CompBioMed
A Centre of Excellence in Computational Biomedicine

Webinar series

Webinar #3
Lattice Boltzmann method for CompBioMed (incl. Palabos)

19 March 2018

The webinar will start at 12pm CET / 11am GMT

Dr Jonas Latt (Head of Research, University of Geneva)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 675451

The series is run in collaboration with:

VPH Institute
Building the Virtual Physiological Human

CompBioMed
A Centre of Excellence in Computational Biomedicine

Webinar series

Webinar #3
Lattice Boltzmann method for CompBioMed (incl. Palabos)

19 March 2018

Welcome!

Dr Jonas Latt (Head of Research, University of Geneva)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 675451

The series is run in collaboration with:

VPH Institute
Building the Virtual Physiological Human

Today’s topics

- The role of Computational Fluid Dynamics in biomedical modelling.
- Presentation of the Lattice Boltzmann Method.
- Two application examples of our tool Palabos in Computational biomedicine.

Example: treatment of aneurysms

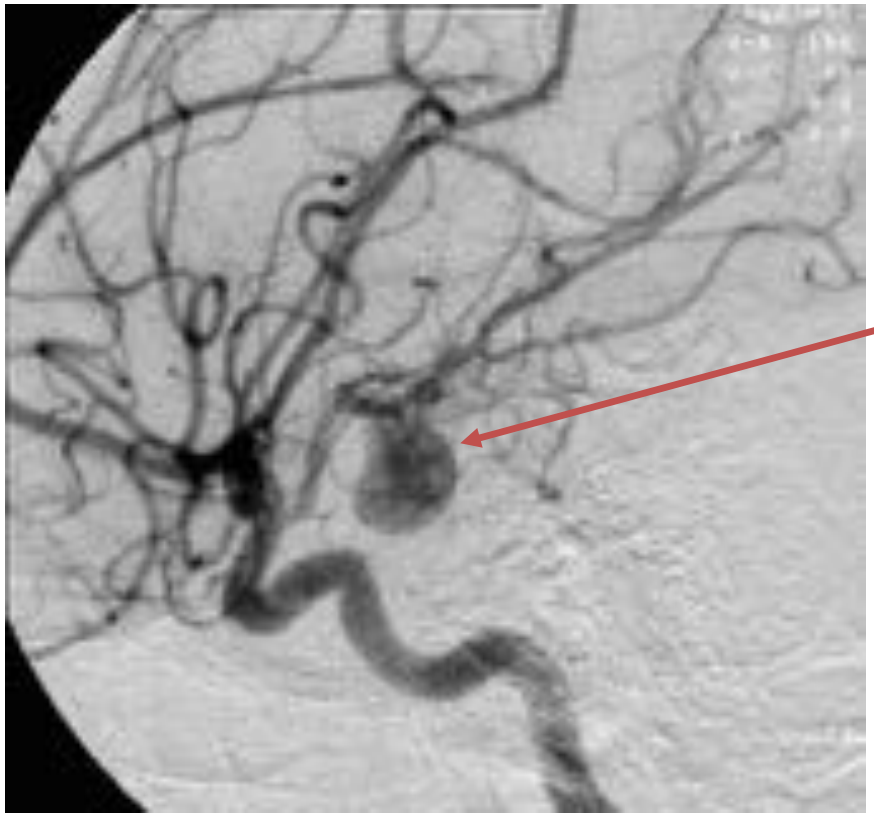
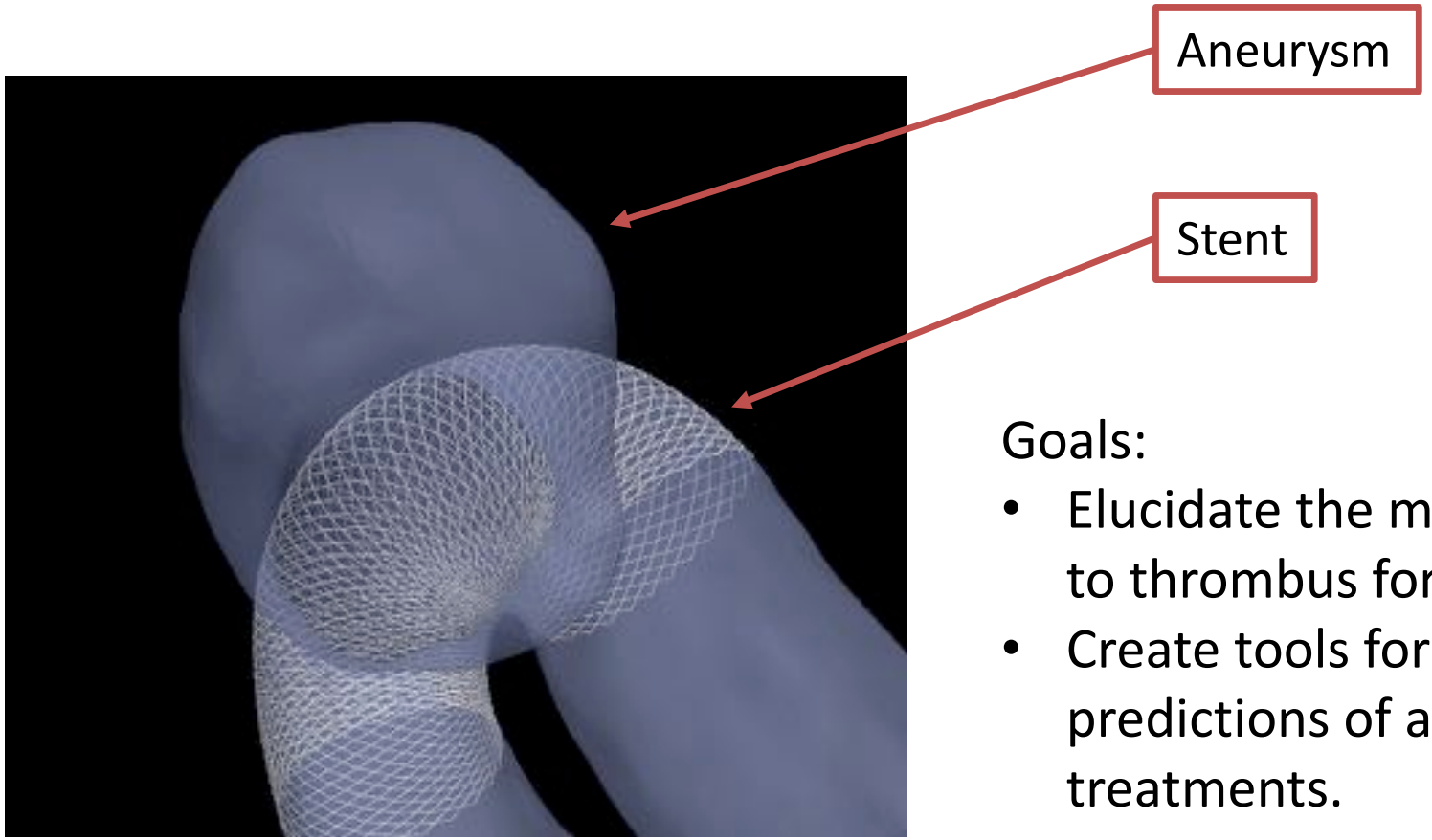


Image by Lucien Monfils

Angiography of an aneurysm in a brain artery.

Flow Diverters



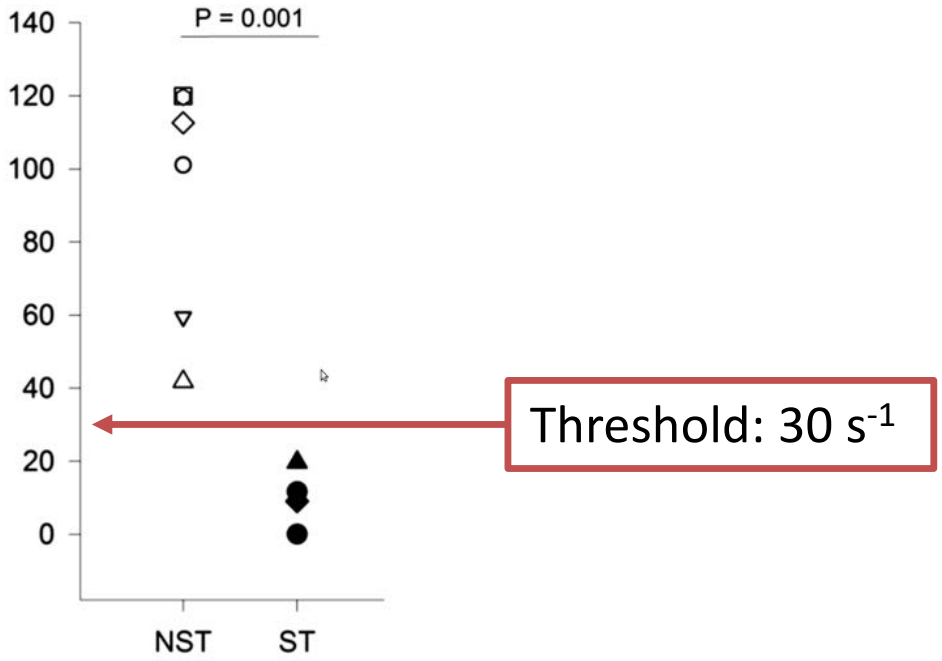
Goals:

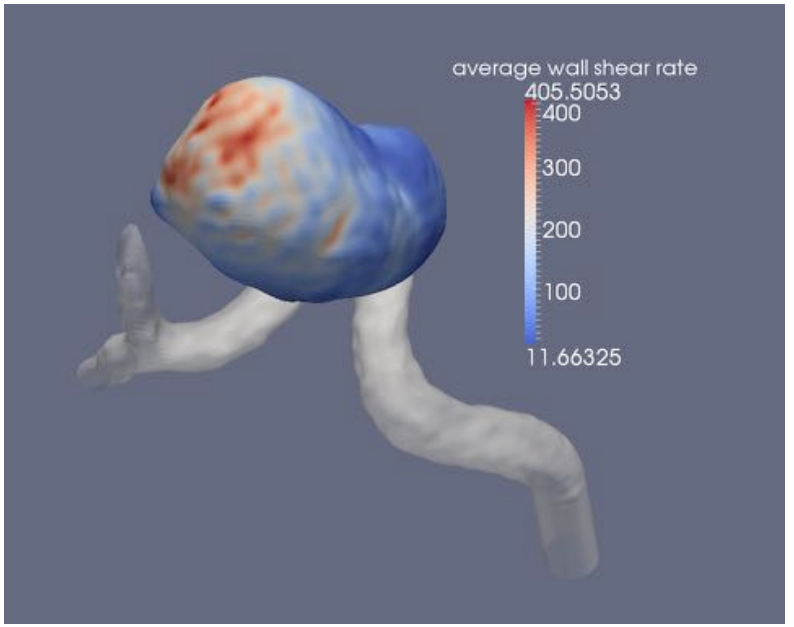
- Elucidate the mechanisms leading to thrombus formation.
- Create tools for patient-specific predictions of aneurysm treatments.

Insights from computer simulations

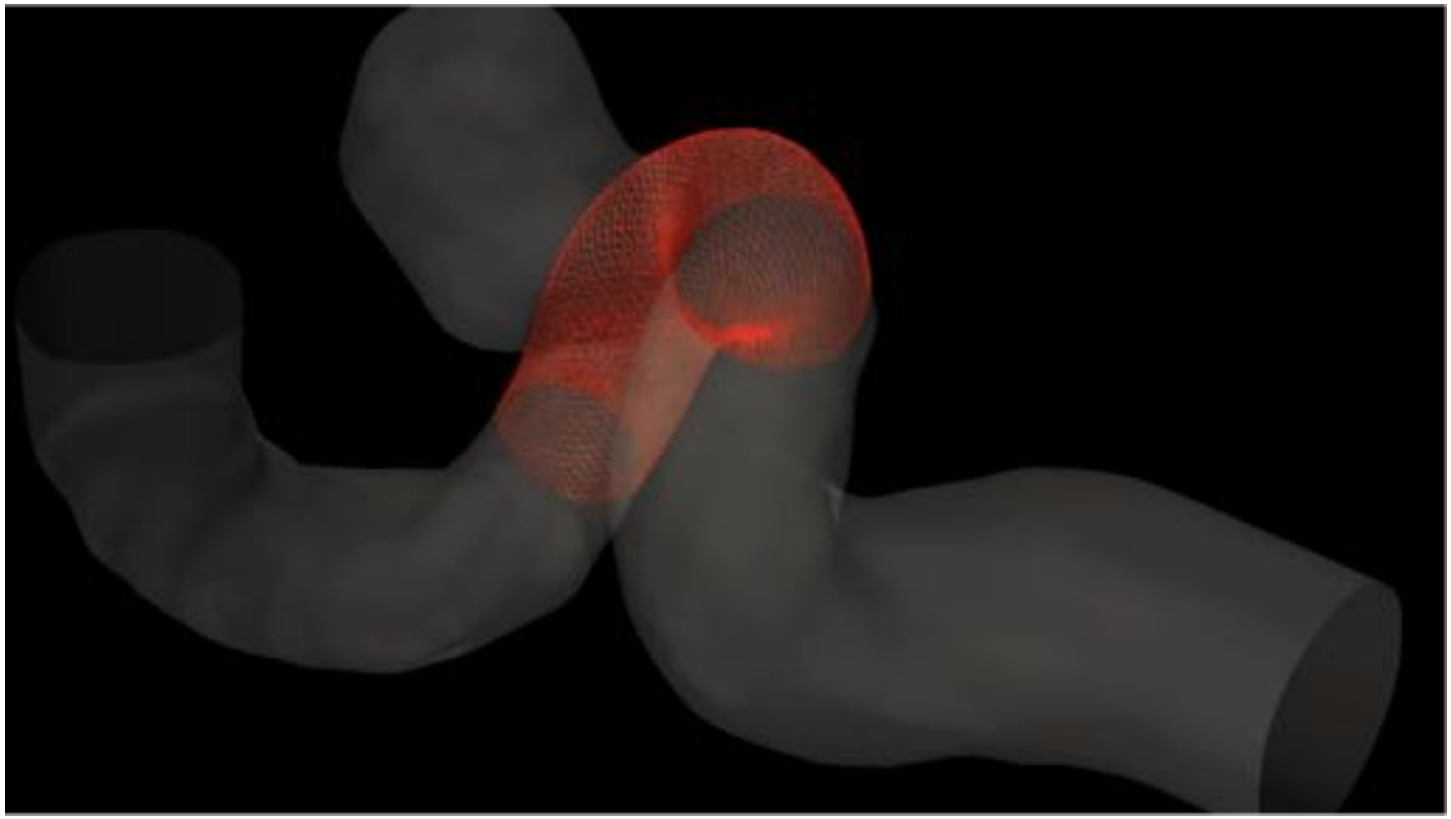
Simulations show aneurysm Wall-Shear-Rate (WSR) deciding factor for thrombus formation.

J NeuroIntervent. Surg. 2015, 011737



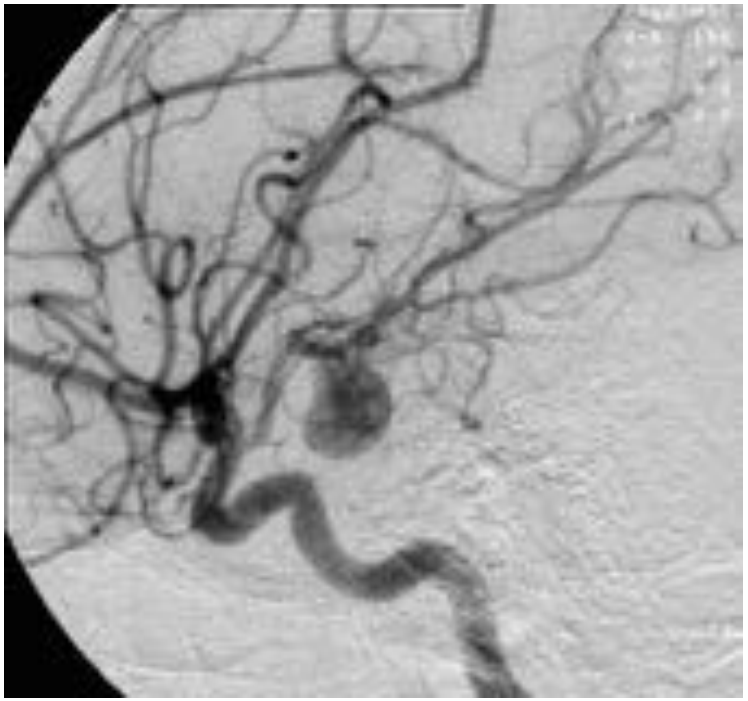


Patient-specific aneurysm treatment



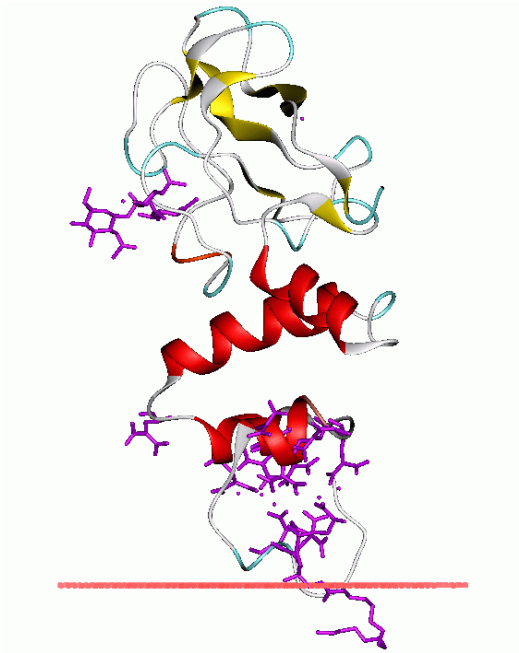
Computational Biomedicine: Elements

1. Medical knowledge



Computational Biomedicine: Elements

2. Expertise in (molecular) biology



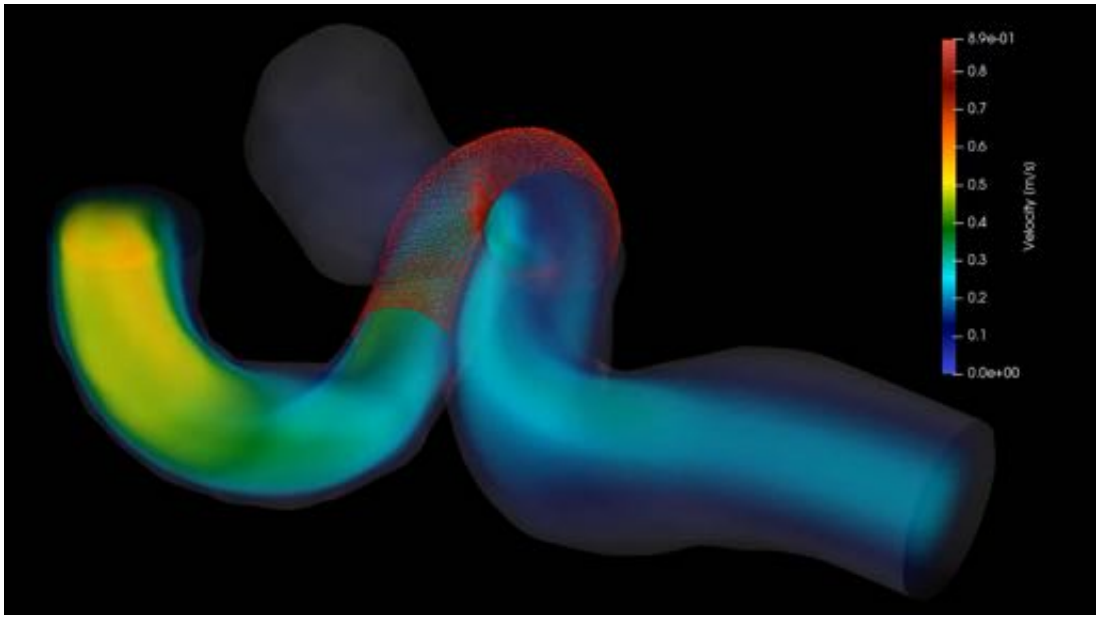
1. Medical Knowledge

Anchoring of bovine
prothrombin to the membrane

Image by Andrei Lomize

Computational Biomedicine: Elements

3. Numerical Modeling



1. Medical Knowledge

2. Expertise in (molecular) biology

Computational Biomedicine: Elements

4. Computer Programming

1. Medical Knowledge

2. Expertise in (molecular) biology

3. Numerical Modeling

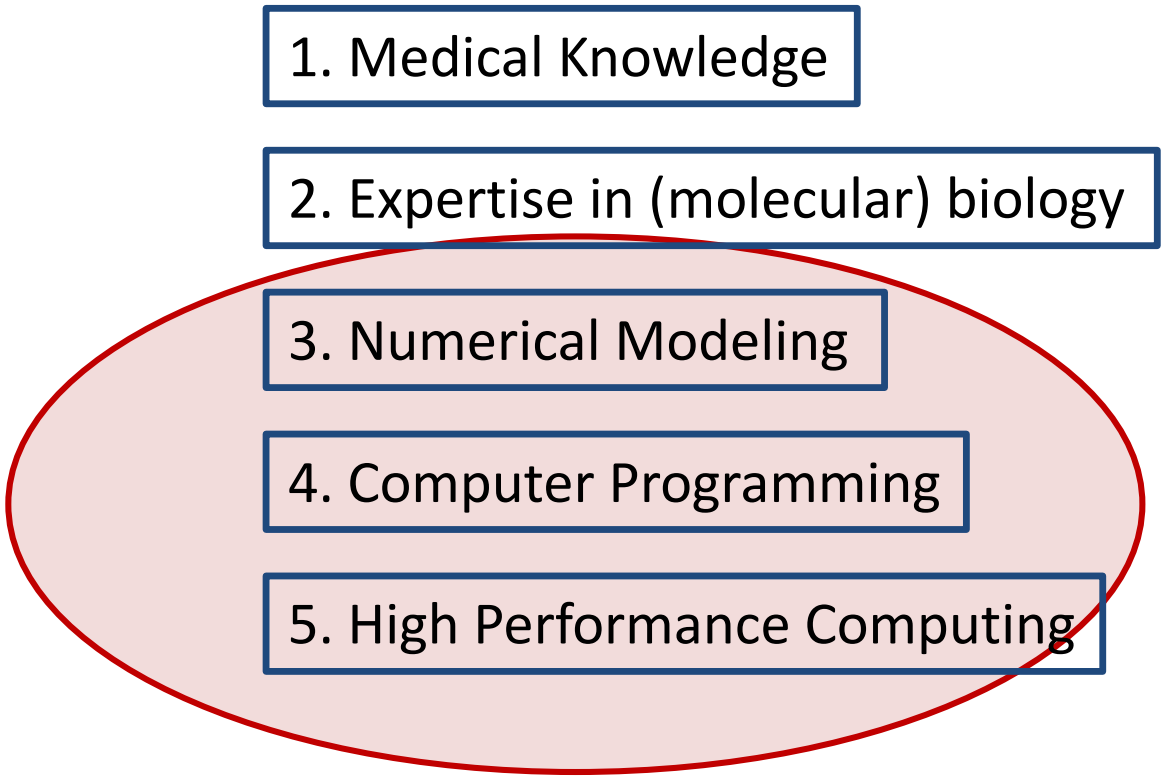
Computational Biomedicine: Elements

5. High Performance Computing

Piz Daint Supercomputer, CSCS (Switzerland)

1. Medical Knowledge
2. Expertise in (molecular) biology
3. Numerical Modeling
4. Computer Programming

Computational Biomedicine: Elements



Today's topic:
Computational Fluid Dynamics

Computer solution to fluid flow equations (so-called Navier-Stokes equations).

Examples of fluids:

- The blood or blood plasma in an artery.
- The air in the lungs.
- The Cerebrospinal fluid.
- Fluids used by doctors in medical interventions.
- …

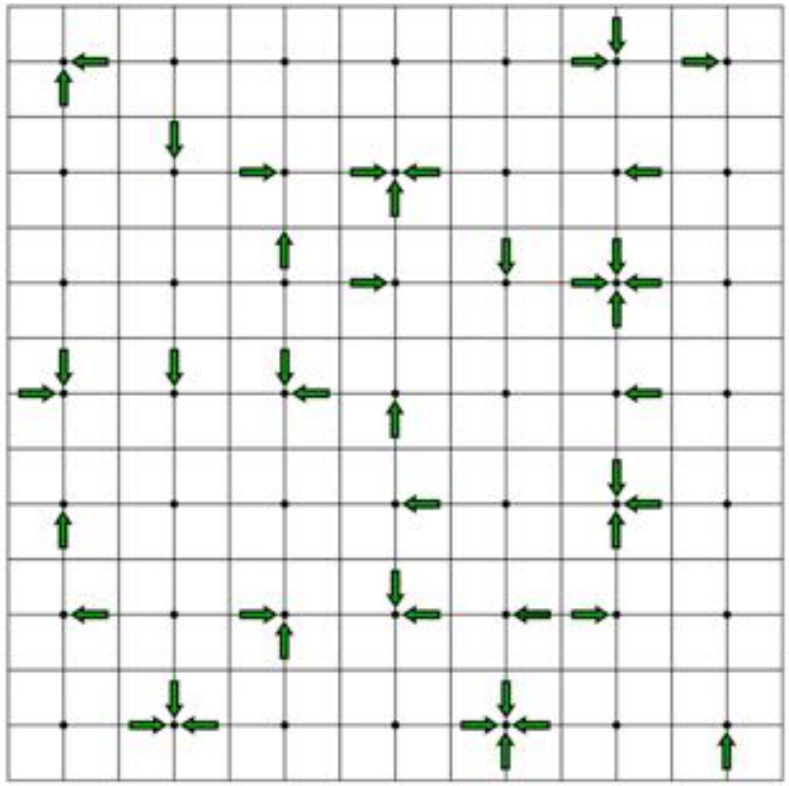
Today's method:
Lattice Boltzmann Method

- Modern approach to Computational Fluid Dynamics.
- Particularly useful for complex, coupled physical systems (e.g. representation of Red Blood Cells in the blood plasma).
- Adjusts in a simple manner to the requirements of high performance computers.

Technical part of the Webinar: Intro to Lattice Boltzmann Method

- Technical, precise introduction with reference to actual code.
- A simple, illustrative program in the Python language is provided alongside with this Webinar.
- No previous knowledge required!

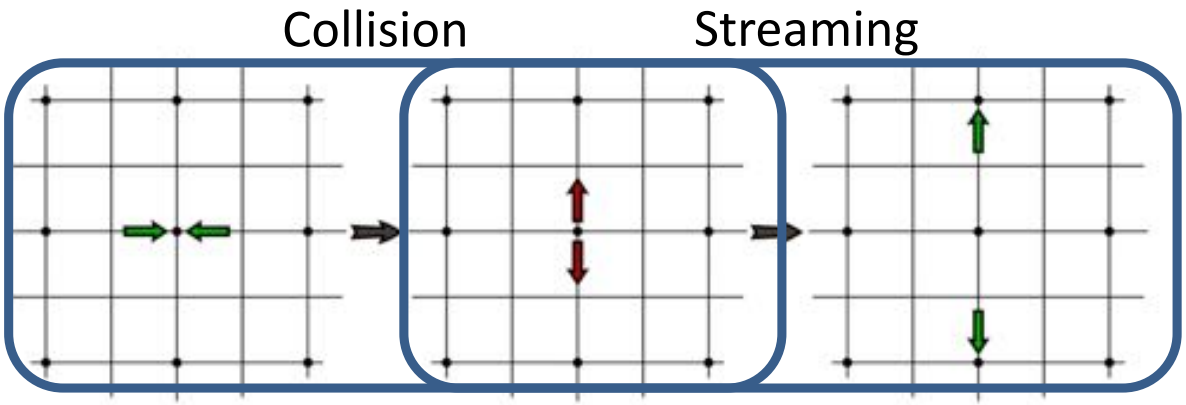
Background: Lattice Gas Automata



- HPP: A model from the 1980ies.
- Goal: represent a gas by modeling individual, yet highly simplified gas molecules.
- Space is discretized: represented by individual cells.
- Boolean model: each cell has between 0 and 4 «molecules», traveling in 1 one the main directions.

Lattice Gas Automata: Model

- When several particles meet on a node, they collide: all physics takes place here.
- After collision, particles are streamed to a nearest neighbor.



Our color scheme:

Green: pre-collision

Red: post-collision

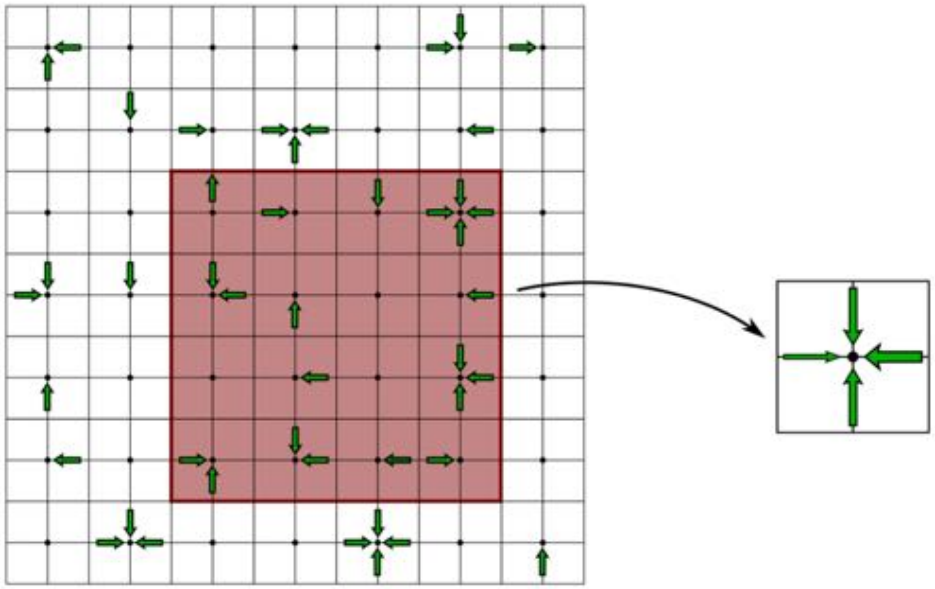
Poll time!

Please respond to the poll that will appear on the screen

Starting from Cellular Automata, how could simulation speed be improved substantially?

Right Answer: By replacing a discrete model by a statistical one.

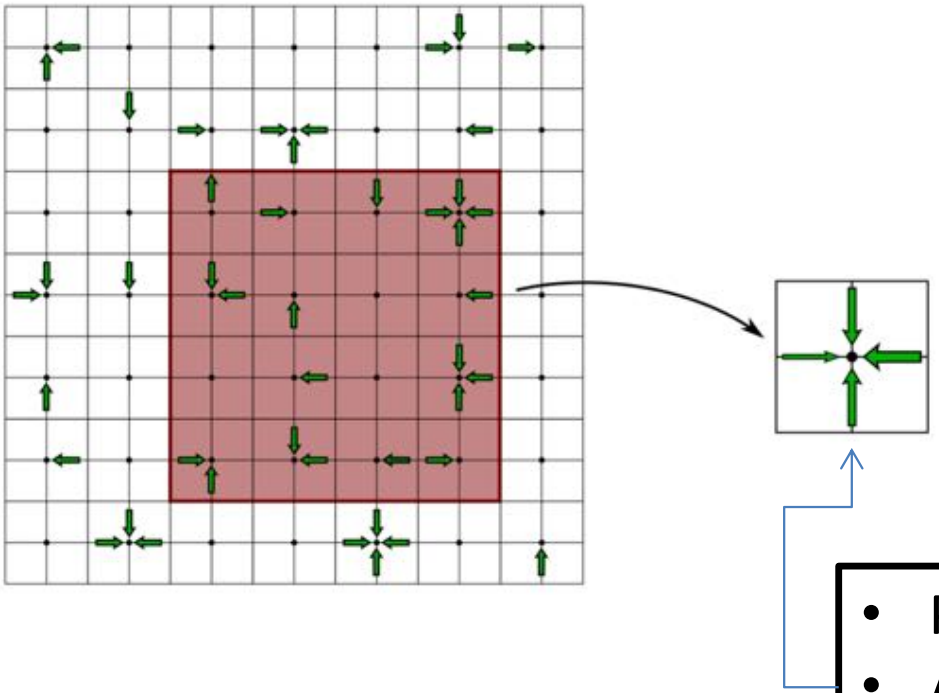
From discrete to continuum variables



In a lattice gas automaton:

- Run the simulation with discrete variables.
- At the end, extract macroscopic variables by taking averages.

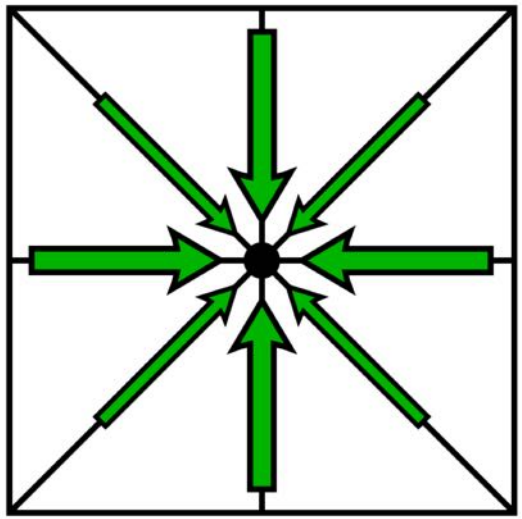
Lattice Boltzmann Method: Idea



- Run simulation with continuum variables right away, to save space.
- Continuum model is derived from discrete method by means of statistics.

- Particle density: real-valued.
- Arrow thickness stands for particle density.

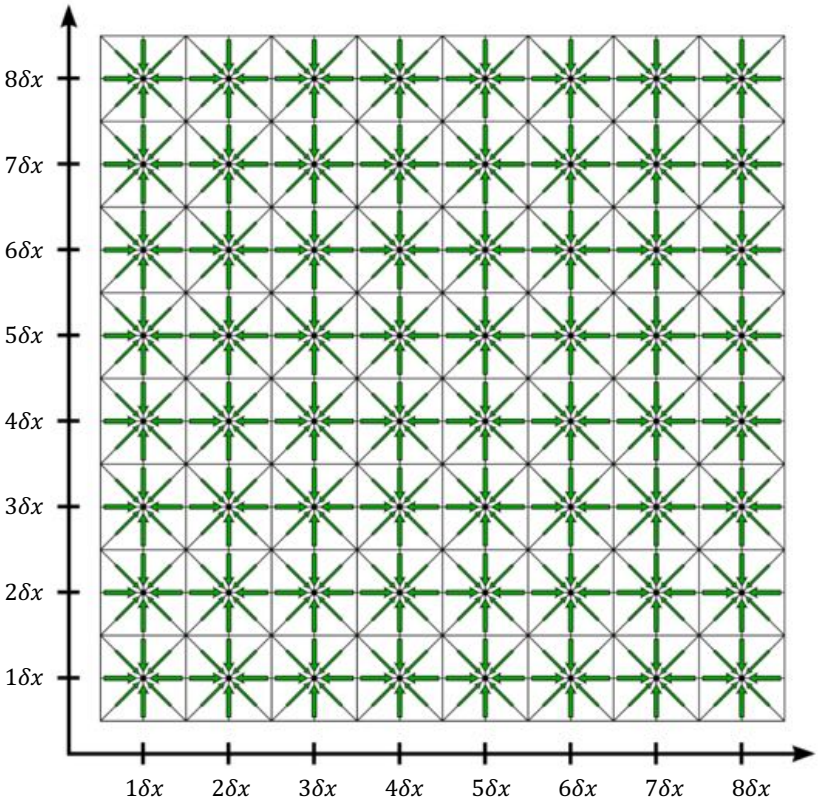
Lattice Boltzmann Method: details



Lattice Boltzmann 2D model D2Q9:

- Four directions aren’t quite enough.
- Nine directions: eight connections to nearest neighbors + «rest population».
- Variables representing particle densities are called «populations».

Lattice Boltzmann Variables



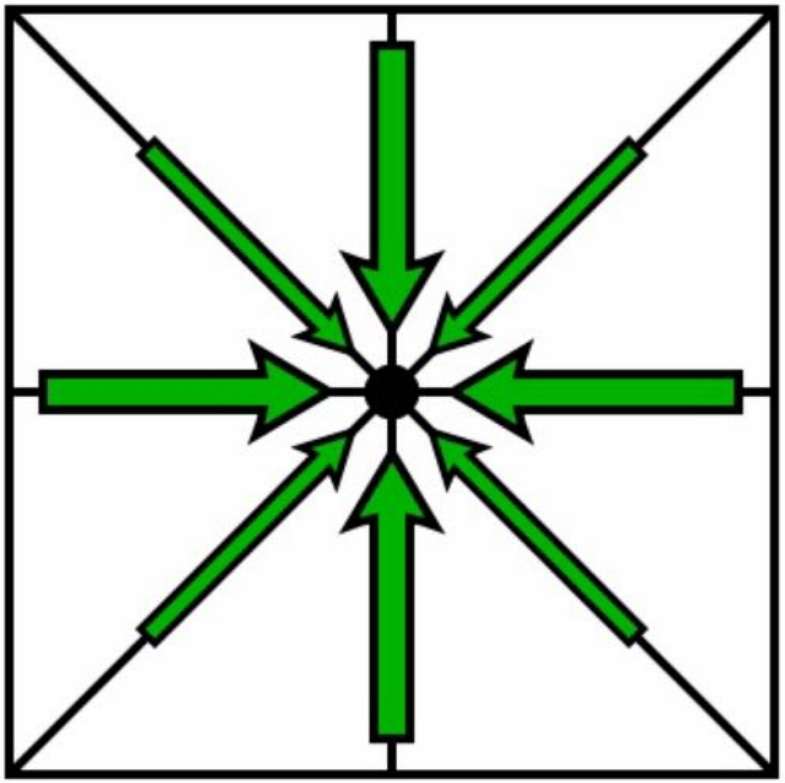
Space Discretization:

- Every cell represents flow variables at a point in space.
- To represent space with 8x8 points, we need a total of 8x8x9 = 576 floating point variables.
- Equal distance δx between cells in x- and y-direction.

Python code to allocate memory:

```
f = zeros(9, 8, 8)
```

Collision step



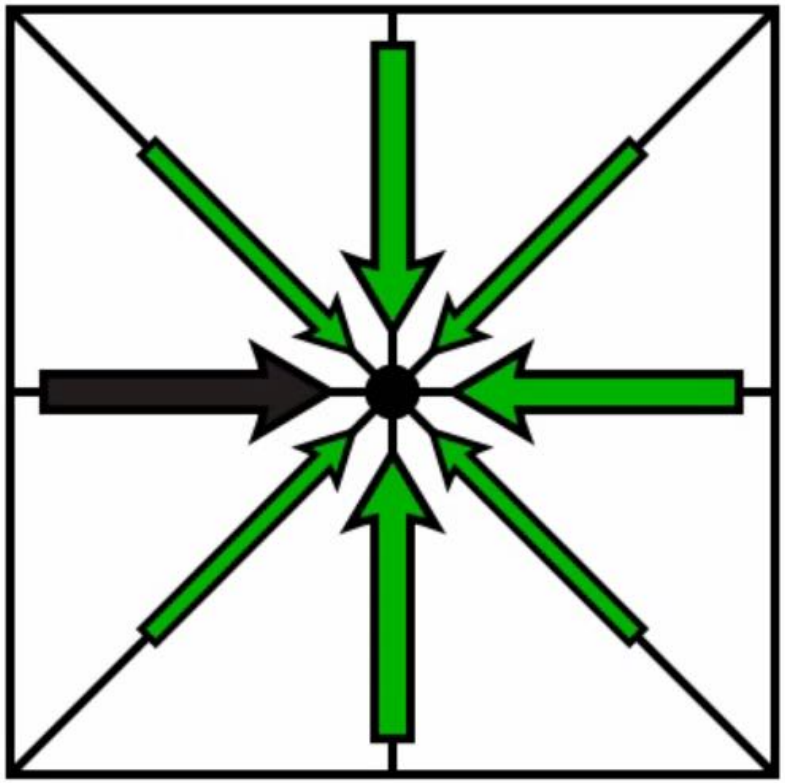
- Like lattice gas: maps pre-collision to post-collision populations.
- All populations are modified during collision. Changes are not visible on video, because they are small perturbations.

Our color scheme:

Green: pre-collision

Red: post-collision

Collision step



Model:

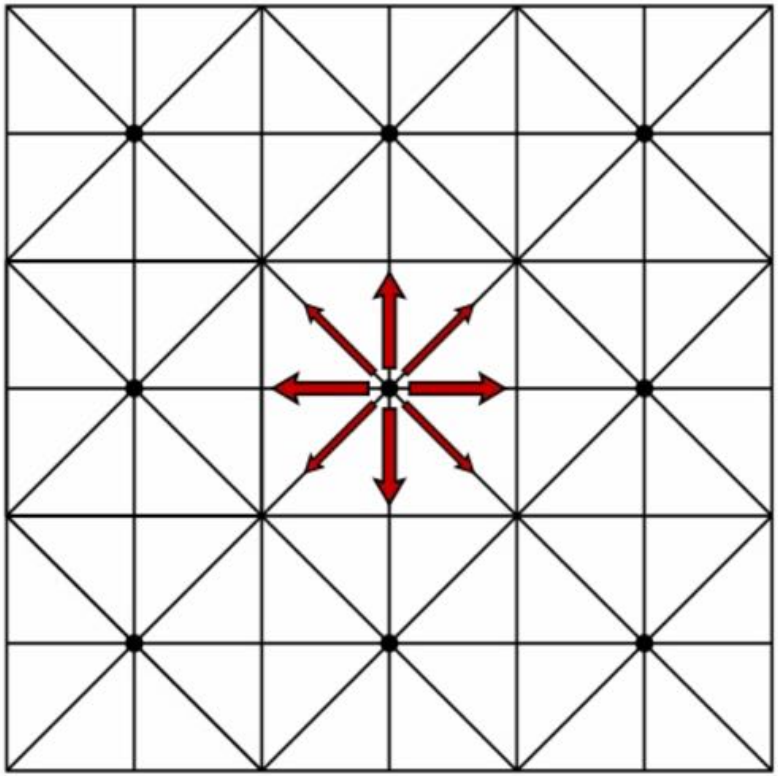
- Collision is instantaneous.
- Collision is localized at cell coordinates.

Our color scheme:

Green: pre-collision

Red: post-collision

Streaming step



Model:

- Streaming takes the system from time t to time $t + \delta t$.
- Streaming includes nearest-neighbor access.

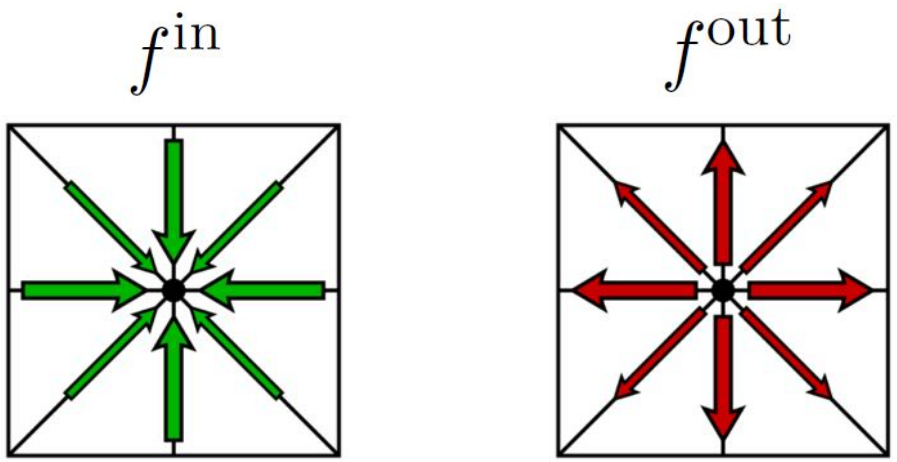
Our color scheme:

Green: pre-collision

Red: post-collision

Pre- and post-collision: variables

In our Python code, we save incoming and outgoing populations in separate matrices.



Python code:

```
# assign some size
nx, ny = …
fin = zeros(9, nx, ny)
fout = zeros(9, nx, ny)
```

Density

Each cell has its own density, which is the sum of the nine populations.

Equation

$$\rho(\boldsymbol{x}, t) = \sum_{i=0}^{8} f_i^{\text{in}}(\boldsymbol{x}, t)$$

Python Code

```
rho = sum(fin, axis=0)
```

Pressure

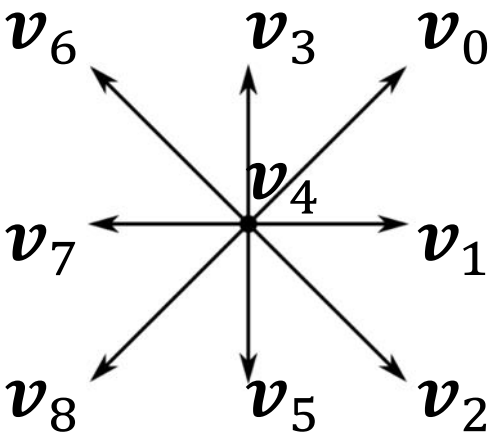
Pressure is proportional to density, like in an ideal gas:

$$\boxed{p = c_s^2 \rho}$$

- Blood is not a gas, it is incompressible: Density is constant.
- We still solve for a quantity we call “density”, just as a trick to compute the pressure cheaply.

Velocity

Let's introduce a set of 9 vectors:



$\boldsymbol{v}_0 = (1,1),$
$\boldsymbol{v}_1 = (1,0),$
$\boldsymbol{v}_2 = (1,-1),$
…
$\boldsymbol{v}_4 = (0,0),$
…

Velocity is a weighted sum of the populations:

$$\boldsymbol{u}(\boldsymbol{x},t) = \frac{1}{\rho(\boldsymbol{x},t)}\frac{\delta x}{\delta t}\sum_{i=0}^{8}\boldsymbol{v}_i f_i^{\text{in}}(\boldsymbol{x},t)$$

Python code:

```
u = zeros((2, nx, ny))
for i in range(9):
    u[0,:,:] += v[i,0] * fin[i,:,:]
    u[1,:,:] += v[i,1] * fin[i,:,:]
u /= rho
```

Collision: BGK model

Collision is a relaxation to local equilibrium:

$$f_i^{\text{out}} - f_i^{\text{in}} = -\omega \left(f_i^{\text{in}} - E(i, \rho, \vec{u}) \right)$$

- *E*: local equilibrium. Depends on the macroscopic variables and has a different value for every direction *i*.
- ω: frequency of relaxation, relates to the fluid viscosity.

Collision: Python code

Collision formula:

$$f_i^{\text{out}} - f_i^{\text{in}} = -\omega \left(f_i^{\text{in}} - E(i, \rho, \vec{u}) \right)$$

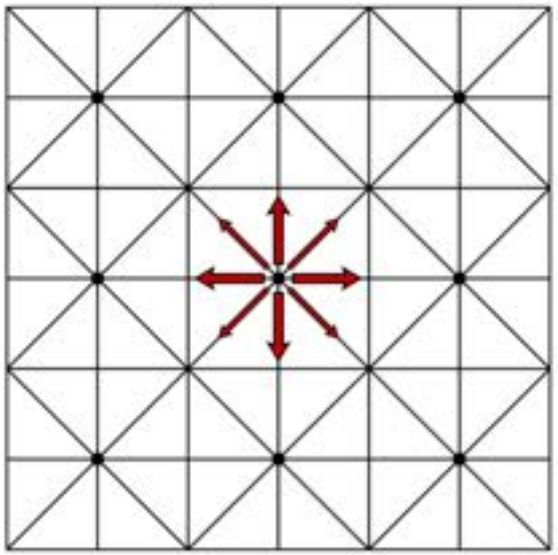
Python program:

```
fout = fin - omega * (fin - eq)
```

Streaming step

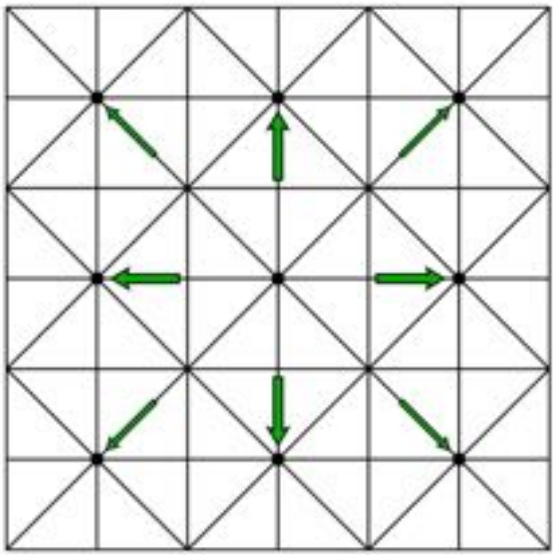
Streaming takes the system to the next time iteration, $t + \delta t$

Time t



Post-collision f^{out}

Time $t + \delta t$



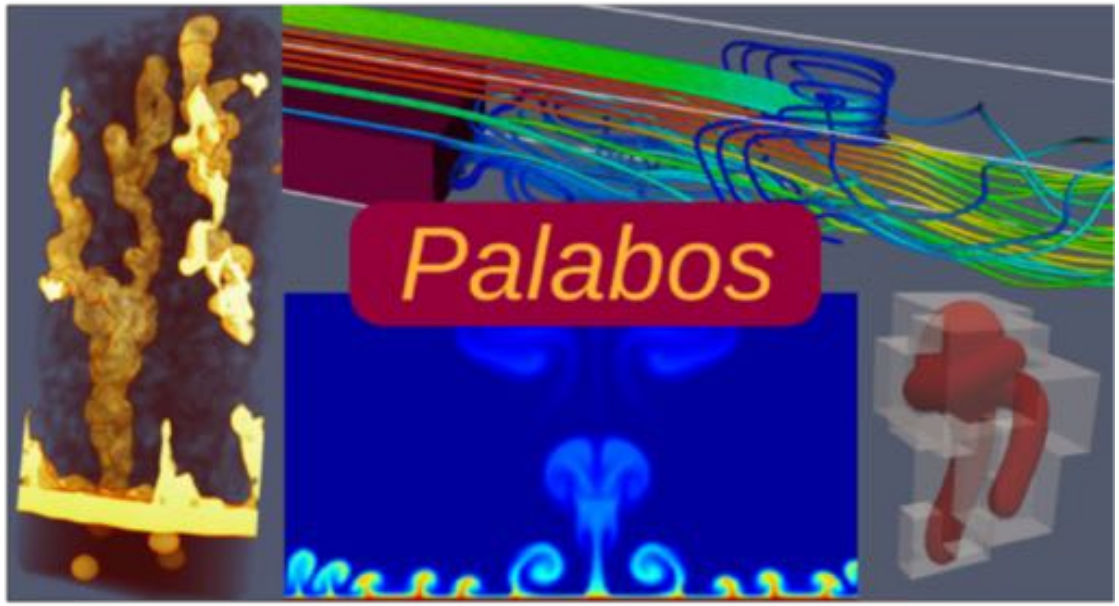
Pre-collision f^{in}

Streaming step: code

Python code:

```
for i in range(9):
    fin[i,:,:] = roll(
                    roll(fout[i,:,:], v[i,0], axis=0),
                    v[i,1], axis=1 )
```

The Palabos software



- Developed at University of Geneva.
- Our tool for biomedial fluid dynamics: all examples today run by Palabos.
- Open-Source: www.palabos.org

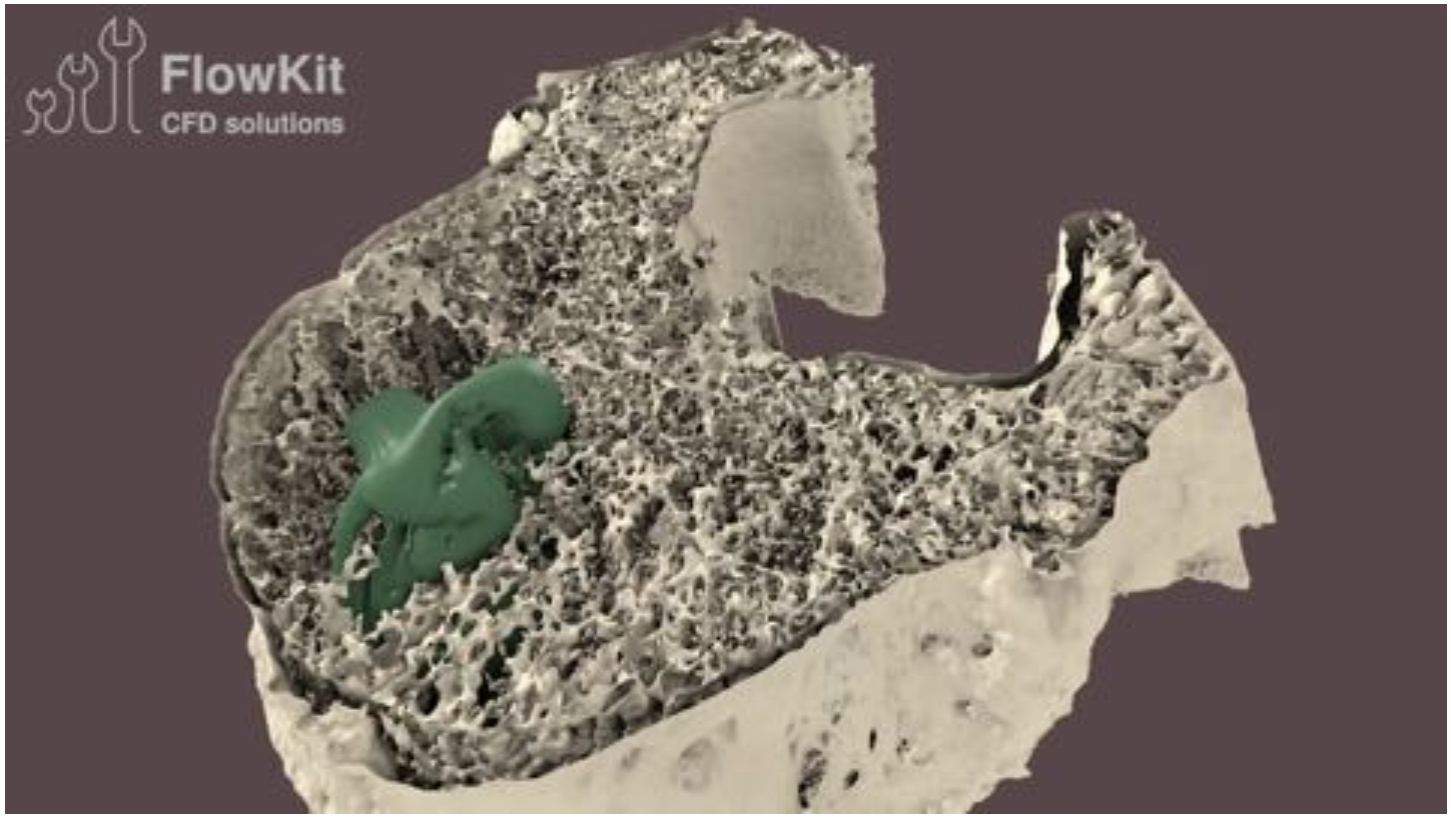
Another simulation example

Vertebroplasty:

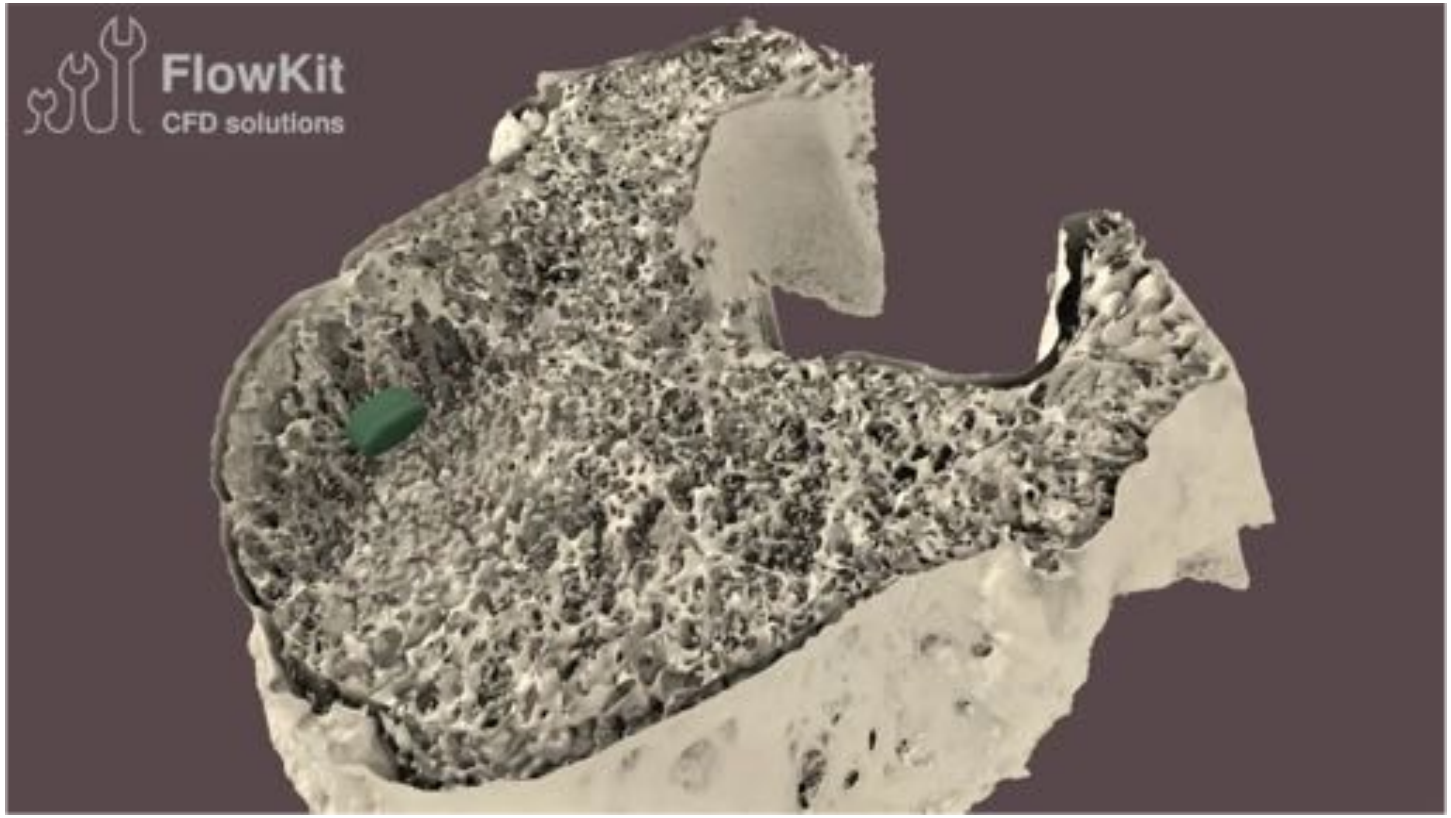
Treatment of compression fractures in a vertebra.

Image: Blausen.com staff

Micro-CT: porous bone structure

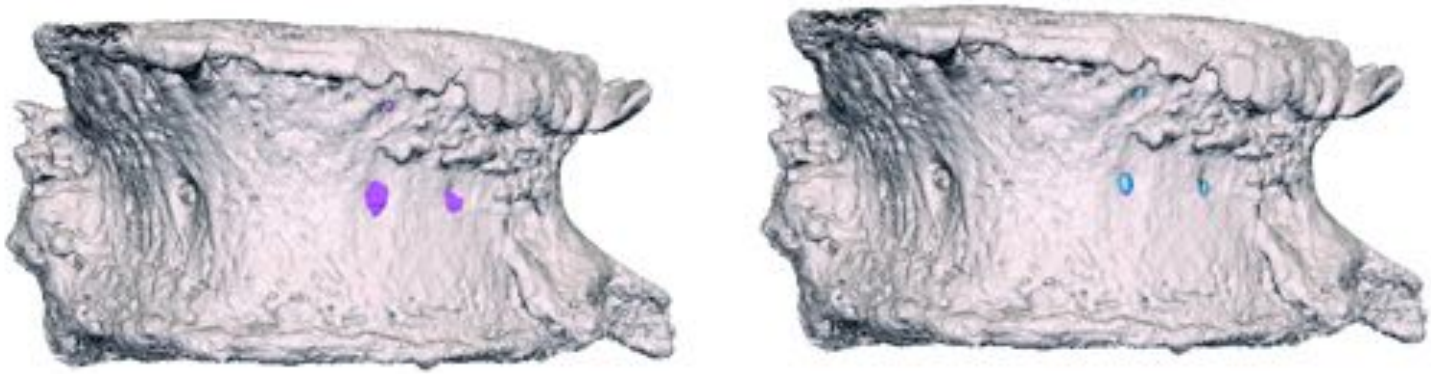


Computer simulation



Experiment vs. Simulation: Case 1

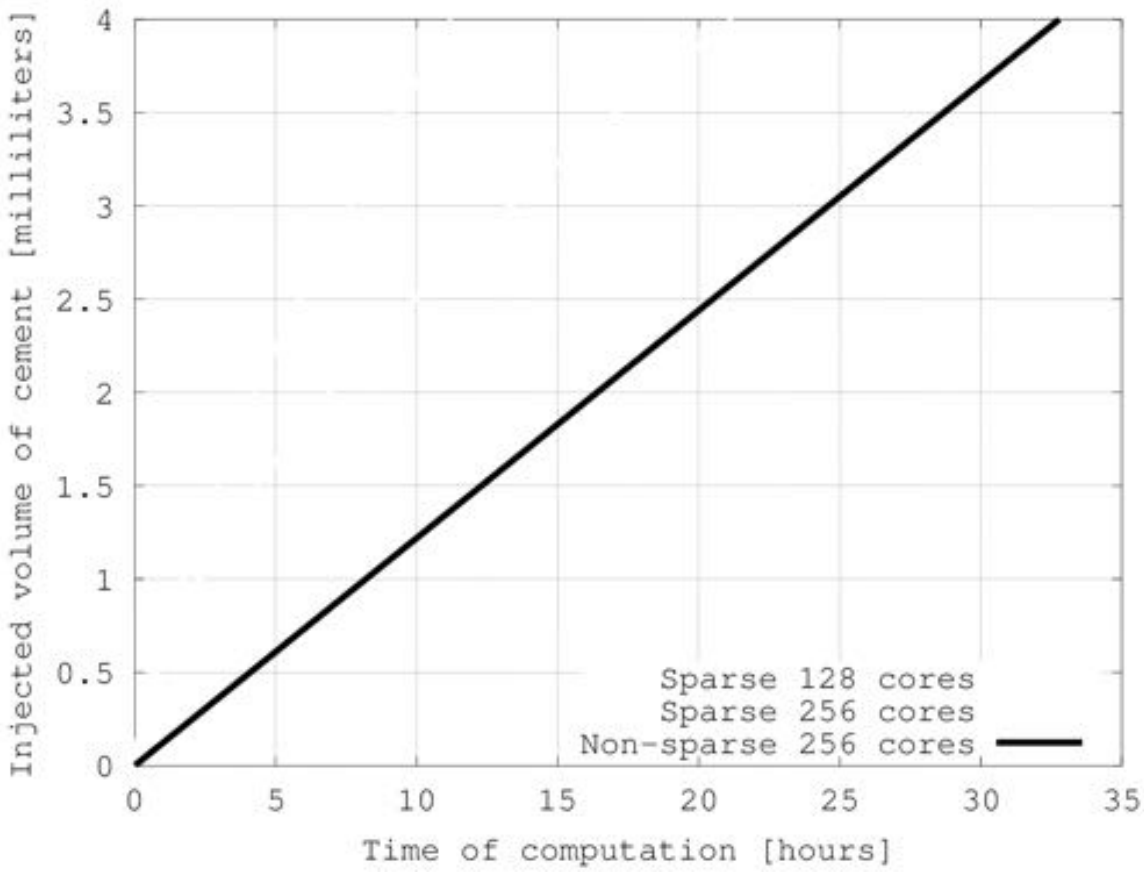
After 3 milliliters



Experiment

Simulation

Can we do « real time »?



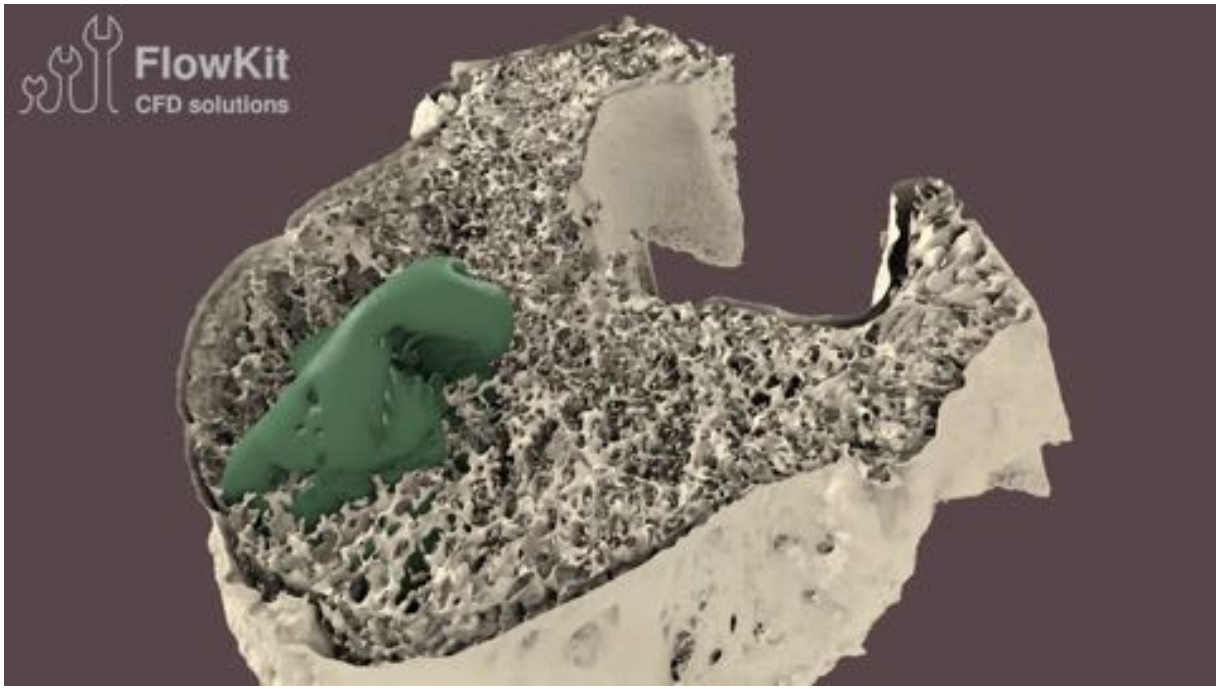
Poll time!

Please respond to the poll that will appear on the screen

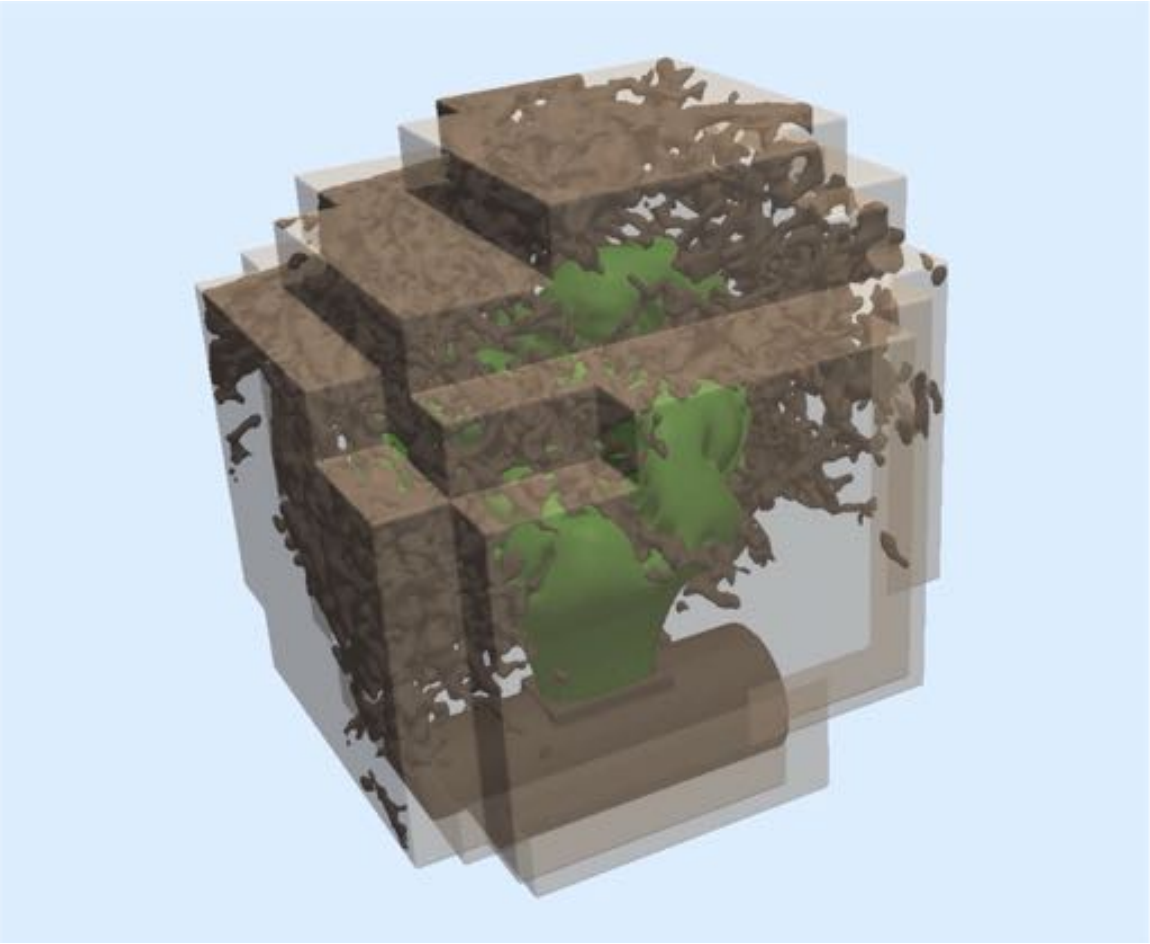
Which trick made this simulation substantially faster?

Right Answer: Allocate only areas filled by cement, dynamically

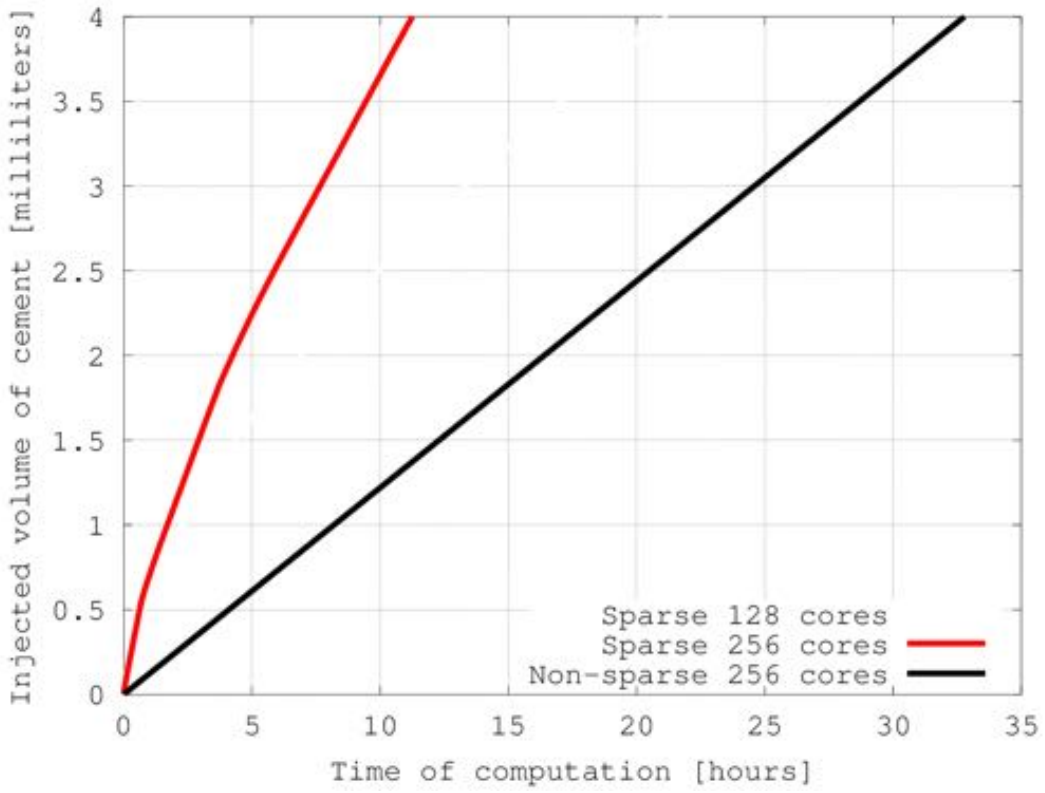
A better idea: sparse memory



A better idea: sparse memory



Sparse-Mem 3-4 times faster



Many thanks to …

The UniGe CompBioMed Team

Prof. Bastien Chopard

Sha Li

Francesco Marson

Christos Kotsalos

The FlowKit Ltd team

Dimitrios Kontaxakis

Orestis Malaspinas

Andrea Di Blasio

Annick Baur

Further Resources

Palabos Web Page:

www.palabos.org

A more thorough course:

https://www.coursera.org/learn/modeling-simulation-natural-processes

Q&A

To pose a question, you can:

1. if you have a microphone, raise your hand and we will unmute you, when it is your turn to speak or
2. write your question in the “Questions” tab

Thank you for participating!

Visit the CompBioMed website (www.compbiomed.eu/training-3)
for a full recording of this and other webinars,
to download the slides
and to keep updated on our upcoming trainings

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 675451

The series is run in collaboration with:
VPH Institute
Building the Virtual Physiological Human