Stanford University

CME 253A

Introduction to High Performance Computing and Parallel (GPU) Computing

Stanford Summer Session 3

1 July 2019 | Y2E2 111

Ludovic Räss

Stanford University, Department of Geophysics

lraess@stanford.edu

SIGMA

Session 3 - Performance evaluation

Today's agenda

- Lecture: Performance limiters and evaluation

- Programming: 1/ Evaluate memory copy throughput
  2/ Effective memory throughput acoustic 2D

- Tasks: 1/ Evaluate performance of your code
  2/ Produce a graph similar to shown today (to be included in your project)
  3/ Work on your Elastic wave 3D or Stokes 3D

Session 3 - Performance evaluation

Today's agenda

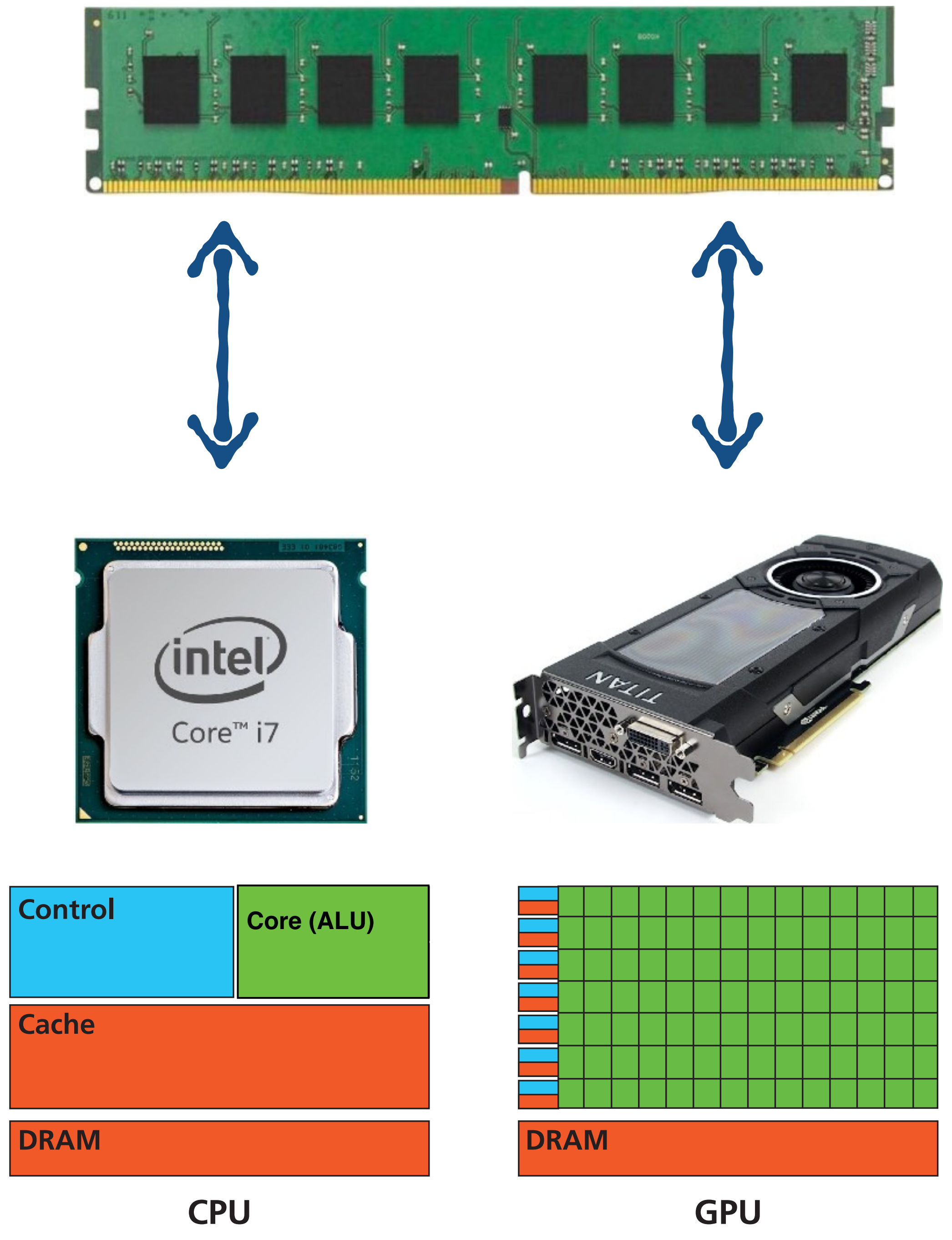
- Lecture: Performance limiters and evaluation

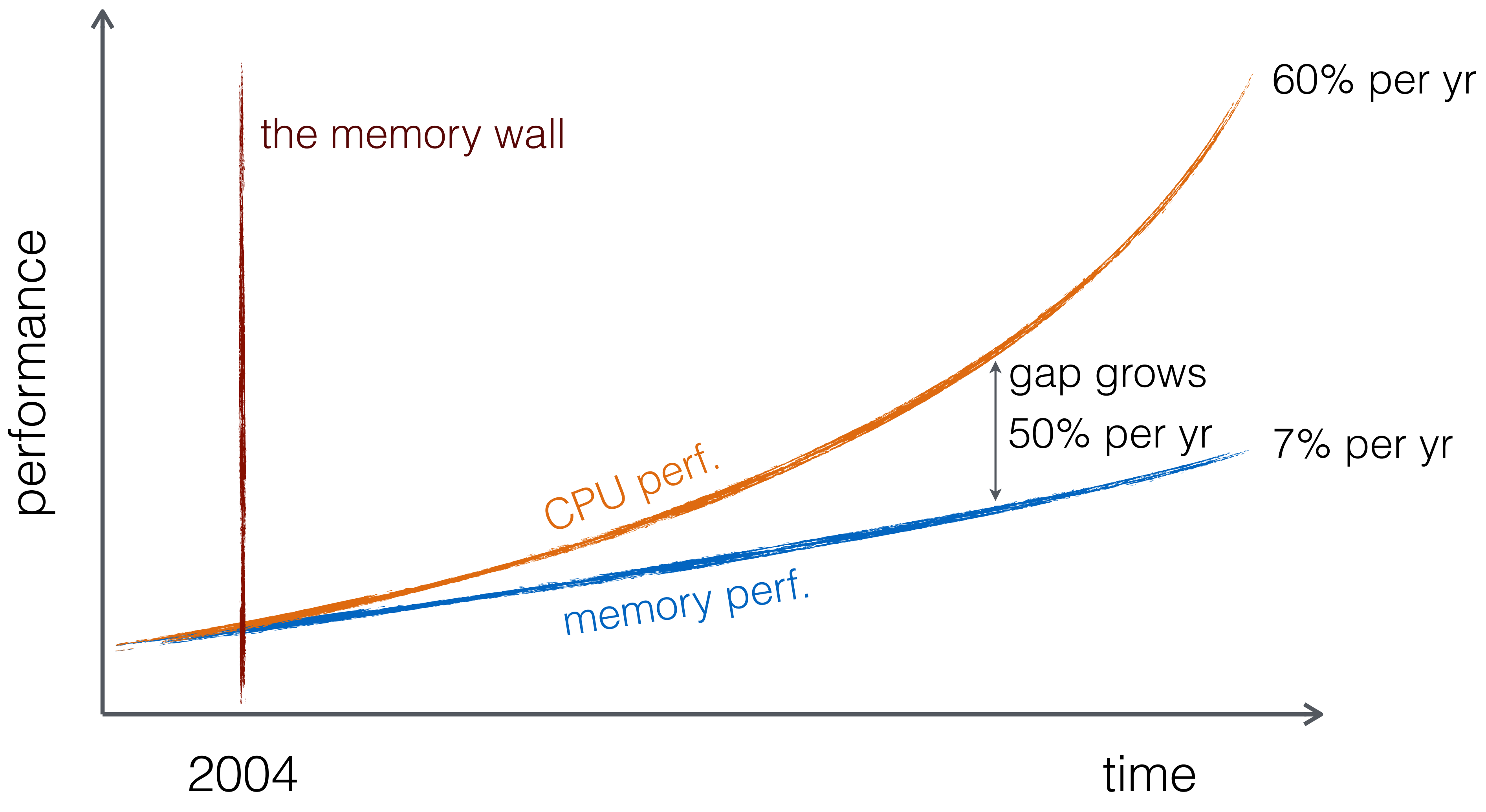
- Programming: 1/ Evaluate memory copy throughput
  2/ Effective memory throughput acoustic 2D

- Tasks: 1/ Evaluate performance of your code
  2/ Produce a graph similar to shown today (to be included in your project)
  3/ Work on your Elastic wave 3D or Stokes 3D

Performance limiters

- Data transfers
- Computations



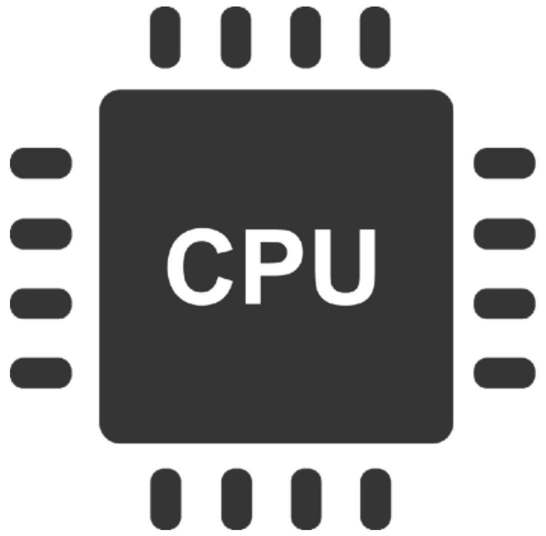


Performance limiters

3D diffusion	3D acoustic wave
$\frac{\partial T}{\partial t} = -\left(\frac{\partial q_x}{\partial x} + \frac{\partial q_y}{\partial y} + \frac{\partial q_z}{\partial z}\right)$ $q_x = -D\frac{\partial T}{\partial x}$ $q_y = -D\frac{\partial T}{\partial y}$ $q_z = -D\frac{\partial T}{\partial z}$	$\frac{1}{k}\frac{\partial P}{\partial t} = -\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}\right)$ $\rho\frac{\partial v_x}{\partial t} = -\frac{\partial P}{\partial x}$ $\rho\frac{\partial v_y}{\partial t} = -\frac{\partial P}{\partial y}$ $\rho\frac{\partial v_z}{\partial t} = -\frac{\partial P}{\partial z}$
1 variable read + write (T)	4 variables read + write (P, Vx, Vy, Vz)

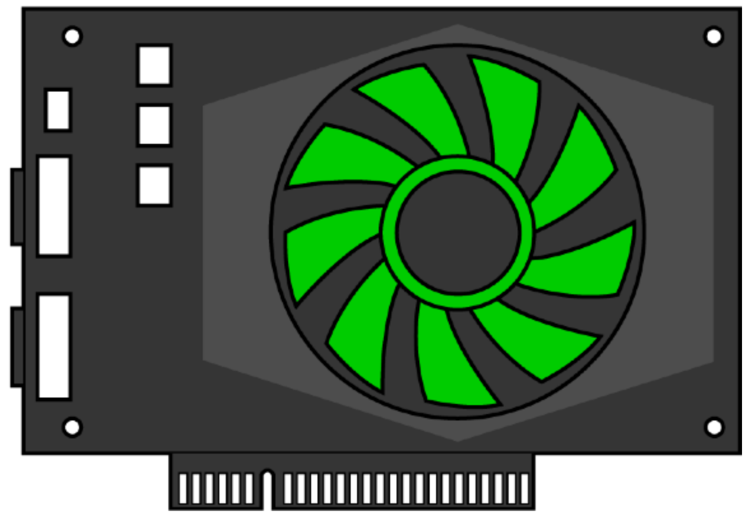
Performance limiters

- Arithmetic intensity



500 GFlop/s
50 GB/s
ratio ~10

Tesla V100



15’000 GFlop/s
700 GB/s
ratio ~21

- Flops / bytes ratio: compute or memory bound

- /!\ bytes are not numbers: double precision -> 8 bytes per read or write
  Example: 1st derivative:

$$\frac{\partial A}{\partial x} \approx \frac{A(ix+1) - A(ix)}{\Delta x}$$

2 reads, 1 write
24 bytes transferred

Performance limiters

	$\frac{\partial}{\partial x}$	Diffusion	Acoustic	Tesla V100
# flops	2	16	22	15.7 x 10^{12}
# bytes	24	168	240	0.7 x 10^{12}
Ratio	0.08	0.1	0.09	21

~0.1 (algorithms) << 21 (machine balance)

We are memory bound.

Performance limiters

Optimality of data access

- PDE solvers are memory bound: computations $ | memory accesses $$$$
- Optimise memory access efficiency (limit it as much as possible):
  - Low memory footprint algorithm
  - Simple and regular data access pattern
- Computations (flops) are for free: recompute fields instead of storing them
- Try to approach memory copy throughput

Effective memory throughput

- Use an effective and absolute metric to measure optimality of data access: MTP_{eff}
- Tells us how far we are from ideal: compare to MTP_{peak} (memcopy only - no flops)
- Optimise memory access: touch every variable once (ideally) - do a minimal amount of read/write
- Count only the minimal # of memory transfers / iteration in MTP_{eff}
- Try not to read neighbours twice
- Do not count neighbours in the MTP metric !

Effective memory throughput

- Use an effective and absolute metric to measure optimality of data access: MTP_{eff}

$$\boxed{MTP_{\text{eff}} = \frac{n_{\text{RW}}\, n_i^{\text{tot}}\, n_{\text{precis}}}{2^{30}\, t_{\text{elapsed}}} \quad [\text{GB/s}]}$$

$$\begin{aligned}
n_{\text{RW}} &= 2 \times (\text{read and write}) + \text{read only fields} \\
n_i^{\text{tot}} &= n_x \times n_y \times n_z \times n_{\text{t}} \\
n_{\text{precis}} &= \text{word size [bytes]} \\
t_{\text{elapsed}} &= \text{elapsed time [sec]}
\end{aligned}$$

MTP_{eff} = lower bound of required memory transfers / time per iteration

≠

$MTP_{profiler}$ = performed memory transfers / time per iteration

Session 3 - Performance evaluation

Today's agenda

- Lecture: Performance limiters and evaluation

- Programming: 1/ Evaluate memory copy throughput
  2/ Effective memory throughput acoustic 2D

- Tasks: 1/ Evaluate performance of your code
  2/ Produce a graph similar to shown today (to be included in your project)
  3/ Work on your Elastic wave 3D or Stokes 3D

1/ Evaluate memory copy throughput

- Add timer to the code

```
// Timer
#include "sys/time.h"
double timer_start = 0;
double cpu_sec(){ struct timeval tp; gettimeofday(&tp,NULL); return tp.tv_sec+1e-6*tp.tv_usec; }
void   tic(){ timer_start = cpu_sec(); }
double toc(){ return cpu_sec()-timer_start; }
void   tim(const char *what, double n){ double s=toc();
       if(me==0){ printf("%s: %8.3f seconds",what,s);if(n>0)printf(", %8.3f GB/s", n/s); printf("\n"); } }
```

- In main

```
size_t N=nx*ny, mem=N*sizeof(DAT);

tim("Time (s), Effective MTP (GB/s) = ", mem*(nt-3)*4/1024./1024./1024.);
```

1/ Evaluate memory copy throughput

MTP code:

- Memory copy only
- A = B + 1 or A = A + 1
- +1 needed other wise compiler is smart enough to only swap pointers
- 1D sufficient, but needs very large array -> saturate bandwidth
- Do 3 warmup iterations

1/ Evaluate memory copy throughput

- Effective memory copy throughput on Nvidia Tesla V100 PCIe 16 GB

  MTP effective: 733 GB/s

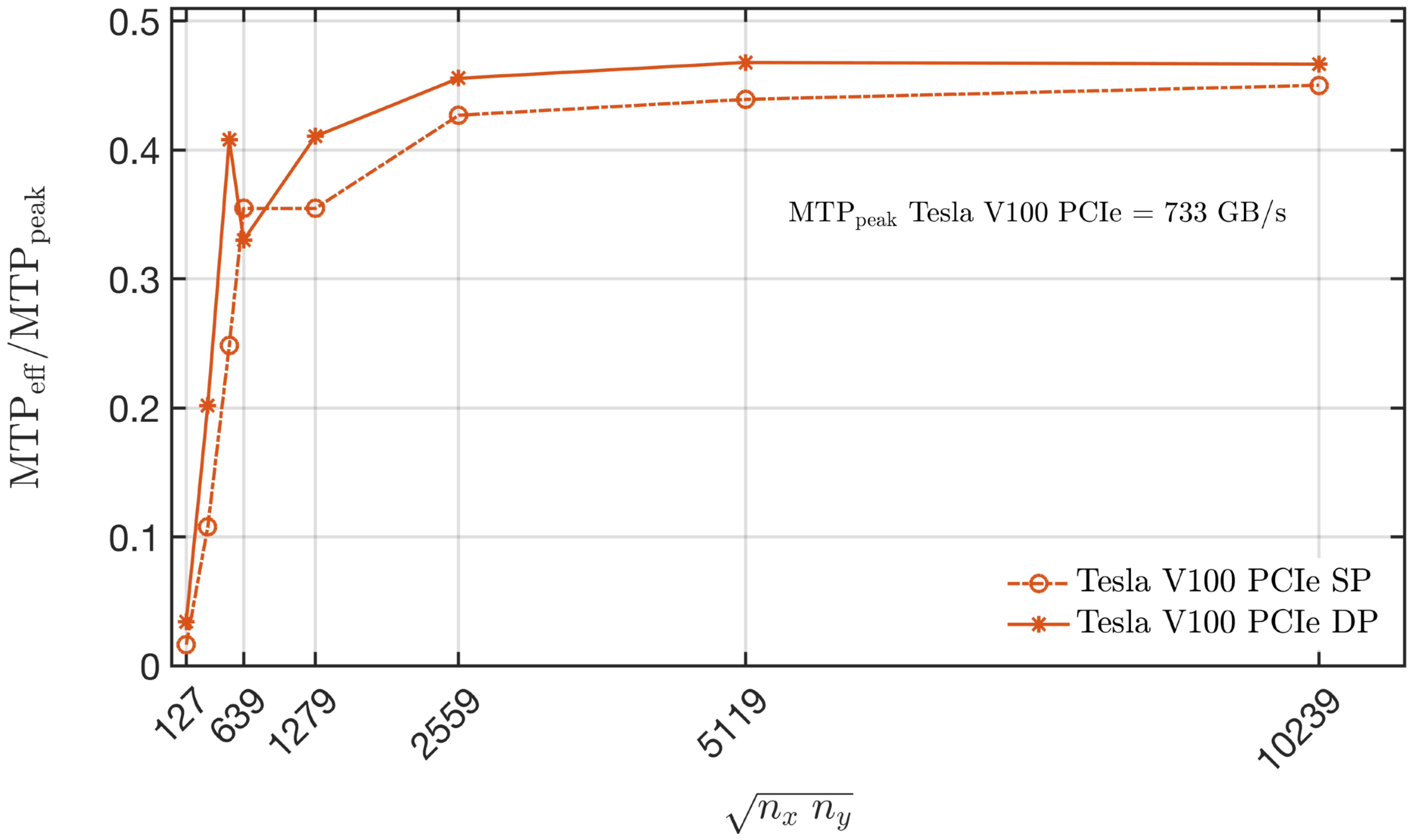
- Fastest possible memory transfer, no computations !

- This should be the reference for evaluating the performance of the stencil code, i.e. memory copy with derivatives = reading neighbouring values.

2/ Effective memory throughput acoustic 2D

- Acoustic 2D # read / writes = 4
- Effective MTP
  Double precision: **343 GB/s**
  Single precision: **330 GB/s**
- /!\ multiplication and division order has an impact !
- Prefer multiplication to division
- See version a,b,c for results

2/ Effective memory throughput acoustic 2D



Session 3 - Performance evaluation

Today's agenda

- Lecture: Performance limiters and evaluation

- Programming: 1/ Evaluate memory copy throughput
  2/ Effective memory throughput acoustic 2D

- Tasks: 1/ Evaluate performance of your code
  2/ Produce a graph similar to shown today (to be included in your project)
  3/ Work on your Elastic wave 3D or Stokes 3D

Outlook - Session 4

- Topic: Accelerating iterative methods - Stokes
- Programming: Fast iterative incompressible Stokes flow
- Tasks: 2D and 3D elastic waves or viscous Stokes

- Discussion on projects and overall Q&A.

Suggested references

- Performance of stencil codes + MPI

  https://on-demand.gputechconf.com/gtc/2019/video/_/S9368/

- Iterative method for solving large 3D problems on GPUs

  http://www.nature.com/articles/s41598-018-29485-5

  https://doi.org/10.1093/gji/ggz239

  https://doi.org/10.1093/gji/ggy434

That's it for today

- wp.unil.ch/geocomputing/
- lraess@stanford.edu

SIGMA