

# CUDA ftest Kernel Execution Time Prediction Report

This report applies the CUDA kernel performance prediction template to the kernel: ftest, launched with Grid = (5,5) and Block = (32,32).

We estimate the expected per-launch execution time on five NVIDIA GPUs:

- GeForce GTX TITAN Black
- GeForce GTX TITAN X
- NVIDIA TITAN V
- GeForce RTX 2080 Ti
- GeForce RTX 4070

```
__global__ void ftest(int diagFlag, int p, int rows, int colsx, int colsy,
                     int rCols, int unrCols,
                     float * obs, int obsDim,
                     float * rCoeffs, int rCoeffsDim,
                     float * unrCoeffs, int unrCoeffsDim,
                     float * rdata, int rdataDim,
                     float * unrdata, int unrdataDim,
                     float * dfStats) {
    int j = blockIdx.x * THREADSPERDIM + threadIdx.x;
    int i = blockIdx.y * THREADSPERDIM + threadIdx.y;
    int idx = i*colsx + j;
    ...
}
```

## 1. Effective Parallelism

Launch configuration:

- Grid = (5,5) → 25 blocks
- Block = (32,32) → 1024 threads/block
- THREADSPERDIM = 16
- colsx = 5, colsy = 5, rows = 100

Indices:

$$\begin{aligned}j &= \text{blockIdx.x} * \text{THREADSPERDIM} + \text{threadIdx.x} \\i &= \text{blockIdx.y} * \text{THREADSPERDIM} + \text{threadIdx.y}\end{aligned}$$

Because THREADSPERDIM = 16, only blockIdx.x = 0 and threadIdx.x = 0..4 produce  $j < \text{colsx}$  (=5). Similarly, only blockIdx.y = 0 and threadIdx.y = 0..4 produce  $i < \text{colsy}$  (=5).

Thus there are  $5 \times 5 = 25$  active (i,j) pairs, all in block (0,0) with threadIdx.{x,y} in [0,4]. All other threads return immediately.

## 2. Per-thread Work and FLOP Count

For each active (i,j) pair, the kernel:

- Loops over k = 0..rows-1 (100 iterations).
- Computes restricted and unrestricted linear-model fits using rCols = 5 and unrCols = 10.

Inside the k-loop, approximate floating-point operations per iteration:

- rEst loop ( $m = 0..4$ ): 5 multiplications + 5 additions → 10 FLOPs
- unrEst loop ( $m = 0..9$ ): 10 multiplications + 10 additions → 20 FLOPs

- Accumulate squared residuals ( $rSsq$ ,  $unrSsq$ ):  $\approx 8$  FLOPs

Total per-k iteration  $\approx 38$  FLOPs.

Over rows = 100:

$F_{loop} \approx 100 \times 38 = 3,800$  FLOPs per thread.

After the loop, the F-statistic computation adds on the order of 10 FLOPs.

We approximate per-thread FLOPs as:

$F_{thread} \approx 3,800 + O(10) \approx 3.8 \times 10^3$  FLOPs.

With 25 active threads:

$F_{total} \approx 3.8 \times 10^3 \times 25 \approx 9.5 \times 10^3$  FLOPs.

### 3. Memory Traffic (Approximate)

For each (i,j) thread and each row k, we access:

- obs value: 1 float load  $\rightarrow 4$  bytes
- rCoeffs & rdata ( $rCols = 5$ ):  $\approx 5$  coefficient loads + 5 data loads
- unrCoeffs & unrdata ( $unrCols = 10$ ):  $\approx 10$  coefficient loads + 10 data loads

We approximate  $\approx 31$  float loads per k  $\rightarrow 31 \times 4$  bytes  $\approx 124$  bytes per row.

Over rows = 100:

$B_{thread} \approx 100 \times 124$  bytes  $\approx 12,400$  bytes  $\approx 1.24 \times 10^3$  bytes.

With 25 active threads:

$B_{total} \approx 1.24 \times 10^3 \times 25 \approx 3.1 \times 10^4$  bytes ( $\approx 0.30$  MiB)

The final write of  $dfStats[idx]$  adds only a single float store per thread and is negligible.

### 4. GPU Specifications Used

Approximate FP32 peak performance and memory bandwidth:

GPU	Peak FP32 (FLOPs/s)	Bandwidth (bytes/s)
GTX TITAN Black	5.12e12	3.36e11
GTX TITAN X	6.14e12	3.365e11
TITAN V	1.49e13	6.528e11
RTX 2080 Ti	1.345e13	6.16e11
RTX 4070	2.9e13	5.04e11

### 5. Time Estimates

Using:

- $F_{total} \approx 9.5 \times 10^3$  FLOPs
- $B_{total} \approx 3.125 \times 10^4$  bytes

We compute:

$$t_{\text{compute}} = F_{\text{total}} / \text{Peak\_FP32}$$

$$t_{\text{mem}} = B_{\text{total}} / \text{Bandwidth}$$

$$t_{\text{body}} = \max(t_{\text{compute}}, t_{\text{mem}})$$

$t_{\text{total}} \approx t_{\text{body}} + 5 \mu\text{s}$  (for CUDA kernel launch overhead).

GPU	$t_{\text{compute}} (\mu\text{s})$	$t_{\text{mem}} (\mu\text{s})$	$t_{\text{body}} (\mu\text{s})$	$t_{\text{total}} (\mu\text{s})$
GTX TITAN Black	0.019	0.930	0.930	$\approx 5.930$
GTX TITAN X	0.015	0.929	0.929	$\approx 5.929$
TITAN V	0.006	0.479	0.479	$\approx 5.479$
RTX 2080 Ti	0.007	0.507	0.507	$\approx 5.507$
RTX 4070	0.003	0.620	0.620	$\approx 5.620$

## 6. Conclusion

The ftest kernel performs roughly  $9.5 \times 10^9$  FLOPs and moves about 0.30 MiB of data per launch, spread across only 25 active (i,j) threads.

On all five GPUs, both  $t_{\text{compute}}$  and  $t_{\text{mem}}$  are well below 1 microsecond. As a result, the overall runtime is dominated by the fixed CUDA kernel launch overhead ( $\sim 5 \mu\text{s}$ ).

Predicted per-launch times:

- GTX TITAN Black:  $\approx 5.93 \mu\text{s}$
- GTX TITAN X:  $\approx 5.93 \mu\text{s}$
- TITAN V:  $\approx 5.48 \mu\text{s}$
- RTX 2080 Ti:  $\approx 5.51 \mu\text{s}$
- RTX 4070:  $\approx 5.62 \mu\text{s}$

These values are approximate but consistent with expectations for a small, branch-free arithmetic kernel whose runtime is dominated by launch latency.