Stanford University

CME 253A

Introduction to High Performance Computing and Parallel (GPU) Computing

---

Stanford Summer Session 1

24 June 2019 | Y2E2 111

Ludovic Räss

Stanford University, Department of Geophysics

lraess@stanford.edu

SIGMA

CME 235A - Course information

Objectives | design a 3D parallel GPU Stokes / Elastic wave solver

- Resources
- Class schedule
- Course format
- Assignments
- Final project

CME 235A - Course information

Resources

- Online resources:
  Stanford canvas: https://canvas.stanford.edu/courses/103823

- Course material (slides and codes) will be uploaded to canvas prior (if needed) and after (in general) each session.

- Do not hesitate to contact me if needed
  email: lraess@stanford.edu
  Slack: Ludovic Räss

CME 235A - Course information

Class schedule

- We will meet for 5 sessions:
  - Monday June 24 > Introduction + Matlab examples
  - Wednesday June 26 > GPU computing
  - Monday July 1 > Performance evaluation
  - Wednesday July 3 > Projects: 3D Elastic waves or Stokes flow
  - Monday July 15 > Final discussion
- 3:30pm to 5:20pm
- Location: Y2E2 111

CME 235A - Course information

Course format

- This course is a “hands-on” and is therefore interactive and dynamic
- General session structure: 1/ introduction to new material or next steps
  2/ quiet working time where you can ask for help/questions
- The assignment will be determined before the end of each session

CME 235A - Course information

Assignments

- Finalise the material and step we discussed in class
- Assignments will be given by the end of each session for the next one
- If you have to leave earlier or cannot attend a session, information will be available on Canvas or contact me via email (lraess@stanford.edu)

CME 235A - Course information

Final project

- Objective: 3D GPU accelerated Stokes / elastic wave propagation solver
- Hand in: 1a/ Short presentation (few slides) about project [preferred]
  (motivation, results, your background, what you learned, what was difficult)
  1b/ Concise report including similar material as a presentation [alternative]
  2 / Commented codes ready to run and produce a figure
- During the last session (or part of it) we will have time for a short presentations of everyone's project results in short presentation format

CME 235A - Course information

Technicalities and computing resources

- Used software: Matlab (Octave or Python are also an option)
  CUDA + C

- Laptop with Unix shell (native in Linux and macOS, install Putty under windows)

- Optional sftp file transfer software (Filezilla, cyberduck, Transmit, …)

- Account on CEES mazama GPU servers

Please give me your SUNet ID if you are attending the class
but are not registered in Stanford Canvas - for GPU cluster account

CME 235A - Course information

Questions so far ?

CME 253A

Introduction to High Performance Computing and Parallel (GPU) Computing

Stanford Summer Session 1

24 June 2019 | Y2E2 111

Ludovic Räss

Stanford University, Department of Geophysics

lraess@stanford.edu

SIGMA

Session 1 - PDEs

Today's agenda

- Lecture: Introduction - scientific GPU computing

- Programming: 1/ diffusion and acoustic wave propagation in 1D
  2/ loop vs vectorised approach

- Tasks: 1/ from 1D to 2D and 3D
  2/ 2D acoustic to elastic
  3/ 2D elastic to viscous
  4/ 2D viscous vectorised to loop

Session 1 - PDEs

Today's agenda

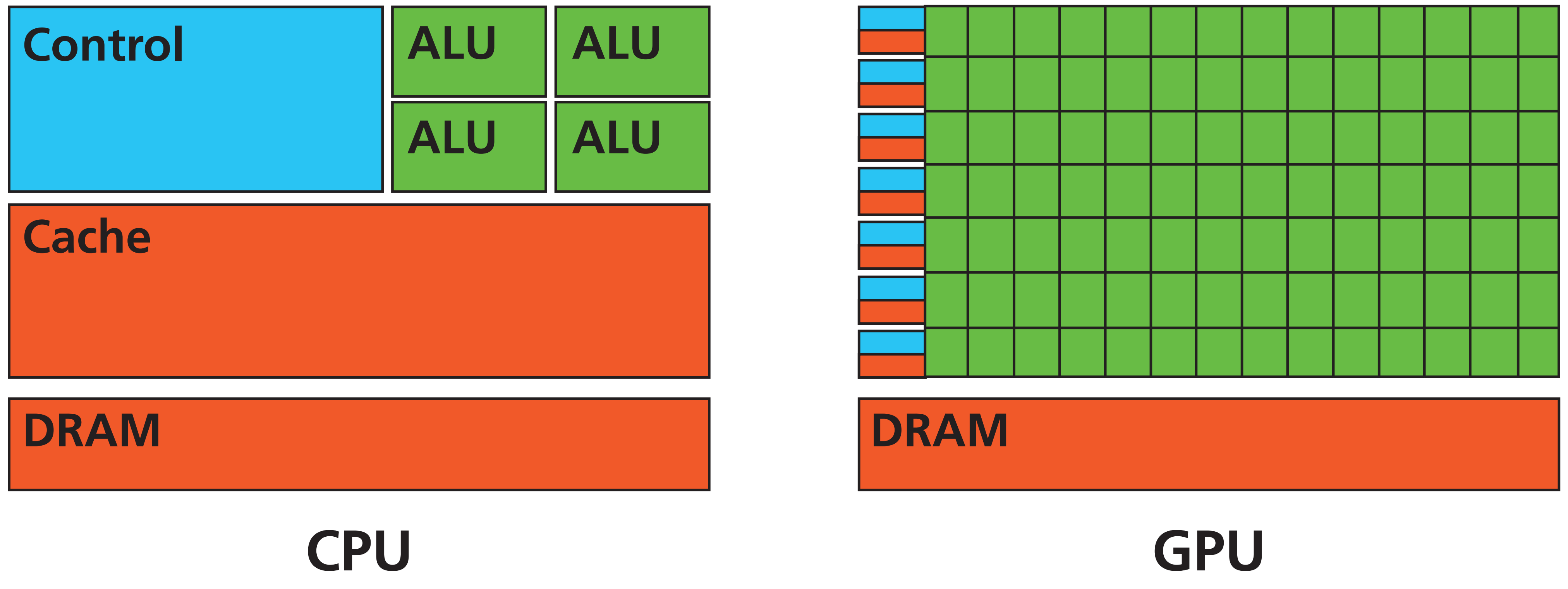
- Lecture: Introduction - scientific GPU computing

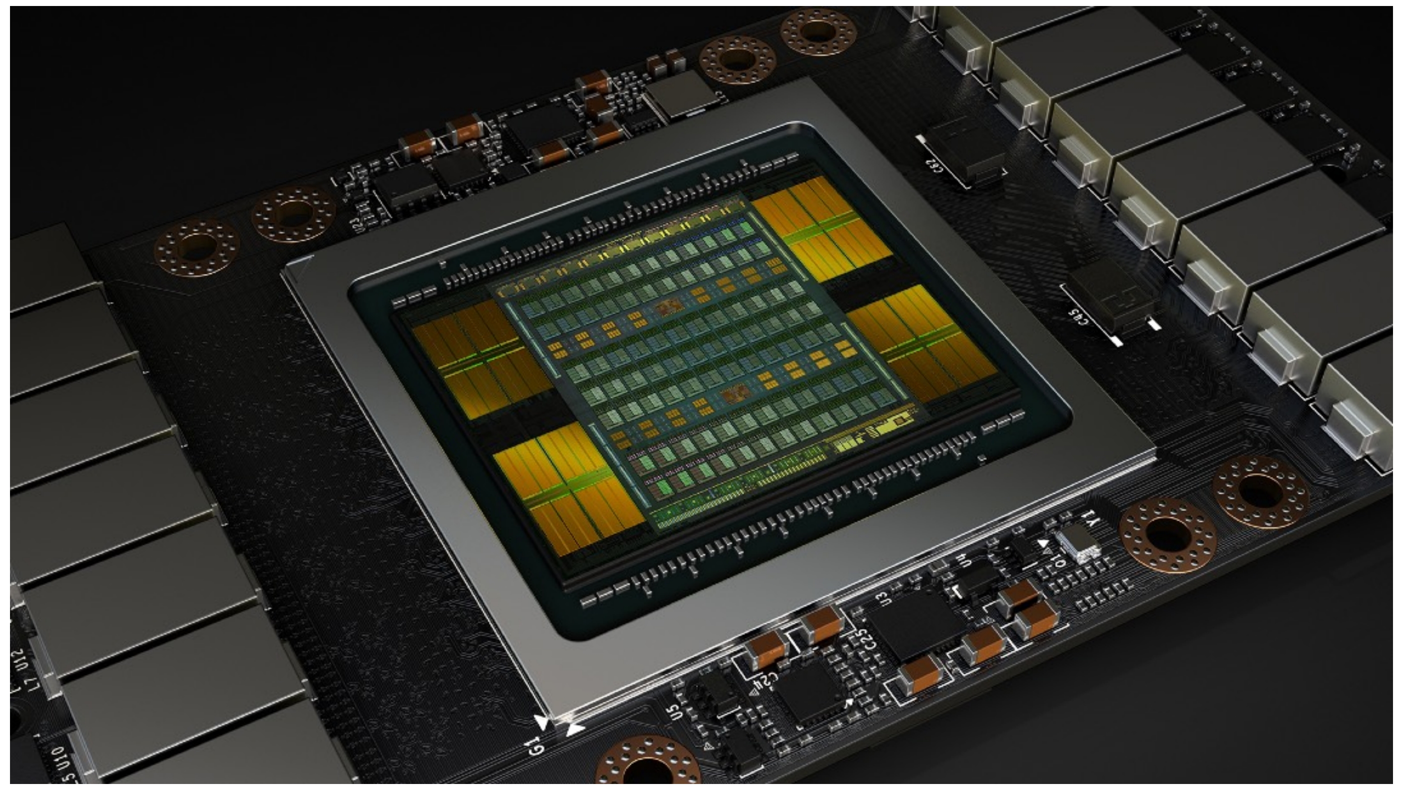
- Programming: 1/ diffusion and acoustic wave propagation in 1D
  2/ loop vs vectorised approach

- Tasks: 1/ from 1D to 2D and 3D
  2/ 2D acoustic to elastic
  3/ 2D elastic to viscous
  4/ 2D viscous vectorised to loop

What is a GPU

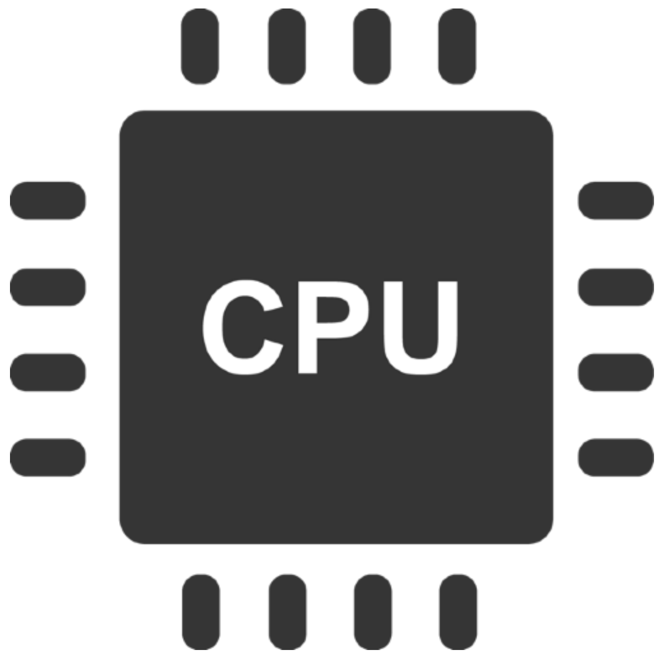
- Graphical Processing Unit (to paint the screen)
- Many small cores all doing the same job
- Large chip, close to memory
- High memory bandwidth - important for PDEs !





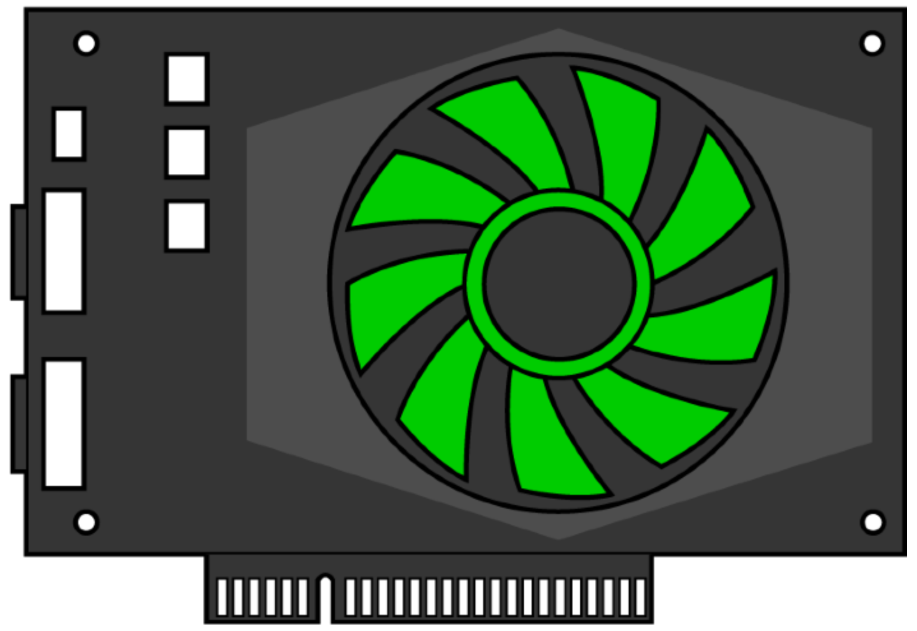
Why to use a GPU

- Serial vs parallel execution > loop vs vectorised code
- Simultaneous calculations > increased concurrency
- Order of magnitude higher memory bandwidth & flop/s count



0.5 TFlop/s

50 GB/s



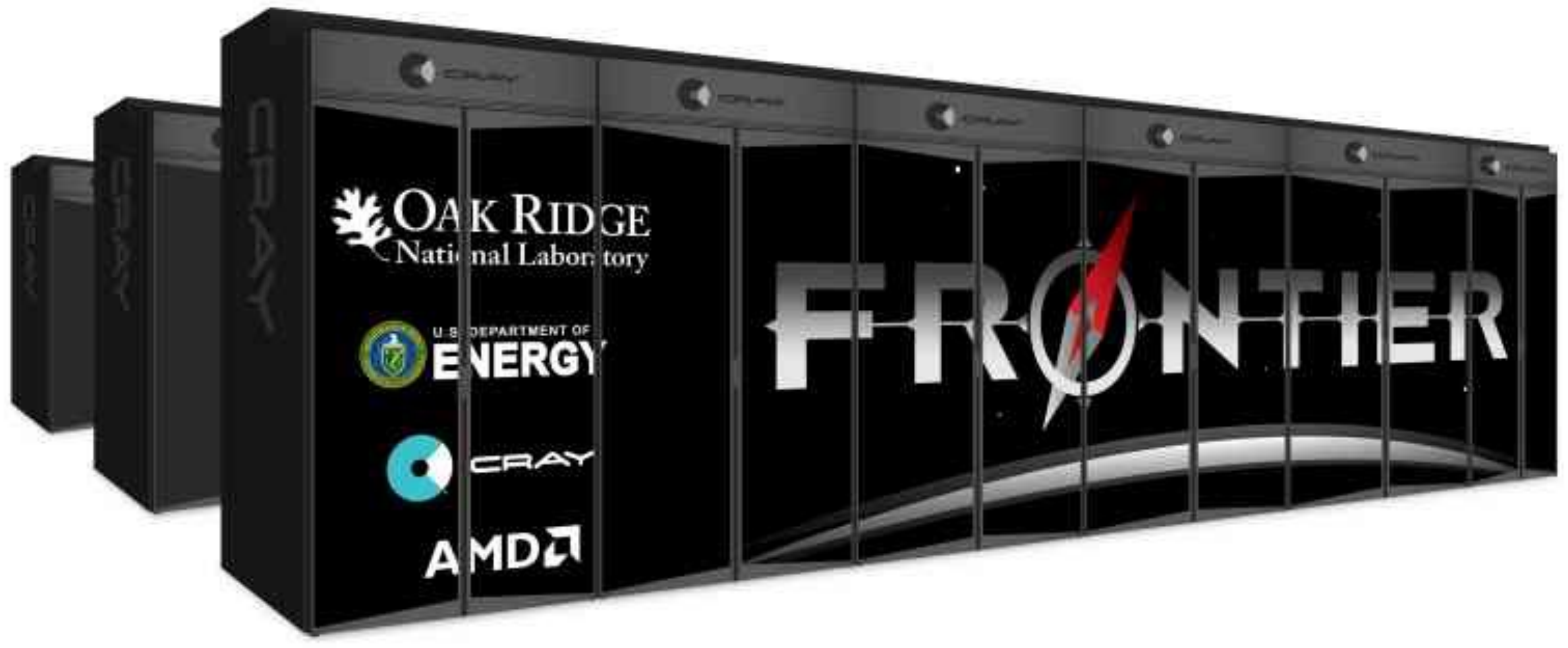
15 TFlop/s

700 GB/s

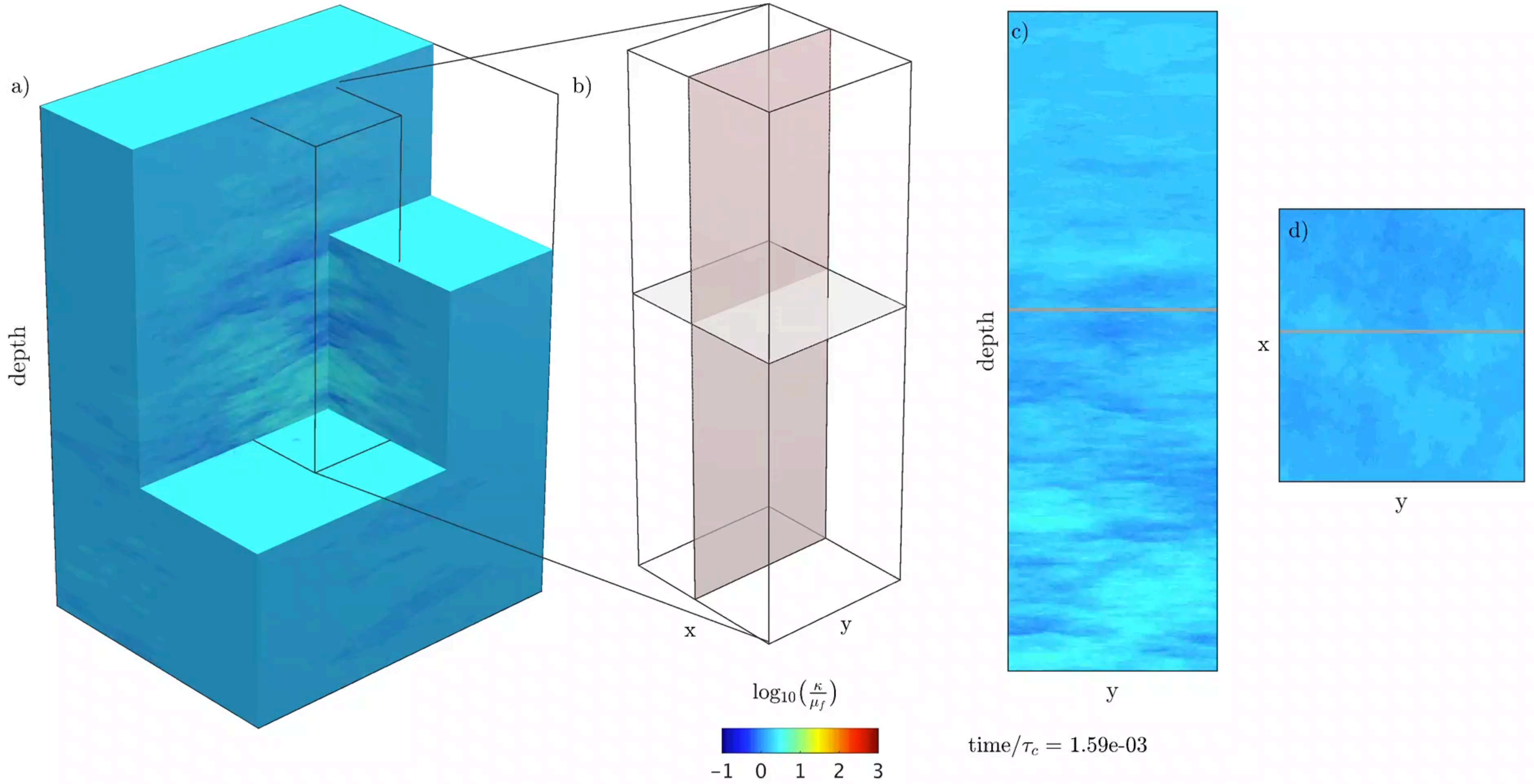
- Supercomputing available on your desk !

Why to use a GPU

- HPC and supercomputing on your desktop
- World’s fasted supercomputer Summit features 27’648 Nvidia V100 GPUs
- Future exascale machines all feature GPUs (Intel + AMD)



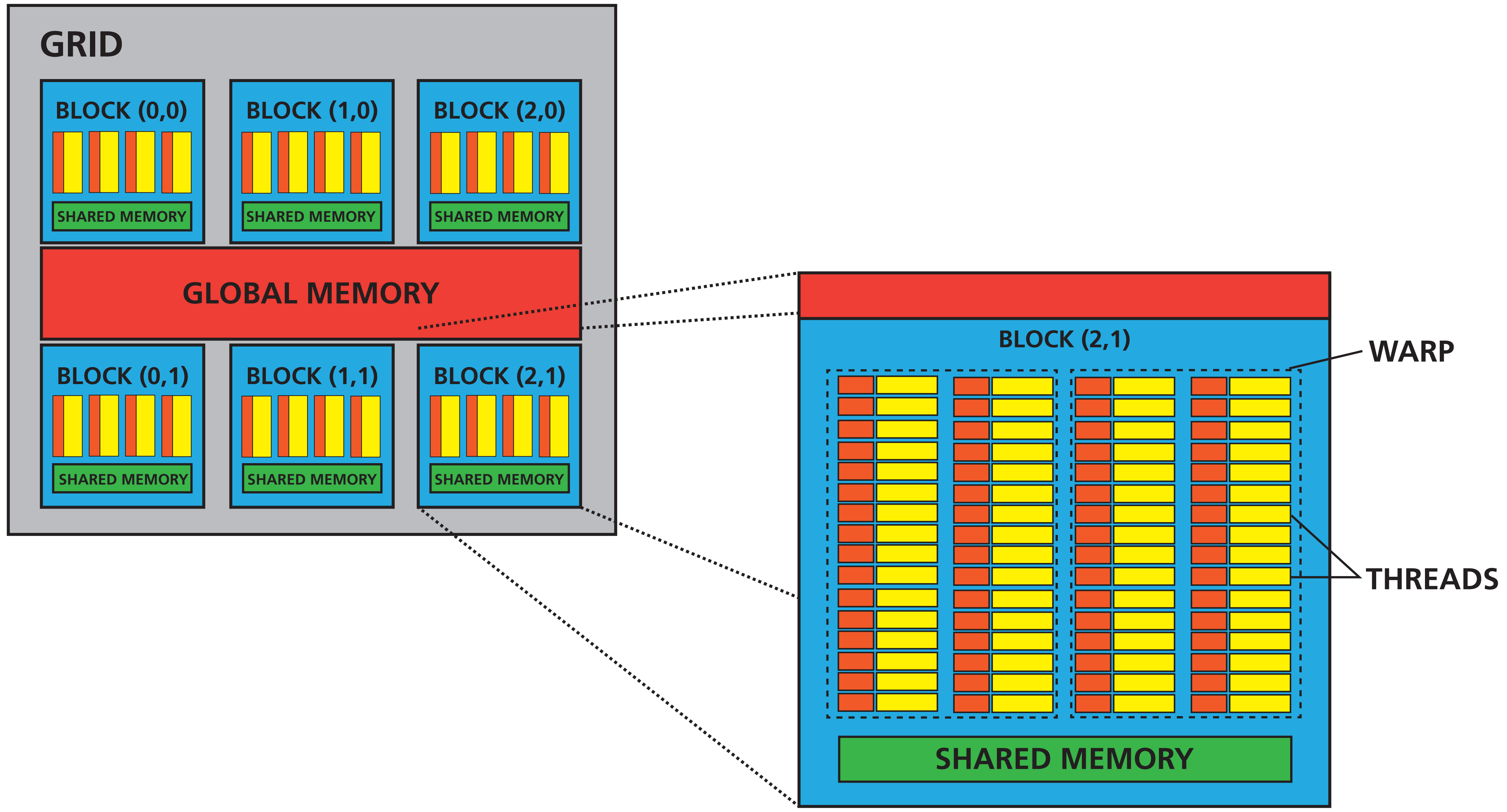

Subsurface reservoirs
Localised buoyancy-driven flow
Geophysical imaging
ms
INT
ghmark
H50
Truncations
H60
H70
Chimney
Gas laye
Extraction and storage
Løseth et al., 2011
Plaza-Faverola et al., 2011
© Statoil



Räss et al., 2018. Nature Scientific Reports

Programming the GPU: CUDA

- CUDA: provides vectorised indexes to C / C++ for parallel execution
- Building elements: `thread, block, grid`
- Max. 1024 threads per block !
- Assign one thread per grid point
- All threads read the same code
  loop bounds > if statements



Session 1 - PDEs

Today's agenda

- Lecture: Introduction - scientific GPU computing

- Programming: 1/ diffusion and acoustic wave propagation in 1D
  2/ loop vs vectorised approach

- Tasks: 1/ from 1D to 2D and 3D
  2/ 2D acoustic to elastic
  3/ 2D elastic to viscous
  4/ 2D viscous vectorised to loop

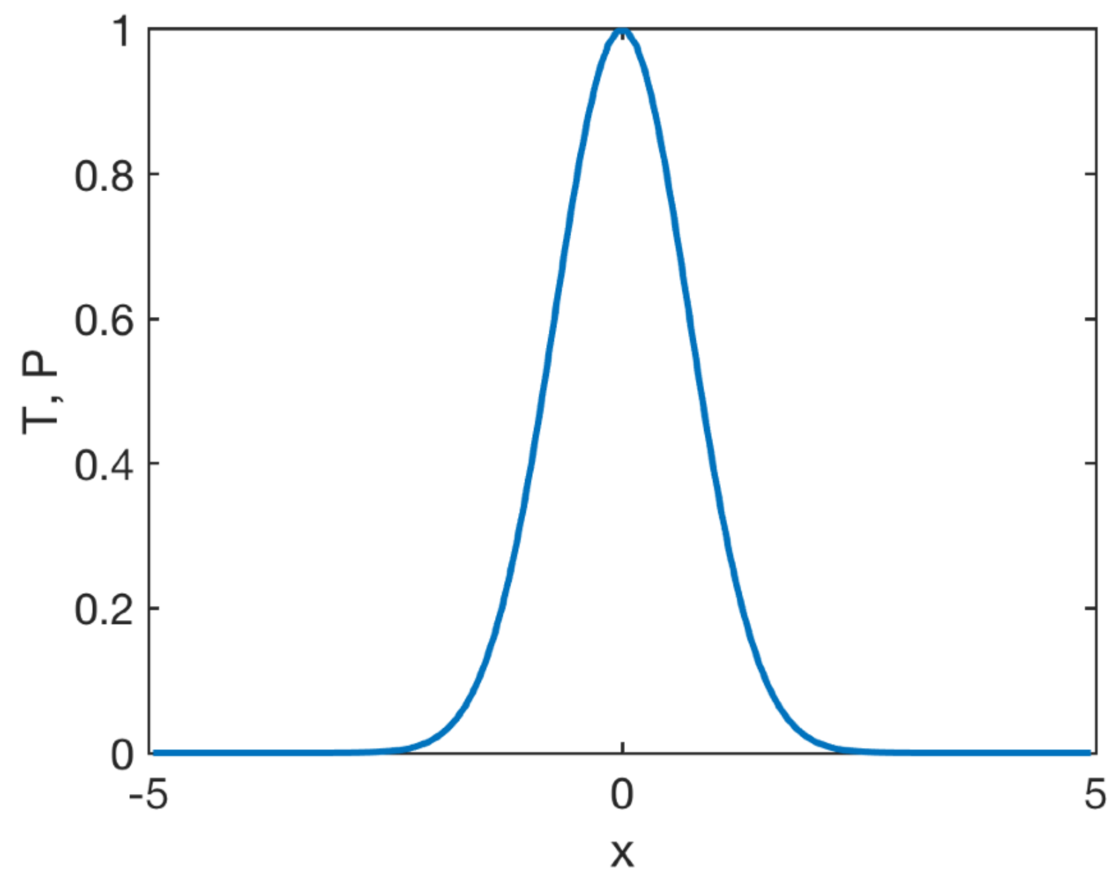
1/ diffusion and acoustic wave propagation in 1D

Diffusion	Acoustic wave
$\frac{\partial T}{\partial t} = -\frac{\partial q_x}{\partial x}$	$\frac{1}{k}\frac{\partial P}{\partial t} = -\frac{\partial v_x}{\partial x}$
$q_x = -D\frac{\partial T}{\partial x}$	$\rho\frac{\partial v_x}{\partial t} = -\frac{\partial P}{\partial x}$

```
% Physics
Lx = 10;
D  = 1;
% Numerics
nx = 100;
dx = Lx/nx;
nt = 200;
dt = dx^2/D/2.1;
x  = (-Lx+dx)/2:dx:(Lx-dx)/2;
% Initial conditions
T  = exp(-x.^2);
```

```
% Physics
Lx  = 10;
k   = 1;
rho = 1;
% Numerics
nx  = 100;
dx  = Lx/nx;
nt  = 200;
dt  = dx/sqrt(k/rho)/2.1;
x   = (-Lx+dx)/2:dx:(Lx-dx)/2;
% Initial conditions
P   = exp(-x.^2);
V   = [0*P 0];
```

Initial gaussian perturbation



1/ diffusion and acoustic wave propagation in 1D

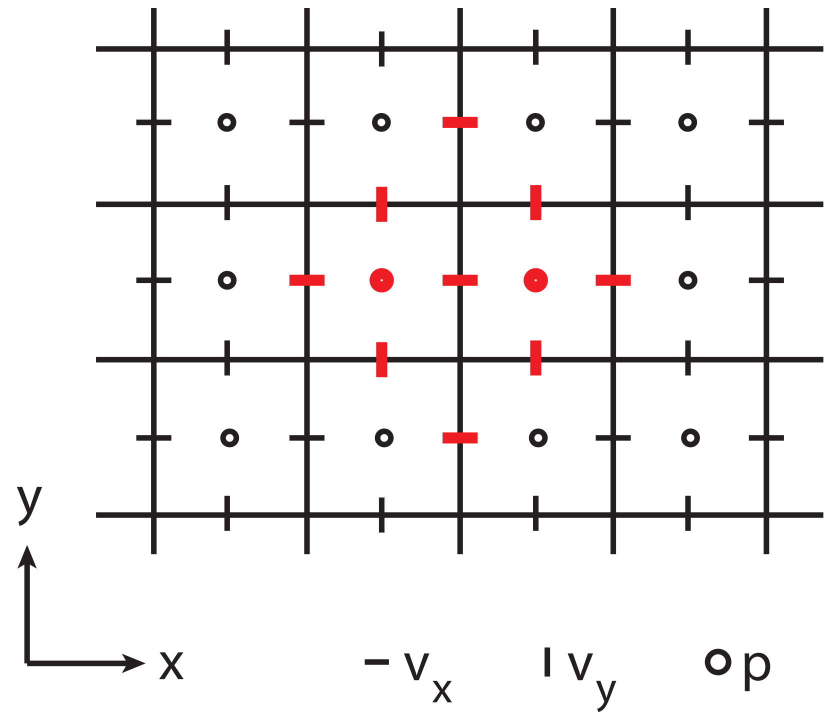
- Finite-difference method

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

- CFL time step stability for diffusion and wave propagation

$$\mathrm{CFL}_{\mathrm{diff}} = \frac{\min(\Delta x_i)^2}{D\ 2.1 n_{\mathrm{dim}}} \qquad \mathrm{CFL}_{\mathrm{acoust}} = \frac{\min(\Delta x_i)}{\sqrt{k/rho}\ 2.1}$$

- Staggered grid for computation



2/ loop vs vectorised approach

- Replace all vectorised statements by for loops

Vectorised version	Loop version

`V(2:end-1) = V(2:end-1) - dt*diff(P)/dx/rho;`	`for ix = 2:nx`
	`    V(ix) = V(ix) - dt*(P(ix)-P(ix-1))/dx/rho;`
	`end`

Session 1 - PDEs

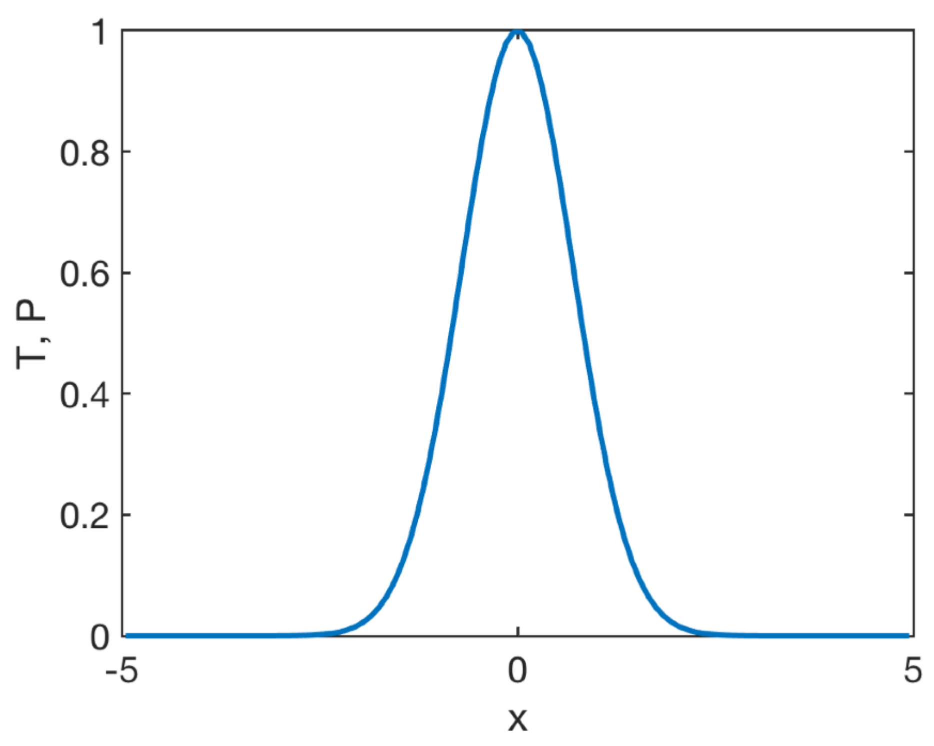
Today's agenda

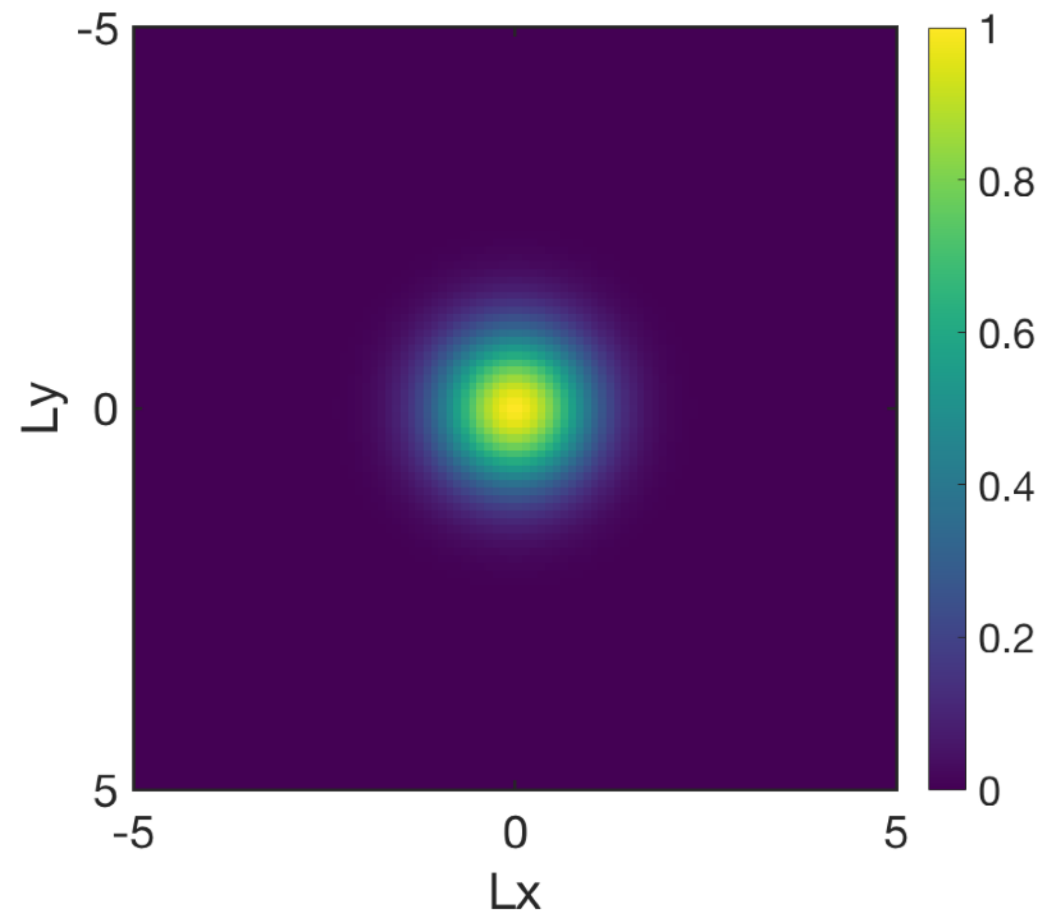
- Lecture: Introduction - scientific GPU computing
- Programming: 1/ diffusion and acoustic wave propagation in 1D
  2/ loop vs vectorised approach
- Tasks: 1/ from 1D to 2D and 3D
  2/ 2D acoustic to elastic
  3/ 2D elastic to viscous
  4/ 2D viscous vectorised to loop

Task 1/ from 1D to 2D

	Diffusion	Acoustic wave
1D	$$\frac{\partial T}{\partial t} = -\frac{\partial q_x}{\partial x}$$ $$q_x = -D\frac{\partial T}{\partial x}$$	$$\frac{1}{k}\frac{\partial P}{\partial t} = -\frac{\partial v_x}{\partial x}$$ $$\rho\frac{\partial v_x}{\partial t} = -\frac{\partial P}{\partial x}$$
2D	$$\frac{\partial T}{\partial t} = -\left(\frac{\partial q_x}{\partial x} + \frac{\partial q_y}{\partial y}\right)$$ $$q_x = -D\frac{\partial T}{\partial x}$$ $$q_y = -D\frac{\partial T}{\partial y}$$	$$\frac{1}{k}\frac{\partial P}{\partial t} = -\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y}\right)$$ $$\rho\frac{\partial v_x}{\partial t} = -\frac{\partial P}{\partial x}$$ $$\rho\frac{\partial v_y}{\partial t} = -\frac{\partial P}{\partial y}$$
hint	`[x2,y2] = ndgrid(x,y);`	`[x2,y2] = ndgrid(x,y);`

Initial gaussian perturbation





Task 1/ from 2D to 3D

	Diffusion	Acoustic wave
2D	$\frac{\partial T}{\partial t} = -\left(\frac{\partial q_x}{\partial x} + \frac{\partial q_y}{\partial y}\right)$ $q_x = -D\frac{\partial T}{\partial x}$ $q_y = -D\frac{\partial T}{\partial y}$	$\frac{1}{k}\frac{\partial P}{\partial t} = -\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y}\right)$ $\rho\frac{\partial v_x}{\partial t} = -\frac{\partial P}{\partial x}$ $\rho\frac{\partial v_y}{\partial t} = -\frac{\partial P}{\partial y}$
3D	$\frac{\partial T}{\partial t} = -\left(\frac{\partial q_x}{\partial x} + \frac{\partial q_y}{\partial y} + \frac{\partial q_z}{\partial z}\right)$ $q_x = -D\frac{\partial T}{\partial x}$ $q_y = -D\frac{\partial T}{\partial y}$ $q_z = -D\frac{\partial T}{\partial z}$	$\frac{1}{k}\frac{\partial P}{\partial t} = -\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}\right)$ $\rho\frac{\partial v_x}{\partial t} = -\frac{\partial P}{\partial x}$ $\rho\frac{\partial v_y}{\partial t} = -\frac{\partial P}{\partial y}$ $\rho\frac{\partial v_z}{\partial t} = -\frac{\partial P}{\partial z}$

Task 2/ 2D acoustic to elastic

Acoustic waves	Elastic waves
$\frac{1}{k}\frac{\partial P}{\partial t} = -\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y}\right)$	$\frac{1}{k}\frac{\partial P}{\partial t} = -\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y}\right)$
$\rho\frac{\partial v_x}{\partial t} = -\frac{\partial P}{\partial x}$	$\rho\frac{\partial v_x}{\partial t} = -\frac{\partial P}{\partial x} + \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y}$
$\rho\frac{\partial v_y}{\partial t} = -\frac{\partial P}{\partial y}$	$\rho\frac{\partial v_y}{\partial t} = -\frac{\partial P}{\partial y} + \frac{\partial \tau_{yy}}{\partial y} + \frac{\partial \tau_{xy}}{\partial x}$
	$\frac{\partial \tau_{xx}}{\partial t} = 2G\left(\frac{\partial v_x}{\partial x} - \frac{1}{3}\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y}\right)\right)$
	$\frac{\partial \tau_{yy}}{\partial t} = 2G\left(\frac{\partial v_y}{\partial y} - \frac{1}{3}\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y}\right)\right)$
	$\frac{\partial \tau_{xy}}{\partial t} = 2G\frac{1}{2}\left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x}\right)$

Task 3/ 2D elastic to viscous

Elastic waves	Viscous flow
$\frac{1}{k}\frac{\partial P}{\partial t} = -\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y}\right)$	$\frac{1}{k}\frac{\partial P}{\partial t} = -\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y}\right)$
$\rho\frac{\partial v_x}{\partial t} = -\frac{\partial P}{\partial x} + \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y}$	$\rho\frac{\partial v_x}{\partial t} = -\frac{\partial P}{\partial x} + \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y}$
$\rho\frac{\partial v_y}{\partial t} = -\frac{\partial P}{\partial y} + \frac{\partial \tau_{yy}}{\partial y} + \frac{\partial \tau_{xy}}{\partial x}$	$\rho\frac{\partial v_y}{\partial t} = -\frac{\partial P}{\partial y} + \frac{\partial \tau_{yy}}{\partial y} + \frac{\partial \tau_{xy}}{\partial x}$
$\frac{\partial \tau_{xx}}{\partial t} = 2G\left(\frac{\partial v_x}{\partial x} - \frac{1}{3}\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y}\right)\right)$	$\tau_{xx} = 2\eta\left(\frac{\partial v_x}{\partial x} - \frac{1}{3}\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y}\right)\right)$
$\frac{\partial \tau_{yy}}{\partial t} = 2G\left(\frac{\partial v_y}{\partial y} - \frac{1}{3}\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y}\right)\right)$	$\tau_{yy} = 2\eta\left(\frac{\partial v_y}{\partial y} - \frac{1}{3}\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y}\right)\right)$
$\frac{\partial \tau_{xy}}{\partial t} = 2G\frac{1}{2}\left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x}\right)$	$\tau_{xy} = 2\eta\frac{1}{2}\left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x}\right)$

Task 4/ 2D viscous vectorised to loop

- Use linear 1D indexing in 2D and 3D:

```
P(ix,iy)     => P(ix+iy*nx)
P(ix,iy,iz) => P(ix+iy*nx+iz*nx*ny)
```

- Check bounds with loop start and end index

Vectorised version	Loop version

`txy = zeros(nx+1,ny+1);`
 	
`txy(2:end-1,2:end-1) = …;`
	`txy = zeros((nx+1)*(ny+1),1);`
	`for iyM=2:ny, iy=iyM-1;`
	`    for ix=2:nx`
	`        txy(ix+iy*(nx+1)) = …;`
	`    end`
	`end`

Outlook - Session 2

- Topic: Intro to GPU computing
- Programming: Wave propagation in 1D and 2D (GPU CUDA C version)
- Tasks: 2D and 3D acoustic, elastic and/or viscous

- Mandatory for session 2: 2D acoustic/elastic/viscous loop code.

Please give me your SUNet ID if you are attending the class
but are not registered in Stanford Canvas - for GPU cluster account

That’s it for today

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