```
times = [512, 1024, 2048]
```

CPU

```
from google.colab import files
uploaded = files.upload()
```

```
Choose Files matrix_cpu.c
matrix_cpu.c(n/a) - 1195 bytes, last modified: 1/27/2026 - 100% done
Saving matrix_cpu.c to matrix_cpu.c
```

```
!gcc matrix_cpu.c -o matrix_cpu -O2
```

```
import subprocess

cpu_times = []

for t in times:
    result = subprocess.run(['./matrix_cpu', str(t)], capture_output=True, text=True)
    print(result.stdout)
    line = result.stdout.strip()
    elapsed = float(line.split(':')[-1].split()[0])
    cpu_times.append(elapsed)
```

```
CPU execution time (N=512): 0.336090 seconds

CPU execution time (N=1024): 3.268395 seconds

CPU execution time (N=2048): 68.398771 seconds
```

Naive GPU

```
uploaded = files.upload()
```

```
Choose Files naive_matrix_gpu.cu
naive_matrix_gpu.cu(n/a) - 2035 bytes, last modified: 1/29/2026 - 100% done
Saving naive_matrix_gpu.cu to naive_matrix_gpu.cu
```

```
!nvcc -arch=sm_75 naive_matrix_gpu.cu -o naive_matrix_gpu
```

```
naive_gpu_times = []

for t in times:
    result = subprocess.run(['./naive_matrix_gpu', str(t)], capture_output=True, text=True)
    print(result.stdout)
    line = result.stdout.strip()
    elapsed = float(line.split(':')[-1].split()[0])
    naive_gpu_times.append(elapsed)

naive_gpu_times = [t/1000 for t in naive_gpu_times]
```

```
GPU execution time (N=512): 1.265440 ms

GPU execution time (N=1024): 9.313984 ms

GPU execution time (N=2048): 74.942337 ms
```

Optimized GPU

```
uploaded = files.upload()
```

Choose Files optimized_matrix_gpu.cu

```
!nvcc -arch=sm_75 optimized_matrix_gpu.cu -o optimized_matrix_gpu
```

```
optimized_gpu_times = []

for t in times:
    result = subprocess.run(['./optimized_matrix_gpu', str(t)], capture_output=True, text=True)
    print(result.stdout)
    line = result.stdout.strip()
    elapsed = float(line.split(':')[-1].split()[0])
    optimized_gpu_times.append(elapsed)

optimized_gpu_times = [t/1000 for t in optimized_gpu_times]
```

```
GPU execution time (N=512): 0.837408 ms

GPU execution time (N=1024): 5.923168 ms

GPU execution time (N=2048): 46.388287 ms
```

Table 1

```
import pandas as pd

data = {
    'Implementation': ['CPU (C)', 'Naive CUDA', 'Optimized CUDA'],
    'N=512': [cpu_times[0], naive_gpu_times[0], optimized_gpu_times[0]],
    'Speedup 512': [1, cpu_times[0] / naive_gpu_times[0], cpu_times[0] / optimized_gpu_times[0]],
    'N=1024': [cpu_times[1], naive_gpu_times[1], optimized_gpu_times[1]],
    'Speedup 1024': [1, cpu_times[1] / naive_gpu_times[1], cpu_times[1] / optimized_gpu_times[1]],
    'N=2048': [cpu_times[2], naive_gpu_times[2], optimized_gpu_times[2]],
    'Speedup 2048': [1, cpu_times[2] / naive_gpu_times[2], cpu_times[2] / optimized_gpu_times[2]]
}

df = pd.DataFrame(data)
df
```

	Implementation	N=512	Speedup 512	N=1024	Speedup 1024	N=2048	Speedup 2048
0	CPU (C)	0.336090	1.000000	3.268395	1.000000	68.398771	1.000000
1	Naive CUDA	0.001265	265.591415	0.009314	350.912671	0.074942	912.685322
2	Optimized CUDA	0.000837	401.345581	0.005923	551.798463	0.046388	1474.483656

Next steps: Generate code with df New interactive sheet

Cublas GPU

```
uploaded = files.upload()
```

Choose Files cublas_matrix.cu

cublas_matrix.cu(n/a) - 1779 bytes, last modified: 1/29/2026 - 100% done

Saving cublas_matrix.cu to cublas_matrix.cu

```
!nvcc cublas_matrix.cu -lcublas -o cublas_matrix
```

```
cublas_gpu_times = []

for t in times:
    result = subprocess.run(['./cublas_matrix', str(t)], capture_output=True, text=True)
    print(result.stdout)
    line = result.stdout.strip()
    elapsed = float(line.split(':')[-1].split()[0])
    cublas_gpu_times.append(elapsed)

cublas_gpu_times = [t/1000 for t in cublas_gpu_times]
```

```
cuBLAS SGEMM time (N=512): 49.311424 ms

cuBLAS SGEMM time (N=1024): 6.330176 ms

cuBLAS SGEMM time (N=2048): 13.898208 ms
```

Table 2

```
df.loc[len(df)] = ['cuBLAS',
                   cublas_gpu_times[0],
                   cpu_times[0] / cublas_gpu_times[0],
                   cublas_gpu_times[1],
                   cpu_times[1] / cublas_gpu_times[1],
                   cublas_gpu_times[2],
                   cpu_times[2] / cublas_gpu_times[2]
                   ]

df
```

	Implementation	N=512	Speedup 512	N=1024	Speedup 1024	N=2048	Speedup 2048
0	CPU (C)	0.336090	1.000000	3.268395	1.000000	68.398771	1.000000
1	Naive CUDA	0.001265	265.591415	0.009314	350.912671	0.074942	912.685322
2	Optimized CUDA	0.000837	401.345581	0.005923	551.798463	0.046388	1474.483656
3	cuBLAS	0.049311	6.815662	0.006330	516.319767	0.013898	4921.409364

Next steps: Generate code with df | New interactive sheet

```
uploaded = files.upload()
```

Choose Files matrix_lib.cu
matrix_lib.cu(n/a) - 1775 bytes, last modified: 1/29/2026 - 100% done
Saving matrix_lib.cu to matrix_lib.cu

```
!nvcc -Xcompiler -fPIC -shared matrix_lib.cu -o libmatrix.so
```

```
uploaded = files.upload()
```

Choose Files lib_matrix.py
lib_matrix.py(text/x-python-script) - 740 bytes, last modified: 1/29/2026 - 100% done
Saving lib_matrix.py to lib_matrix.py

```
!python3 lib_matrix.py
```

```
Python call to CUDA library completed in 0.1770 seconds
```