Introduction to High Performance Machine Learning
ECE-GY 9143

Lab 1

Deadline: Feb 18, 2022

You are allowed to discuss this homework with your classmates. However, any work that you submit must be your own which means you cannot share your code, and any work that you submit with your write-up must be written by only you.

Theoretical questions are identified by Q<number> while coding exercises are identified by C<number>. Submit a tar-archive named with your netID (e.g. ab1234.tar) that unpacks to
<email-id>/dp1.c
<email-id>/dp2.c
<email-id>/dp3.c
<email-id>/dp4.py
<email-id>/dp5.py
<email-id/results.pdf
to NYU classes.

The pdf contains the outputs of the programs and the answers to the questions.

C1 *(6 points)*:

Write a micro-benchmark that investigates the performance of computing the dot-product in C that takes two arrays of 'float' (32 bit) as input. The dimension of the vector space
and the number of repetitions for the measurement are command-line arguments,
i.e. a call 'dp1 1000 10' performs 10 measurements on a dot product with vectors
of size 1000. Initialize fields in the input vectors to 1.0.

```
float dp(long N, float *pA, float *pB) {
  float R = 0.0;
  int j;
  for (j=0;j<N;j++)
    R += pA[j]*pB[j];
  return R;
}
```

Name the program dp1.c and compile it with *gcc -O3 -Wall -o dp1 dp1.c*

Make sure the code is executed on a Prince node with Intel(R) Xeon(R) CPU E5-2690 v2 @ 3.00GHz or similar and enough RAM, the 300000000 size runs should not be killed by the system!

1. Measure the execution time of the function with clock_gettime(CLOCK_MONOTONIC).
2. Measure the time for N=1000000 and N=300000000.
3. Perform 1000 repetitions for the small case and 20 repetitions for the large case.
4. Compute the appropriate mean for the execution time for the second half of the repetitions.

For the average times, compute the bandwidth in GB/sec and throughput in FLOP/sec, print the result as

N: 1000000 <T>: 9.999999 sec B: 9.999 GB/sec F: 9.999 FLOP/sec
N: 300000000 <T>: 9.999999 sec B: 9.999 GB/sec F: 9.999 FLOP/sec

C2 *(3 points)*:

Perform the same microbenchmark with

```
float dpunroll(long N, float *pA, float *pB) {
  float R = 0.0;
  int j;
  for (j=0;j<N;j+=4)
    R += pA[j]*pB[j] + pA[j+1]*pB[j+1] + pA[j+2]*pB[j+2] + pA[j+3] * pB[j+3];
  return R;
}
```

C3 *(3 points)*:

Perform the same microbenchmark with MKL (Intels library), you may need to install a 'module' on the prince to access MKL.

```
#include <mkl_cblas.h>
float bdp(long N, float *pA, float *pB) {
  float R = cblas_sdot(N, pA, 1, pB, 1);
  return R;
}
```

C4 *(6 points)*:

Implement the same microbenchmark in python, using NumPy arrays as input

```
A = np.ones(N,dtype=np.float32)
B = np.ones(N,dtype=np.float32)
```

for a simple loop

```
def dp(N,A,B):
    R = 0.0;
    for j in range(0,N):
        R += A[j]*B[j]
    return R
```

C5 *(4 points)*:

Perform the same measurements using 'numpy.dot'.

Q1 (3 points):

Explain the consequence of only using the second half of the measurements
for the computation of the mean.

Q2 *(6 points)*:

Draw a roofline model based on 200 GFLOPS and 30 GB/s. Add
a vertical line for the arithmetic intensity.
Draw points for the 10 measurements for the average results for the microbenchmarks.

Q3 *(5 points)*:

Using the N=300000000 simple loop as the baseline, explain the
the difference in performance for the other 5 measurements in the C variants.

Q3 *(6 points)*:

Check the result of the dot product computations against the analytically calculated result.
Explain your findings. (Hint: Floating point operations are not exact.)

You need to get access to NYU HPC 'Greene' cluster. Instructions are posted under Discussions on Brightspace.
As a tip for the double-log chart for the roofline model, you can use a tool like gnuplot.
Note that a bandwidth line may not reach an X-axis plotted at Y=0!