seconds, a factor of 9.91.

- BLAS 16-core 105.69 seconds.
- BLIS 16-core 10.66 seconds.

Resources to switch BLAS to BLIS!

- **Dirk Eddelbuettel:** simple scripts to switch BLAS/LAPACK implementations `http://dirk.eddelbuettel.com/blog/2018/04/15/#018_mkl_for_debian_ubuntu`
- **Debian/Ubuntu Wiki:** Implementations alternative versions of BLAS and LAPACK `https://wiki.debian.org/DebianScience/LinearAlgebraLibraries`
- **Brett Klamer:** Windows setup `http://brettklamer.com/diversions/statistical/faster-blas-in-r/`
- **U Texas** Science of High-Performance Computing Group `http://shpc.ices.utexas.edu/software.html`
- **BLIS Github**: `https://github.com/flame/blis`
- **libFLAME Github**: `https://github.com/flame/libflame/`

Table of Contents

Introduction

Background

Testing and Benchmarking

Try BLIS! Thank You