OAK RIDGE National Laboratory

2025 OLCF User Conference Call

ChatBLAS: The First AI-Generated and Portable BLAS Library

PhD. Pedro Valero-Lara

Senior Computer Scientist, valerolara@ornl.gov

U.S. DEPARTMENT OF ENERGY

ORNL IS MANAGED BY UT-BATTELLE LLC
FOR THE US DEPARTMENT OF ENERGY

OAK RIDGE National Laboratory

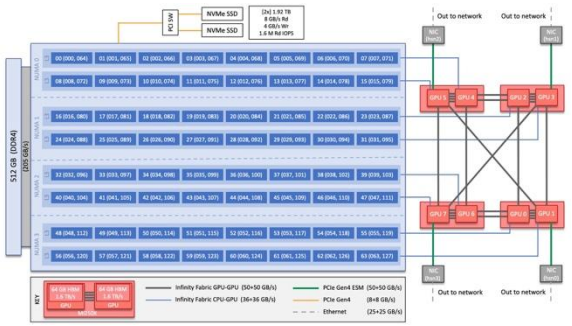
Motivation:

1. Evaluate the capabilities of current large language models (LLMs) to generate a *portable and HPC library* for BLAS operations.

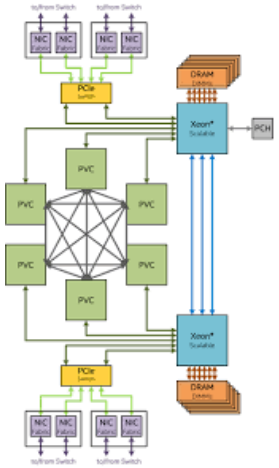
2. Define the *fundamental practices and criteria* to interact with LLMs for HPC targets to elevate the trustworthiness and performance levels of the AI-generated HPC codes.

Towards an AI-Assisted HPC ecosystem!

FRONTIER



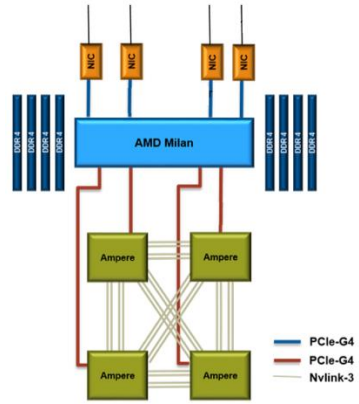
AMD ROCm



Aurora

oneAPI

SYCL™



NVIDIA CUDA

OpenMP

ChatBLAS Implementation

Use of BLAS (Level 1) standard API

One single API, multiple implementations (backends)

- Implement once, deploy everywhere!

Transparent interaction with LLMs (ChatGPT)

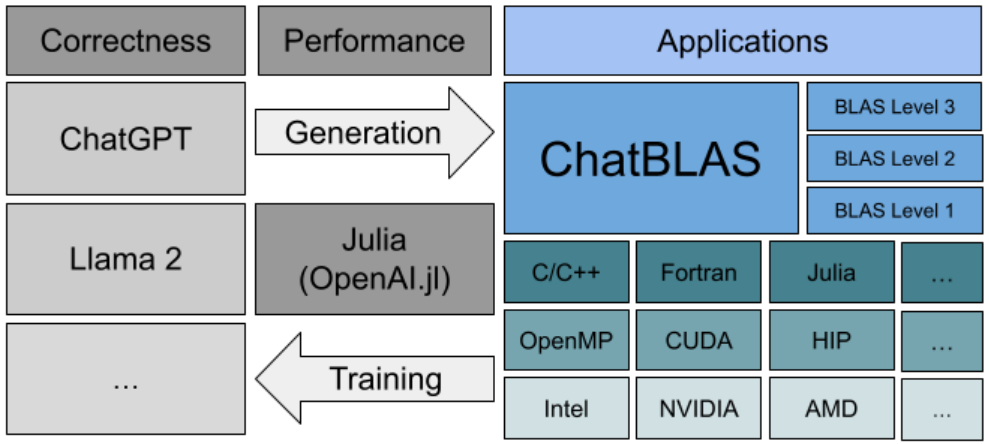
- ChatGPT implements BLAS codes
- ChatBLAS orchestrate the communication with ChatGPT and validates the BLAS codes

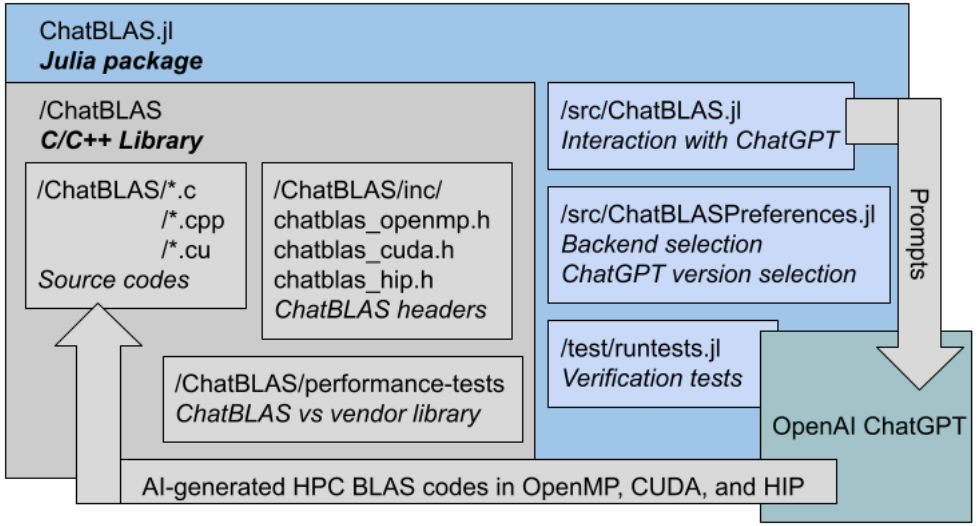
3 Different Backends

- OpenMP (multi-core CPUs), CUDA (NVIDIA GPUs) and HIP (AMD GPUs)

Julia's OpenAI.jl package

- Get benefit from Julia ecosystem





Correctness

Metric: Number of times that the required codes generated by ChatGPT were correct divided by the total number of times that the codes were generated

- We ran the tests for 10 iterations

We define correct codes as *"those that use the programming language (e.g., C/C++) and the programming model (e.g., OpenMP, CUDA, HIP) required and specified in the prompts, that can be compiled and run, and that provide a correct result"*

Prompt: *"Give me a function code only that computes a multiplication of a vector x by a constant a and the result is added to a vector y. Vectors x and y are length n, use C and CUDA to compute in parallel include the next line in the code, and use the next function name and parameters void chatblas saxpy(int n, float a, float *x, float *y). Include the next line at the beginning #include chatblas cuda.h"*

Routine	OpenMP		CUDA		HIP	
ChatGPT Version	3.5	4o	3.5	4o	3.5	4o
saxpy	1.0	1.0	0.8	0.8	1.0	1.0
sscal	0.6	1.0	0.8	0.6	1.0	0.8
sswap	1.0	1.0	0.4	0.4	0.8	1.0
scopy	1.0	1.0	0.6	0.8	1.0	0.8
sdot	1.0	1.0	1.0	0.6	1.0	0.8
sdsdot	1.0	1.0	0.2	0.2	0.6	0.2
sasum	0.8	1.0	0.6	1.0	0.4	1.0
snrm2	1.0	1.0	0.8	0.6	0.6	0.8
isamax	1.0	1.0	0.6	0.8	0.4	1.0

TABLE I
BLAS LEVEL-1 CORRECTNESS OF CHATBLAS USING BASIC PROMPTS.

ChatGPT 3.5

- ***OpenMP 93%, CUDA 66%, and HIP 75%***

ChatGPT 4o

- ***OpenMP 100%, CUDA 66%, and HIP 82%***

Prompt Engineering

This consists of complementing the prompts with more information regarding the operations that will be computed by the codes to minimize the deficiencies found in the codes (e.g., wrong use of CUDA indexes or the lack of CPU-GPU communication).

Prompt: *"Give me a kernel and a function only that computes a multiplication of a vector x by a constant a and the result is added to a vector y. Do not give a main function. Vectors x and y are length n, use C and CUDA to compute in parallel, allocate and free the GPU vectors, and make the CPU - GPU memory transfers in the function. The size of blocks of threads and the number of blocks must be defined. Use the next function name and parameters for the kernel global void saxpy kernel(int n, float a, float *x, float *y), and the next function name and parameters for the function void chatblas saxpy(int n, float a, float *x, float *y). Include the next line at the beginning of the code #include "chatblas cuda.h"."*

Routine	OpenMP		CUDA		HIP	
ChatGPT Version	3.5	4o	3.5	4o	3.5	4o
saxpy	1.0	1.0	0.8	1.0	1.0	0.8
sscal	1.0	1.0	1.0	1.0	1.0	1.0
sswap	1.0	1.0	1.0	1.0	1.0	1.0
scopy	1.0	1.0	1.0	0.8	0.8	1.0
sdot	1.0	1.0	0.6	0.8	1.0	0.8
sdsdot	1.0	1.0	0.4	0.8	0.4	0.2
sasum	1.0	1.0	0.8	0.8	1.0	0.4
snrm2	1.0	1.0	1.0	0.8	0.8	0.8
isamax	1.0	1.0	0.6	0.6	0.8	0.6

TABLE II
BLAS LEVEL-1 CORRECTNESS OF CHATBLAS USING PROMPT ENGINEERING.

ChatGPT 3.5

- ***OpenMP 100% [93%], CUDA 80% [66%], and HIP 86% [75%]***

ChatGPT 4o

- ***OpenMP 100% [100%], CUDA 84% [66%], and HIP 73% [82%]***

Fine-Tuning

This consists of training the AI generative models with the necessary information to reduce the deficiencies of the codes generated by AI.

- This effort is made on only the 3.5-Turbo version of ChatGPT because OpenAI does not allow users to do so on the 4o version

JSON file

- Question: Basically, the one that is used by ChatBLAS to interact with ChatGPT
- Answer: We used the correct codes generated by ChatGPT 4o to fine-tune ChatGPT 3.5 Turbo
- There must be at least 10 cases to fine-tune ChatGPT

We elevated the correctness of the BLAS Level-1 codes generated by ChatGPT 3.5-Turbo until 100% in all the cases (OpenMP, CUDA and HIP)

```
{ "messages": [
  { "role": "system", "content":
    "Code generator for saxpy kernel implemented in C and HIP"
  },
  { "role": "user", "content":
    "Only respond with code as plain text without code block syntax around it. Give me a kernel and a function only that
    ↪  computes a multiplication of a vector x by a constant a and the result is added to a vector y. Do not give a main
    ↪  function. Vectors x and y are length n, use C and HIP to compute in parallel, allocate and free the GPU vectors
    ↪  and make the CPU - GPU memory transfers in the function. The size of blocks of threads and the number of blocks
    ↪  must be defined. Use next function name and parameters for the kernel __global__ void saxpy_kernel(int n, float
    ↪  a, float *x, float *y) { and the next function name and parameters for the function void chatblas_saxpy(int n,
    ↪  float a, float *x, float *y) {. Include the next line at the beginning of the code #include \"chatblas_hip.h\"."
    },
    { "role": "assistant", "content":
      "
      #include \"chatblas_hip.h\"
      __global__ void saxpy_kernel(int n, float a, float *x, float *y) {
        int i = blockIdx.x * blockDim.x + threadIdx.x;
        if (i < n) {
          y[i] = a * x[i] + y[i];
        }
      }
      void chatblas_saxpy(int n, float a, float *x, float *y) {
        float *d_x, *d_y;
        hipMalloc((void**)&d_x, n * sizeof(float)); hipMalloc((void**)&d_y, n * sizeof(float));
        hipMemcpy(d_x, x, n * sizeof(float), hipMemcpyHostToDevice);
        hipMemcpy(d_y, y, n * sizeof(float), hipMemcpyHostToDevice);
        int blockSize = 256;
        int numBlocks = (n + blockSize - 1) / blockSize; hipLaunchKernelGGL(saxpy_kernel, numBlocks, blockSize, 0, 0, n,
        ↪  a, d_x, d_y);
        hipMemcpy(y, d_y, n * sizeof(float), hipMemcpyDeviceToHost);
        hipFree(d_x);
        hipFree(d_y);
      }"
    }
] }
```

Performance

We compare the performance of ChatBLAS against vendor libraries:

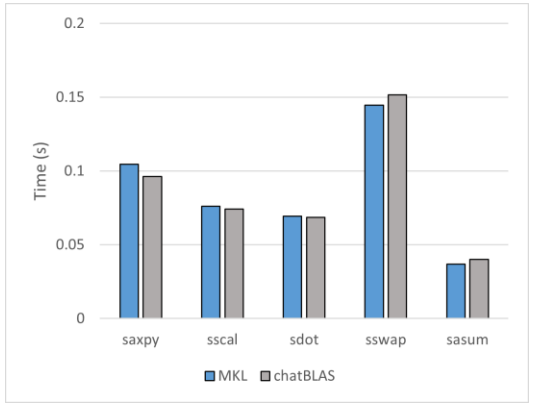
- Intel MKL on an Intel CPU Xeon E5-2698 v4 Broadwell 20-core
- AMD AOCL on an AMD CPU EPYC 7742 Rome 64-core
- cuBLAS on a NVIDIA GPU Ampere A100
- hipBLAS on a AMD GPU Mi100

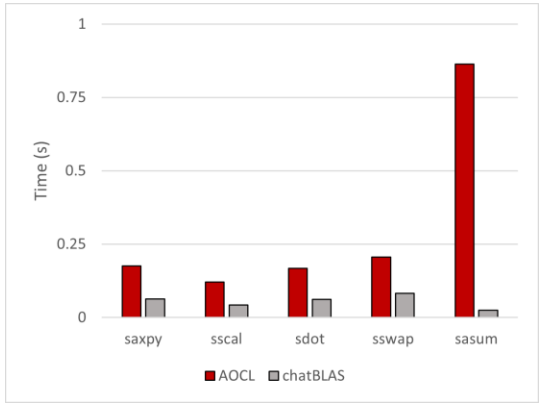
Competetive performance w.r.t. Intel MKL

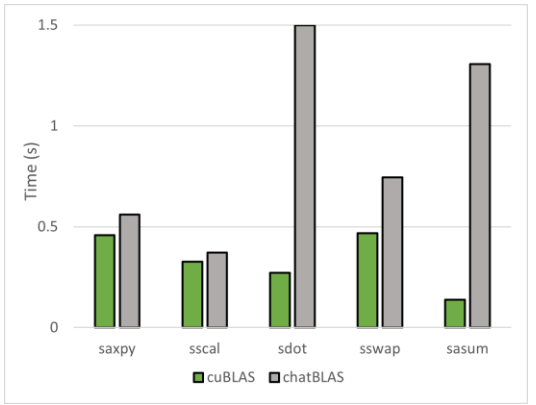
Better performance w.r.t. AMD AOCL

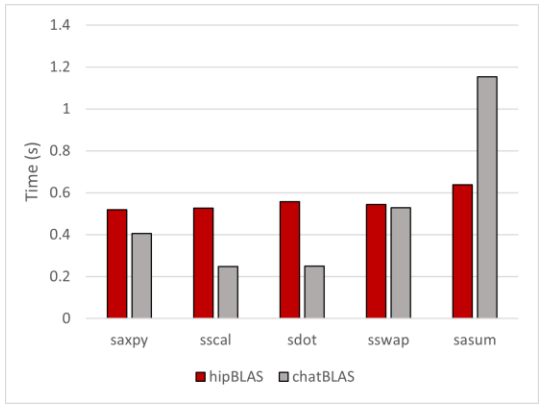
- AMD AOCL does not provide parallel codes for BLAS Level-1

Mixed results w.r.t. cuBLAS and hipBLAS







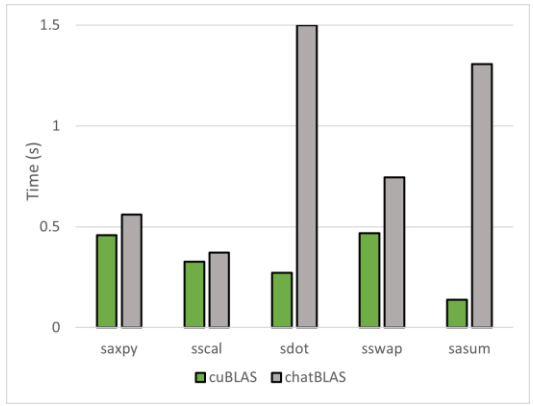


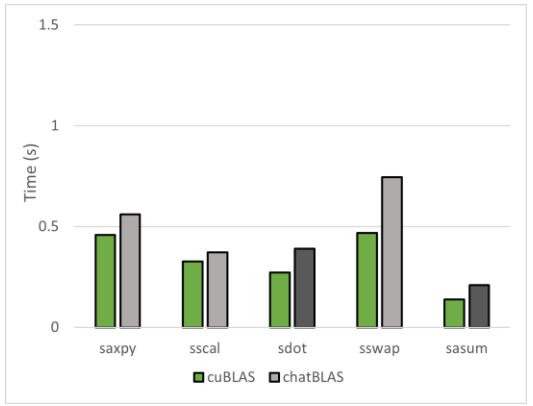
Fine-Tuning for Performance

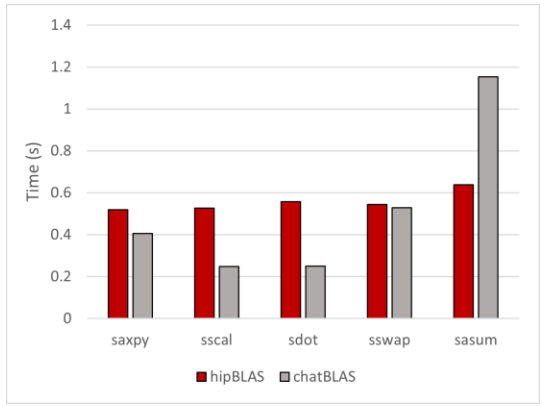
Train ChatBLAS to use the proper algorithm

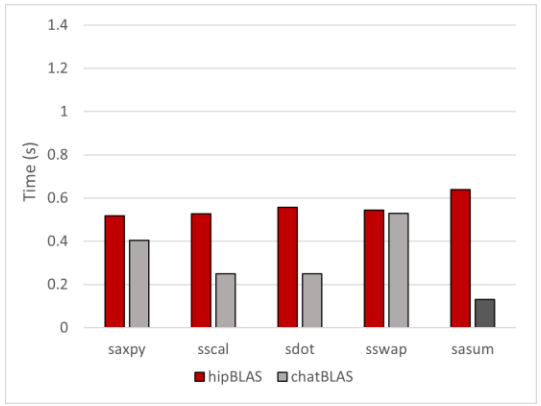
```
__global__ void sasum_kernel(int n, float *x, float *sum) {
    int idx = blockIdx.x * blockDim.x + threadIdx.x;
    if (idx < n) {
        atomicAdd(sum, fabsf(x[idx]));
    }
}
```

```
__global__ void sasum_kernel(int n, float *x, float *sum) {
    __shared__ float cache[512];
    int tid = threadIdx.x + blockIdx.x * blockDim.x;
    int cacheIndex = threadIdx.x;
    float temp = 0;
    while (tid < n) {
        temp += abs(x[tid]);
        tid += blockDim.x * gridDim.x;
    }
    cache[cacheIndex] = temp;
    __syncthreads();
    int i = blockDim.x / 2;
    while (i != 0) {
        if (cacheIndex < i)
            cache[cacheIndex] += cache[cacheIndex + i];
        __syncthreads();
        i /= 2;
    }
    if (cacheIndex == 0) sum[blockIdx.x] = cache[0];
}
```









OAK RIDGE National Laboratory

2025 OLCF User Conference Call

Conclusions
What's next??

ChatBLAS:
The First AI-Generated and Portable BLAS Library

https://github.com/pedrovalerolara/ChatBLAS/

- Evaluate and define best practices LLMs for portable and HPC BLAS (level-1) codes
  - Prompt Engineering
  - Fine-tuning
- Competitive and better performance than vendor's libraries
  - Filling the gaps found in vendor's stack (AMD AOCL)
- Extend ChatBLAS for the other BLAS levels and programming languages
  - Algorithms

OAK RIDGE National Laboratory

2025 OLCF User Conference Call

Thanks!! Questions??

PhD. Pedro Valero-Lara

Senior Computer Scientist, valerolara@ornl.gov

U.S. DEPARTMENT OF ENERGY

ORNL IS MANAGED BY UT-BATTELLE LLC
FOR THE US DEPARTMENT OF ENERGY