

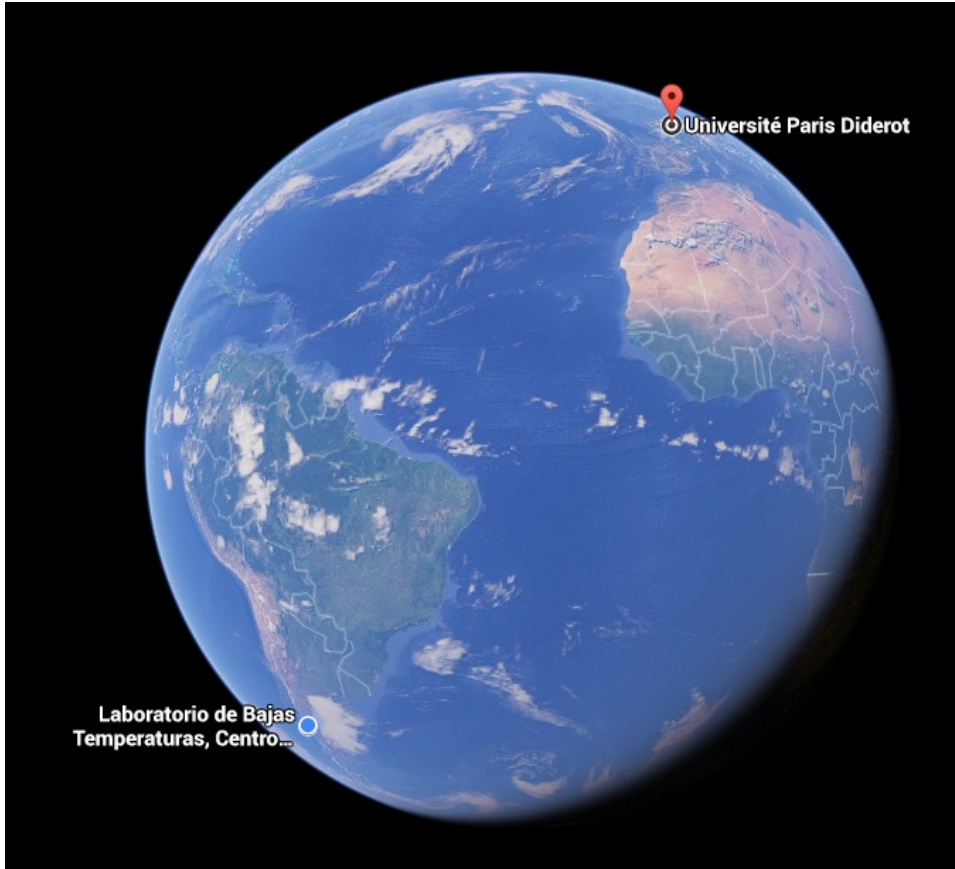
# Interfaces in Random Media: “Playing the elastic string game”

**Alejandro B. Kolton**

CONICET - Centro Atómico Bariloche - Argentina



# Bariloche, Argentina





**Bariloche, Argentina, by Vivien Lecomte**

# Agenda

- **Motivation:**
  - Nature, experiments
  - Phenomenological relevant questions
- **Concepts:**
  - Universality, models, interesting quantities
- **Playing the game:**
  - Numerical Algorithms
  - Programming Graphics Cards
- **Some answers, a lot of new questions!**

Have you ever observed a similar phenomenon?



QUIZ

# Magnetic Domain Wall

Domain 1

Wall



Domain 2

Thin films  
 $e \approx 0.5nm$

DOWN

$\sim 10\mu m$

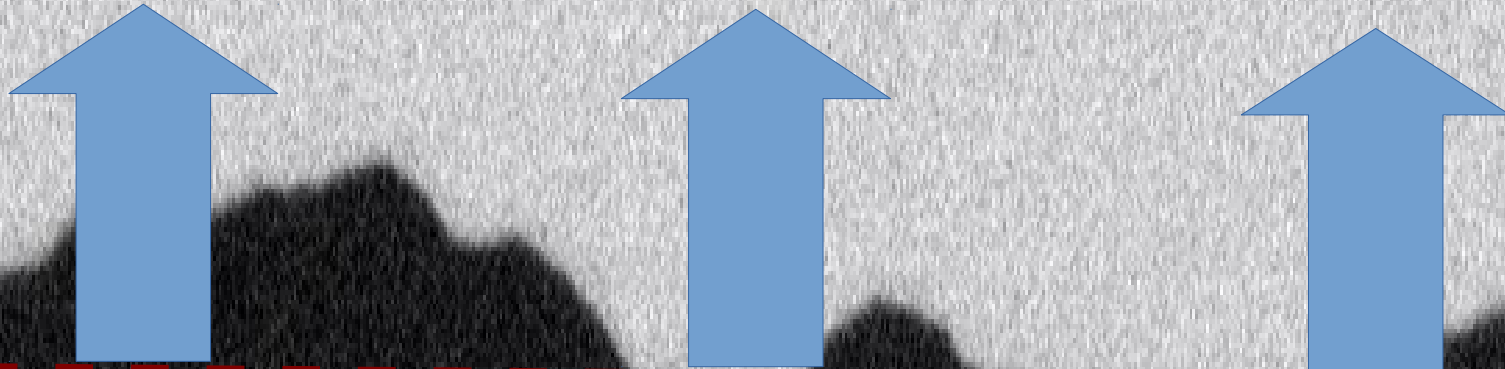
*Disorder vs Elasticity*  
*A dynamic competition*

UP

Response to H in  
the UP direction?

# TRANSPORT PROBLEM

Mean velocity  $V$  vs applied field  $H$  and Temperature  $T$  ?



$$V(H, T) = ?$$



$V(H,T)?...$  does it matter?

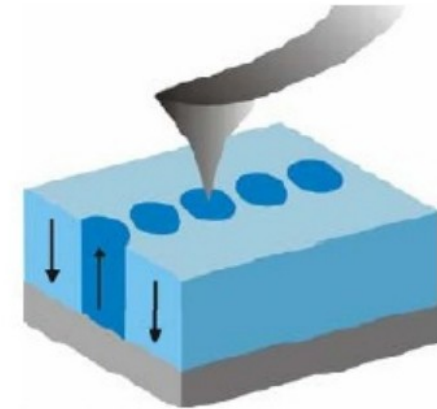
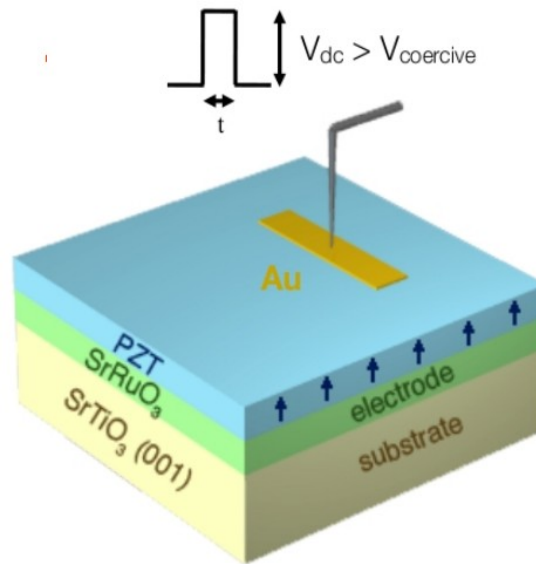
Interfaces Motion control → **Applications**

Disorder vs Elasticity → **Universality**  
[in collective transport]

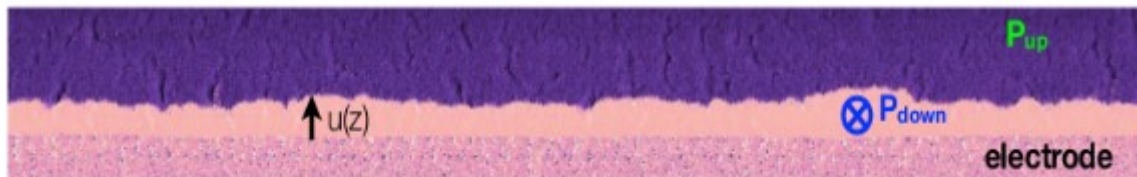
→ Similar Effective Physics shared by rather different systems

# Ferroelectric Domain Walls

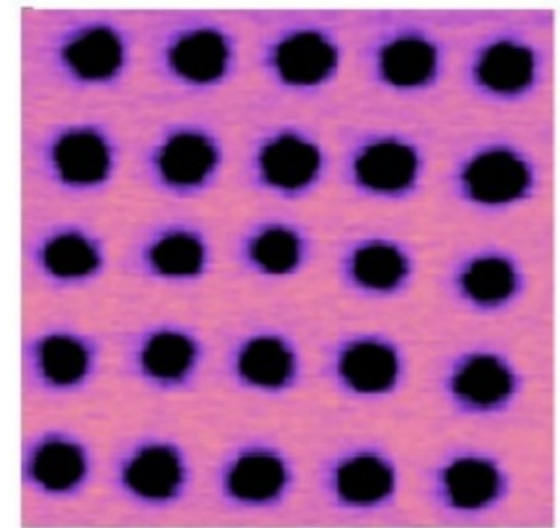
# Ferroelectric Domain Walls



Pinning  $\rightarrow$  rough walls



with different writing times



*Ferroelectric Domain Walls*  
P. Paruch et al., (Geneva).

# Contact lines in partial wetting

# Wetting: three phases contact

*Spreading parameter*

$$S = [E_{\text{substrate}}]_{\text{dry}} - [E_{\text{substrate}}]_{\text{wet}}$$

$$S = \gamma_{SO} - (\gamma_{SL} + \gamma),$$

*Contact Angle*

*Law of Young-Dupré*

$$\gamma \cos \theta_E = \gamma_{SO} - \gamma_{SL}$$

$$S = \gamma(\cos \theta_E - 1)$$

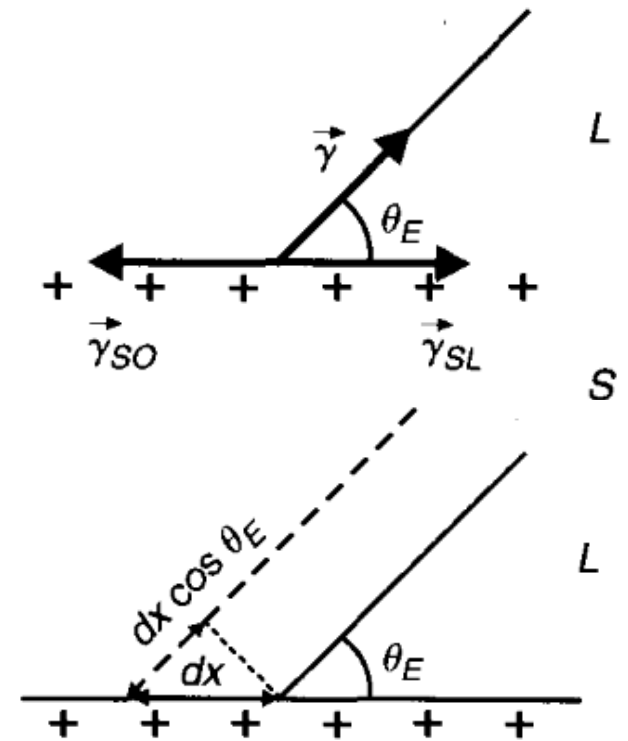
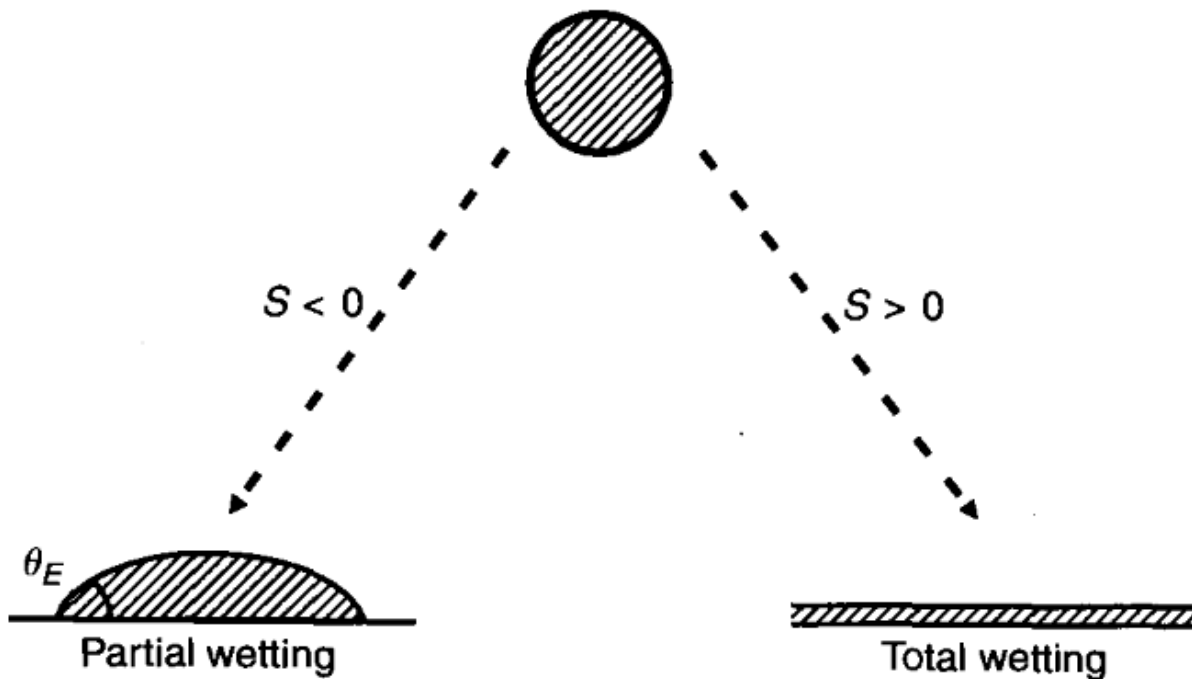


FIGURE 1.13. The two wetting regimes for sessile drops.

# Drops at equilibrium

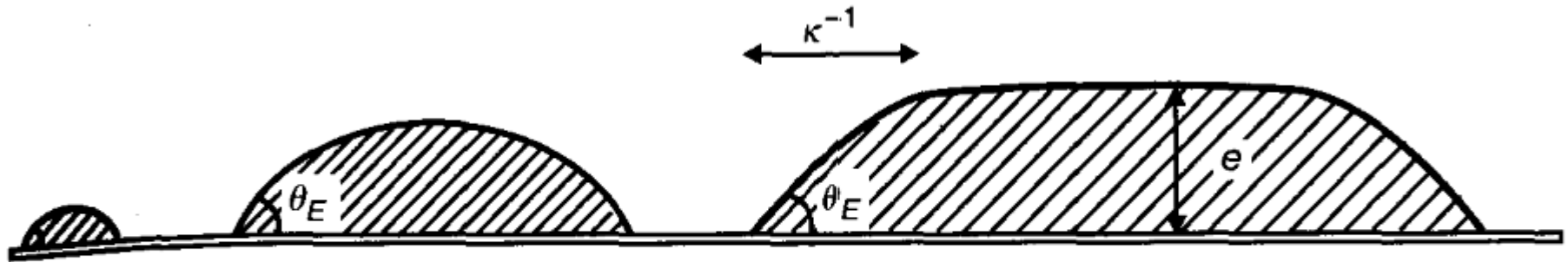


FIGURE 2.3. Water drops of increasing size on a sheet of plastic. Gravity causes the largest drops to flatten.

Capillary length

$$\kappa^{-1} = \sqrt{\gamma/\rho g}$$

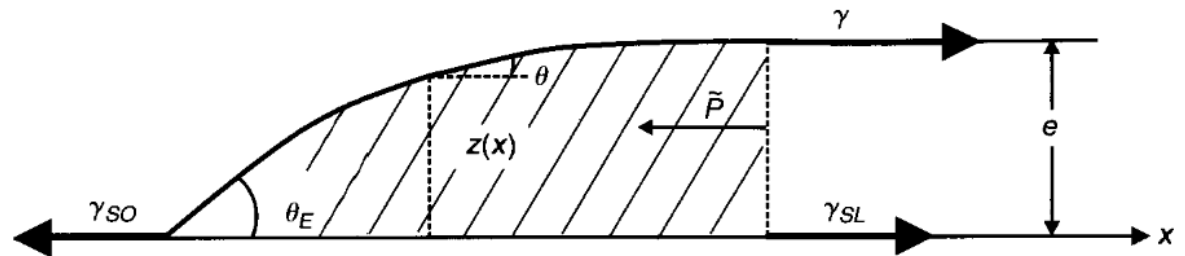
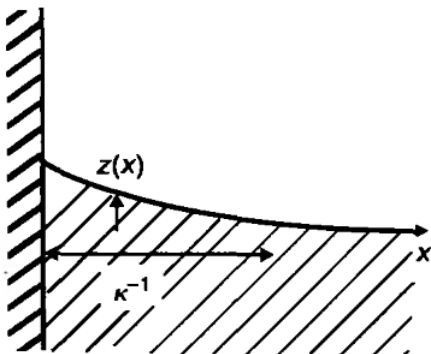


FIGURE 2.4. Equilibrium of the forces (per unit length of the line of contact) acting on the edge of a puddle.  $\tilde{P} = \rho g e^2 / 2$  is the hydrostatic pressure.

$$F_g = -SA + \frac{1}{2}\rho g e^2 A. \longrightarrow e = 2\kappa^{-1} \sin\left(\frac{\theta_E}{2}\right)$$



# Evaporating Drops (at home)



Before



After

Very common phenomenon





# Try this at home!

- Why do solutes accumulate in the borders?
- **Why they do not just shrink like this?**

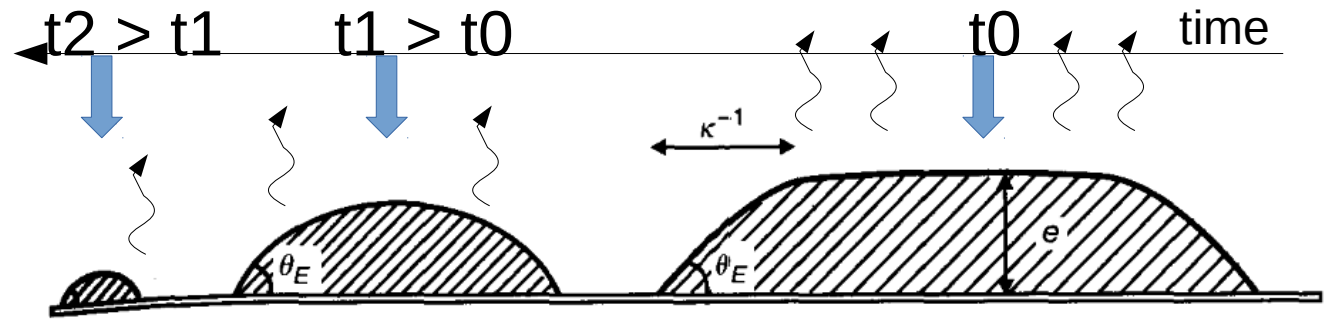


FIGURE 2.3. Water drops of increasing size on a sheet of plastic. Gravity causes the largest drops to flatten.

# Imperfect surfaces → pinning of the contact line

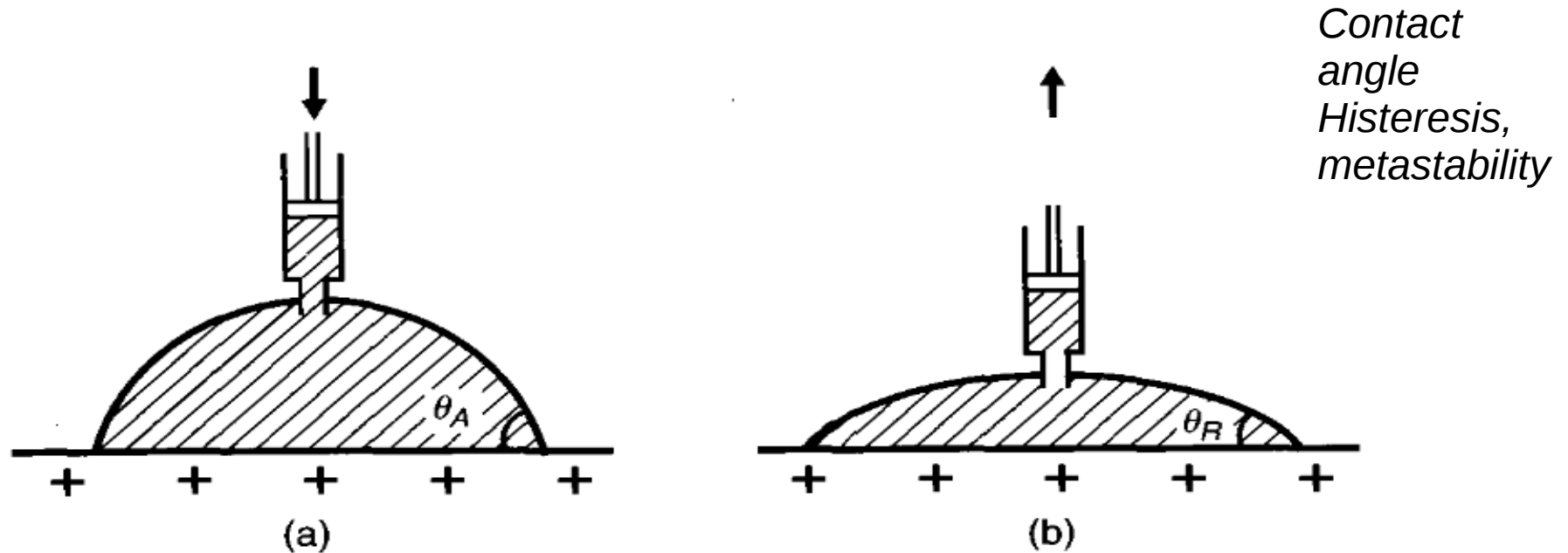


FIGURE 3.1. (a) Advancing angle when the drop is inflated; (b) receding angle when the drop is deflated.

# Ring Formation

NATURE | VOL 389 | 23 OCTOBER 1997

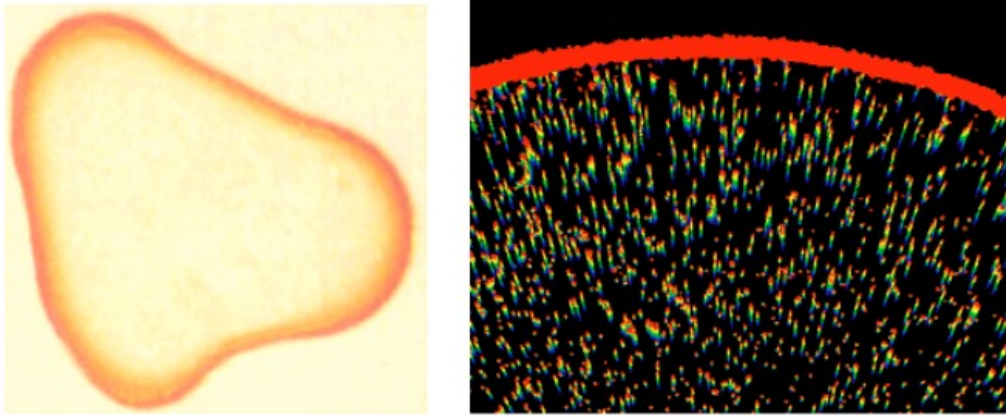


FIG. 1. a) A 2-cm-diameter drop of coffee containing one-weight-percent solids has dried to form a perimeter ring, accentuated in regions of high curvature. b) Video micrographs showing dispersion of fluorescent one-micron polystyrene spheres in water during evaporation, as described in the text. Multiple exposure shows different times in different colors to indicate the motion. Earliest time (blue) is 3 sec before latest time (red).

## Capillary flow as the cause of ring stains from dried liquid drops

Robert D. Deegan\*, Olgica Bakajin\*, Todd F. Dupont†, Greb Huber\*, Sidney R. Nagel\* & Thomas A. Witten\*

\* James Franck Institute, 5640 South Ellis Avenue, Chicago, Illinois 60637, USA

† Department of Computer Science, University of Chicago, 1100 East 58th Street, Chicago, Illinois 60637, USA

Our qualitative observations show that rings form for a wide variety of substrates, dispersed materials (solutes), and carrier liquids (solvents), as long as (1) the solvent meets the surface at a nonzero contact angle, (2) the contact line is pinned to its initial position, and (3) the solvent evaporates. In addition, we found that mechanisms typically responsible for solute transport—surface tension gradients, solute diffusion, electrostatic, and gravity effects—are negligible in ring formation. Based on these



## Effects of Particle Shape on Growth Dynamics at Edges of Evaporating Drops of Colloidal Suspensions

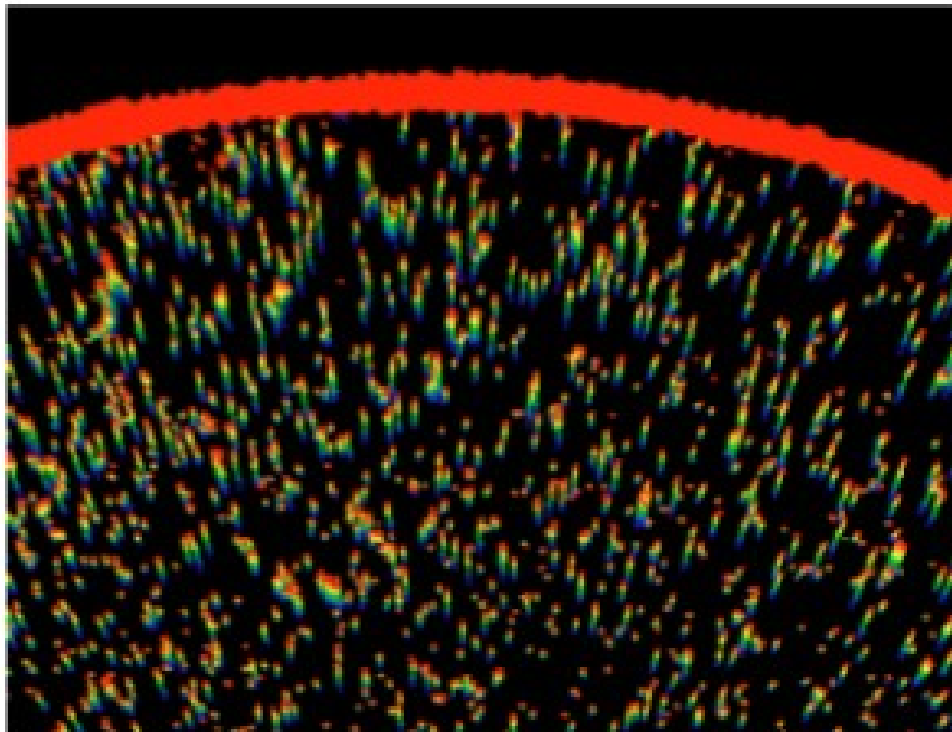
Peter J. Yunker,<sup>1</sup> Matthew A. Lohr,<sup>1</sup> Tim Still,<sup>1,2</sup> Alexei Borodin,<sup>3</sup> D. J. Durian,<sup>1</sup> and A. G. Yodh<sup>1</sup>

<sup>1</sup>*Department of Physics and Astronomy, University of Pennsylvania, Philadelphia, Pennsylvania 19104, USA*

<sup>2</sup>*Complex Assemblies of Soft Matter, CNRS-Rhodia-University of Pennsylvania, UMI 3254, Bristol, Pennsylvania 19007, USA*

<sup>3</sup>*Department of Mathematics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA*

(Received 23 July 2012; published 18 January 2013)



“surface growth” phenomena  
from the pinned (rough)  
contact line

# Very common phenomenon



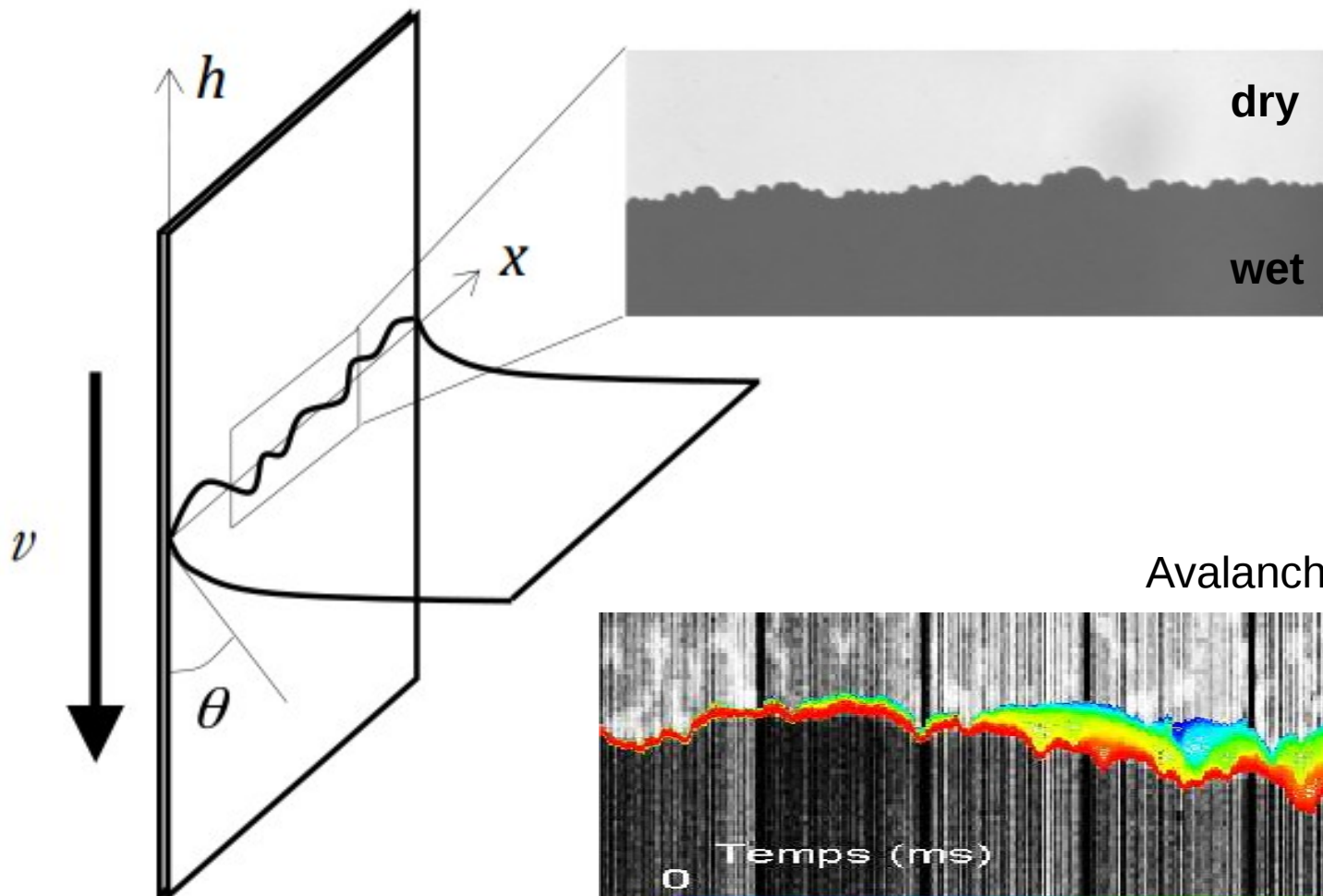
Memory of the pinned contact line (microscopically rough)

# Dynamics of pinned contact lines in the lab

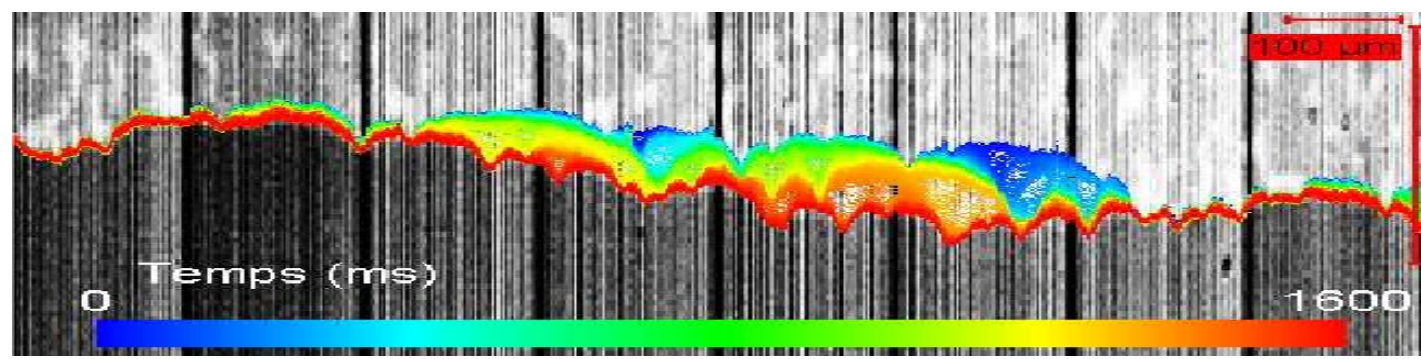
## Contact lines in partial wetting

Moulinet, Rolley (Paris).

Pinning → roughness

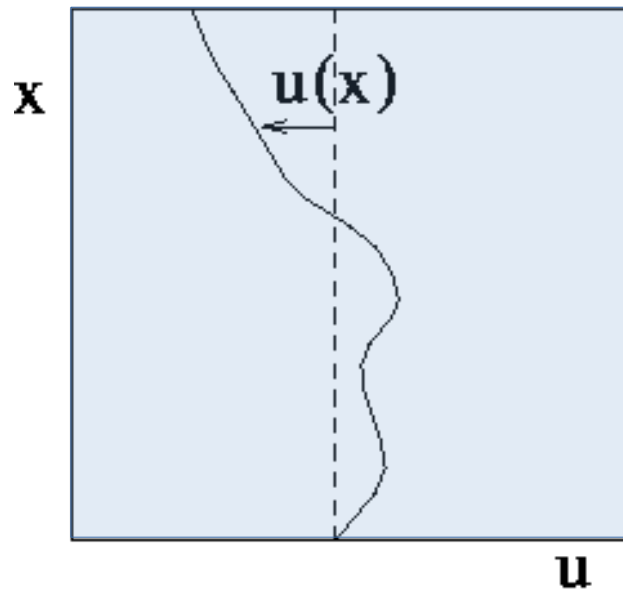


Avalanches!

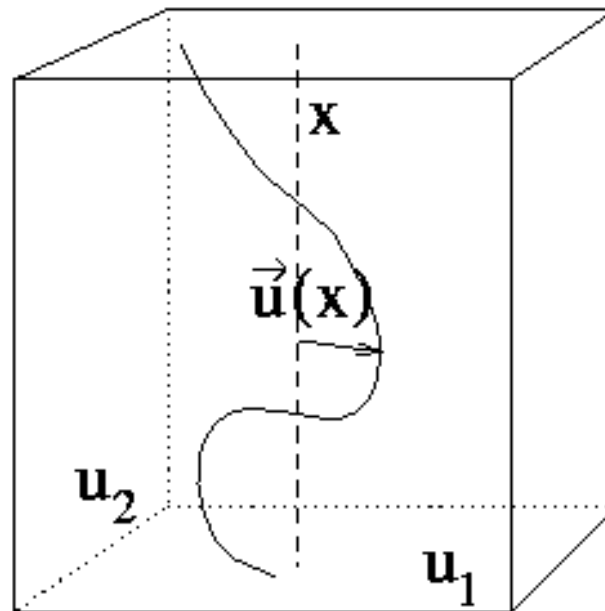


# Other Disordered Elastic Manifolds

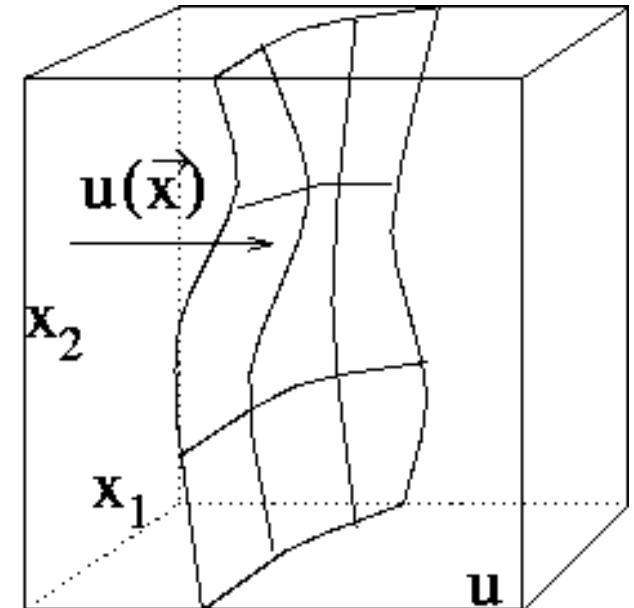
Examples so far



$N=d=1$   
(a)



$N=2$   $d=1$   
(b)

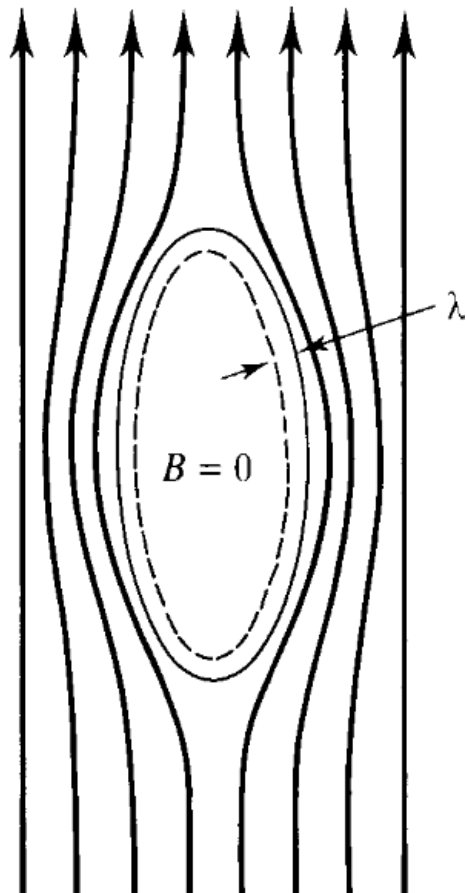


$N=1$   $d=2$   
(c)

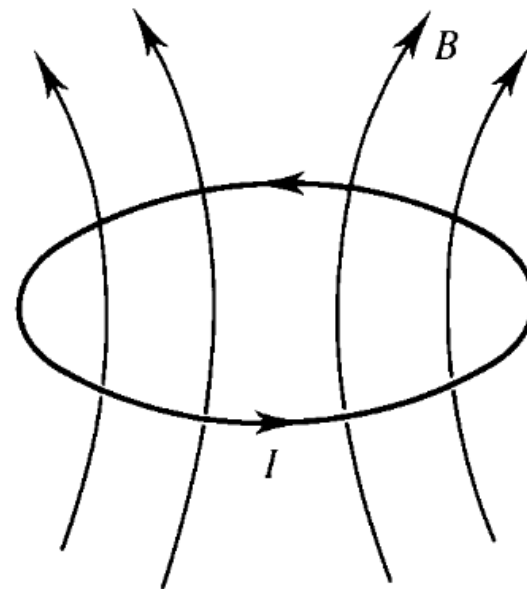
*Or evenmore, a collection of this kind of objects but coupled by some repulsive interaction*

# Phenomenological Superconductivity

Perfect Diamagnetism



Perfect Conductivity

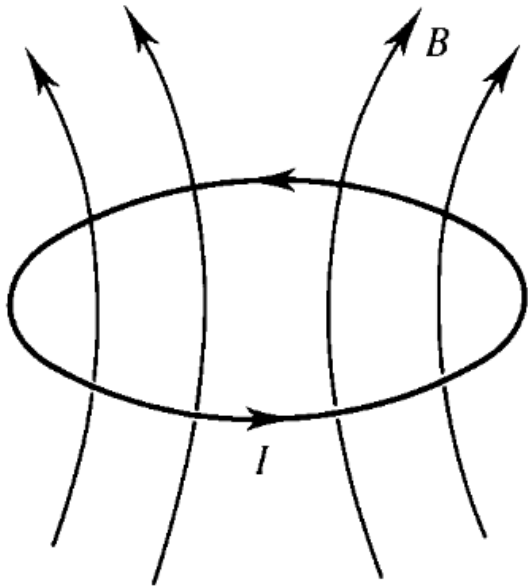




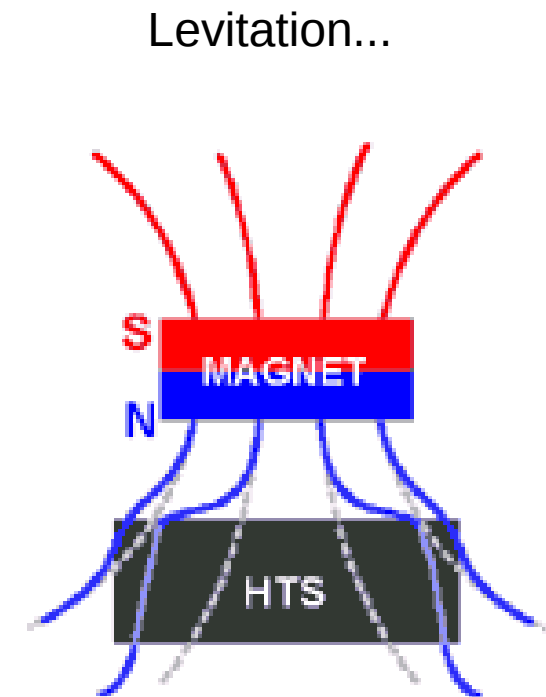
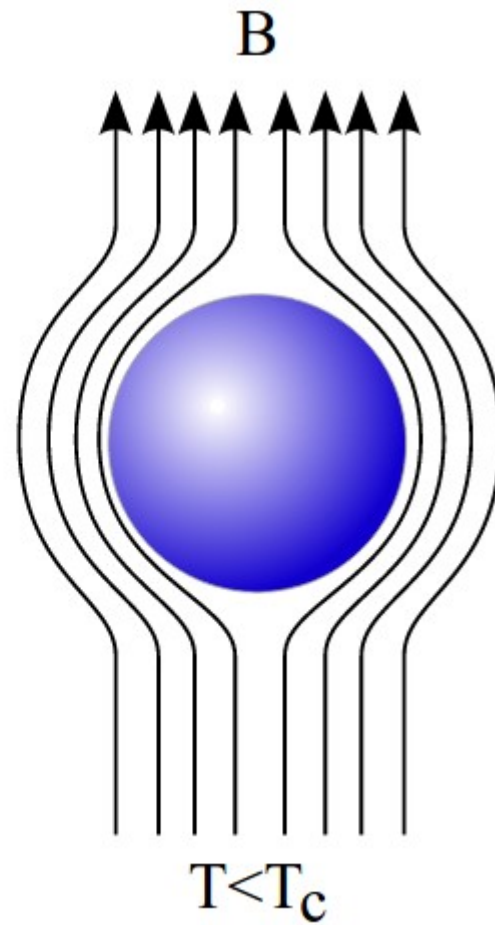
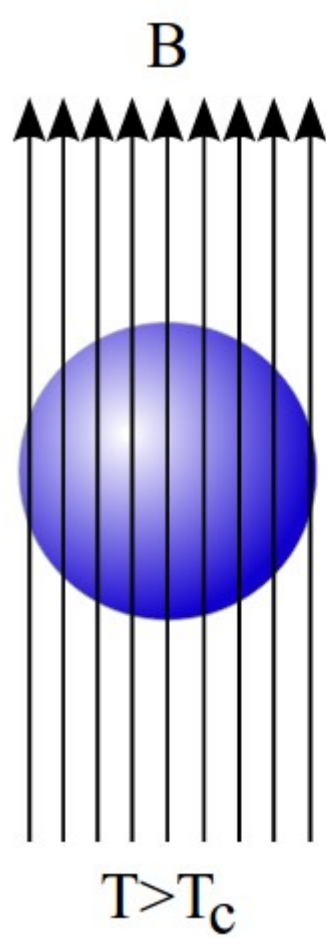
# Perfect Conductivity

Electric cables for accelerators at CERN. Both the massive and slim cables are rated for 12,500 A. Top: conventional cables for LEP; bottom: superconductor-based cables for the LHC

“perfect” conductivity



# Perfect diamagnetism



# Diamagnetism and Levitation

Magnetic levitation is a method by which an object is suspended with no support other than magnetic fields. Magnetic force is used to counteract the effects of the gravitational and any other accelerations.

*Earnshaw's theorem (1842) proves that using only paramagnetic or ferromagnetic materials it is impossible for a static system to stably levitate against gravity.*  
[TRY IT WITH MAGNETS!]

# Water is diamagnetic...

- Earnshaw theorem does not forbid stable static equilibrium in a magnetic field...
- Live animals, plants, etc, are full of water...
- Can we levitate them in a magnetic field?

QUIZ

# (Dia)magnetic Levitation

Not forbidden by Earnshaw's theorem!



<https://www.youtube.com/watch?v=A1vyB-O5i6E>

**Geim: Ig Nobel To Nobel**

# How to magnetically levitate yourself [using your diamagnetism]

The minimum criterion for diamagnetic levitation is  $B \frac{dB}{dz} = \mu_0 \rho \frac{g}{\chi}$ , where:

- $\chi$  is the magnetic susceptibility
- $\rho$  is the density of the material
- $g$  is the local gravitational acceleration ( $-9.8 \text{ m/s}^2$  on Earth)
- $\mu_0$  is the permeability of free space
- $B$  is the magnetic field
- $\frac{dB}{dz}$  is the rate of change of the magnetic field along the vertical axis.

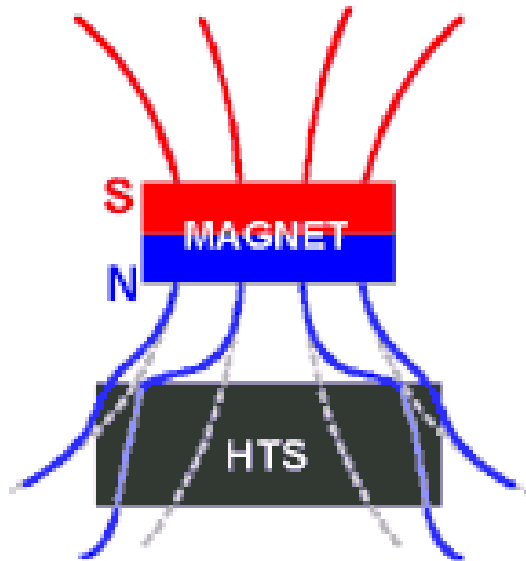
Assuming ideal conditions along the z-direction of solenoid magnet:

- Water levitates at  $B \frac{dB}{dz} \approx 1400 \text{ T}^2/\text{m}$
- Graphite levitates at  $B \frac{dB}{dz} \approx 375 \text{ T}^2/\text{m}$ .

# Magnetic fields

10 <sup>-9</sup>	nanotesla	100 pT to 10 nT	1 μG to 100 μG	magnetic field strength in the <a href="#">heliosphere</a>
10 <sup>-6</sup>	microtesla	24 μT	240 mG	strength of <a href="#">magnetic tape</a> near <a href="#">tape head</a>
10 <sup>-5</sup>		31 μT	310 mG	strength of <a href="#">Earth's magnetic field</a> at 0° latitude (on the <a href="#">equator</a> )
		58 μT	580 mG	strength of Earth's magnetic field at 50° <a href="#">latitude</a> ←
10 <sup>-3</sup>	millitesla	0.5 mT	5 G	the suggested exposure limit for <a href="#">cardiac pacemakers</a> by American Conference of Governmental Industrial Hygienists (ACGIH)
		5 mT	50 G	the strength of a typical <a href="#">refrigerator magnet</a> <sup>[4]</sup> ←
10 <sup>-1</sup>		0.15 T	1.5 kG	the magnetic field strength of a <a href="#">sunspot</a>
10 <sup>0</sup>	tesla	1 T to 2.4 T	10 kG to 24 kG	coil gap of a typical <a href="#">loudspeaker magnet</a> . <sup>[5]</sup>
		1 T to 2 T	10 kG to 20 kG	inside the core of a modern 60 Hz power transformer <sup>[6][7]</sup>
		1.25 T	12.5 kG	strength of a modern <a href="#">neodymium–iron–boron (Nd<sub>2</sub>Fe<sub>14</sub>B)</a> <a href="#">rare earth magnet</a> . A coin-sized neodymium magnet can lift more than 9 kg, pinch skin and erase credit cards. <sup>[8]</sup>
		1.5 T to 3 T	15 kG to 30 kG	strength of medical <a href="#">magnetic resonance imaging</a> systems in practice, experimentally up to 8 T <sup>[9][10]</sup>
		9.4 T	94 kG	Modern high resolution research <a href="#">magnetic resonance imaging</a> system; field strength of a 400 MHz <a href="#">NMR spectrometer</a>
		11.7 T	117 kG	field strength of a 500 MHz <a href="#">NMR spectrometer</a>
		16 T	160 kG	strength used to levitate a frog <sup>[11]</sup> ←

# Magnetic Levitation



“Field cooling”



# 浮いた 土佐ノ海

## TOSANOUMI (Sumo Wrestler)

Height of Tosanoumi 186cm  
Weight of Tosanoumi 142kg  
Weight of disk 60kg  
Total weight 202kg

As of February '98

# Magnetic Levitation



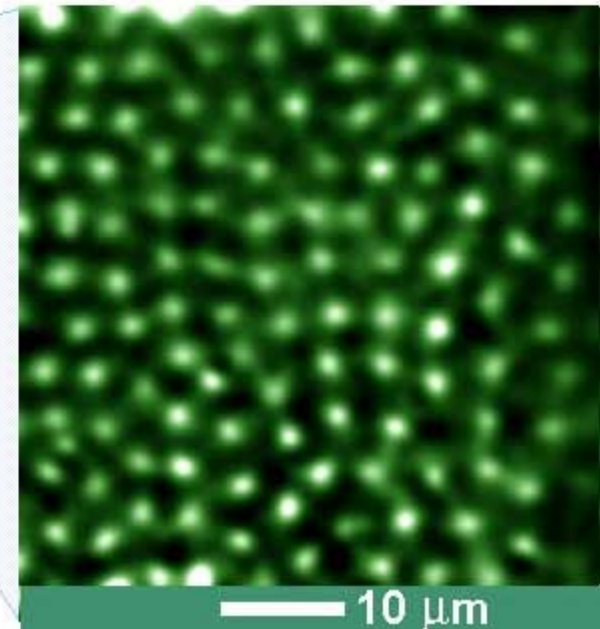
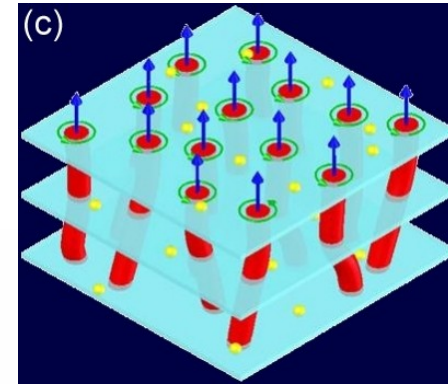
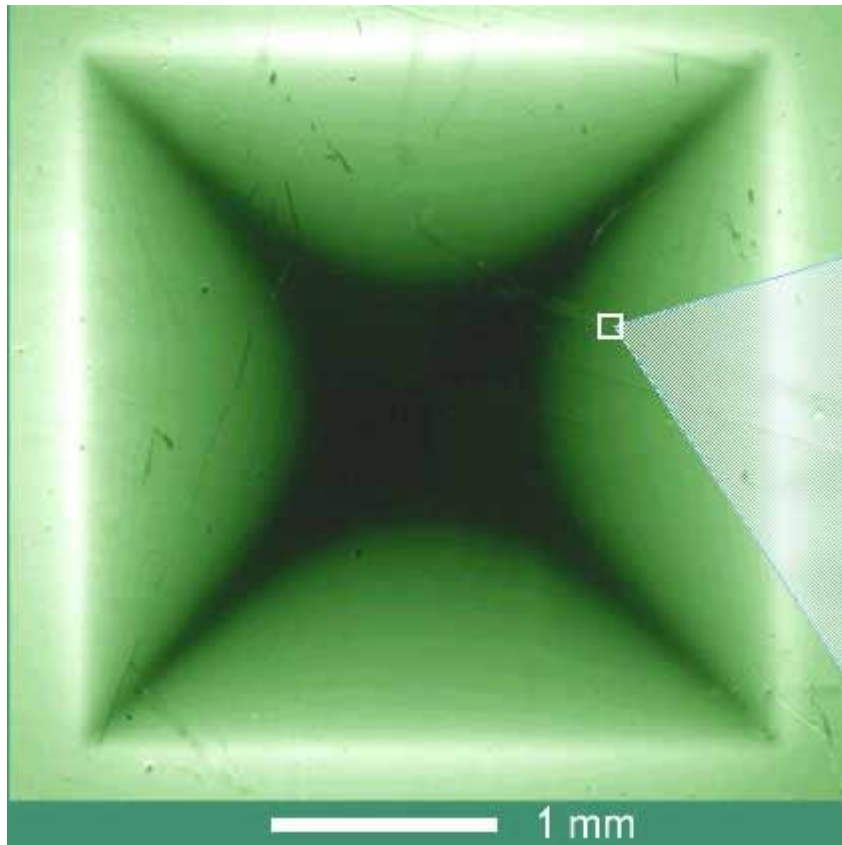
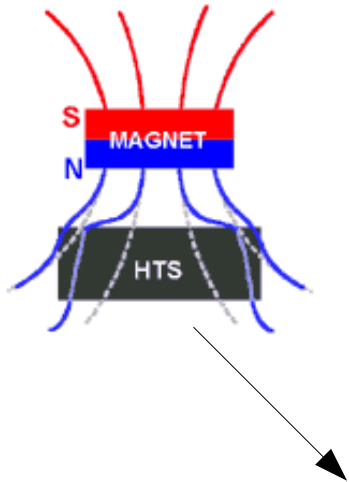
“Zero field cooling”

Some differences with respect to normal diamagnetic levitation:

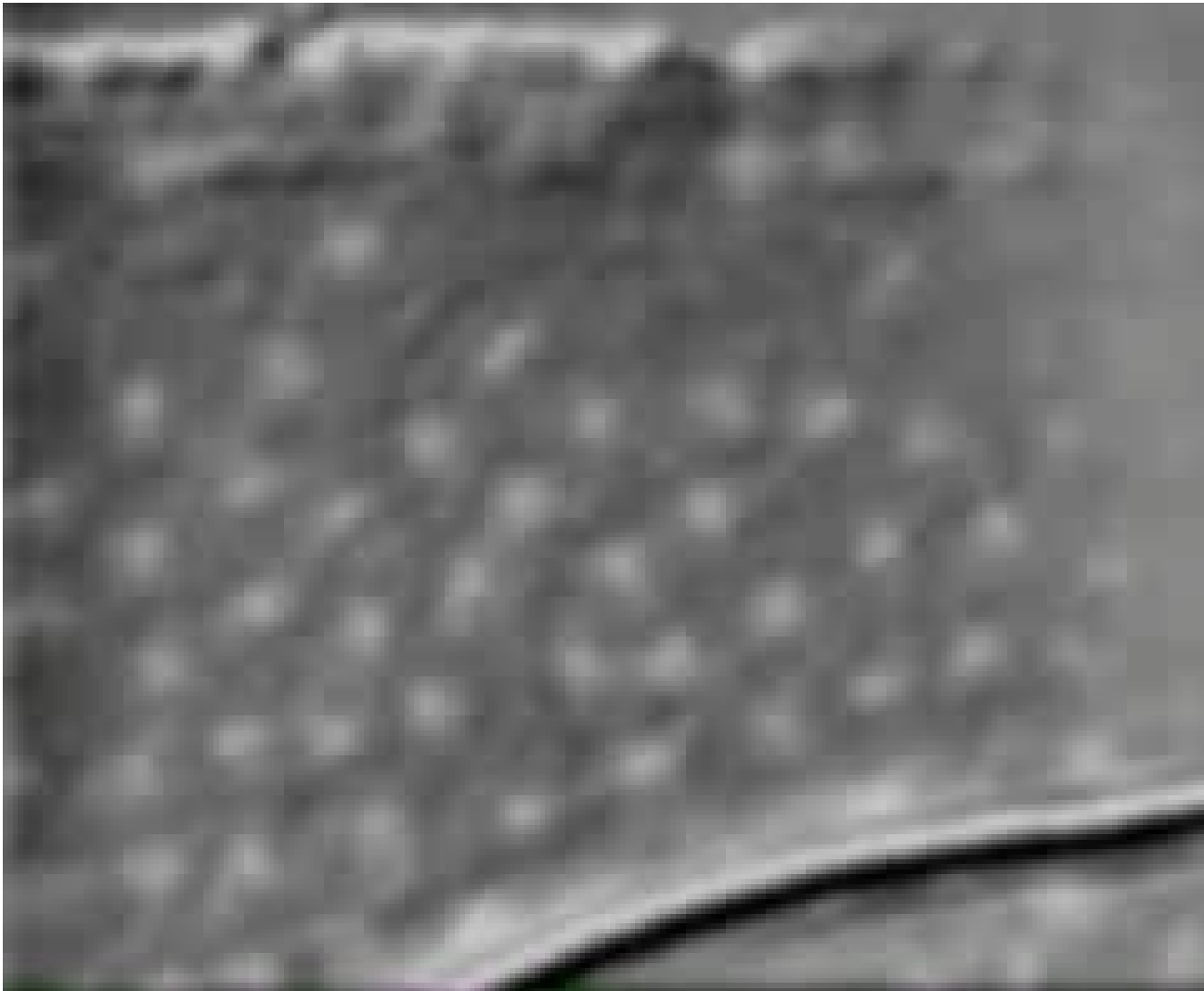
- More effective
- There is also *attraction!*
- There are many static levitating configurations for the magnet!

Where does this increased stability and metastable equilibrium come from?

# Vortices in superconductors



# Vortices in superconductores

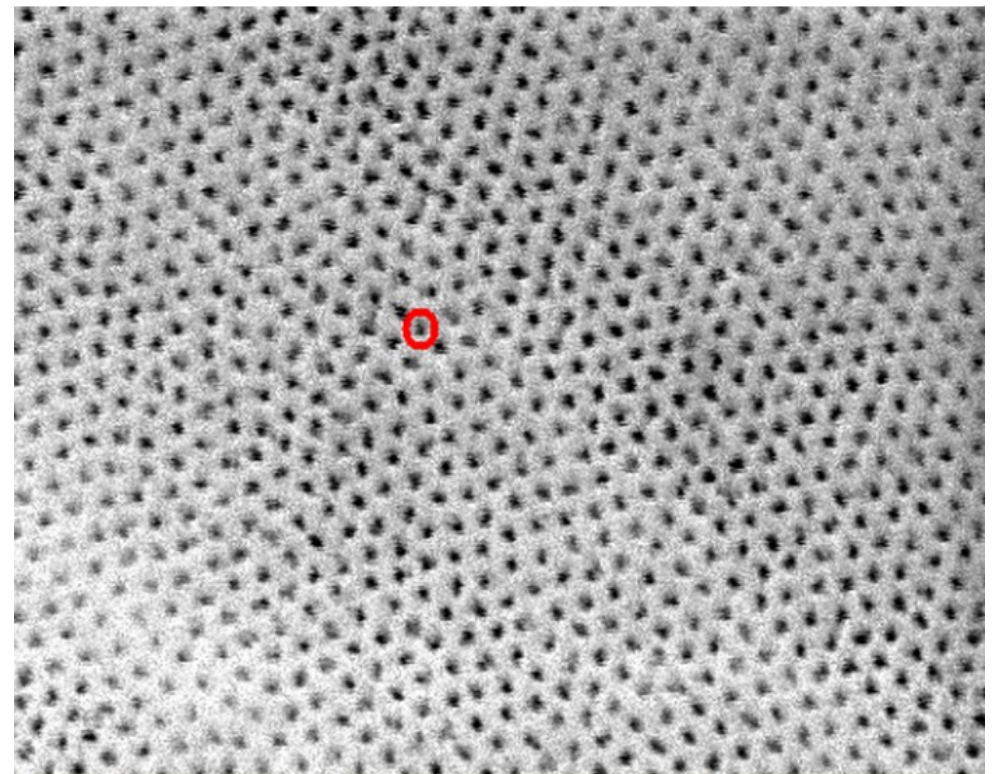
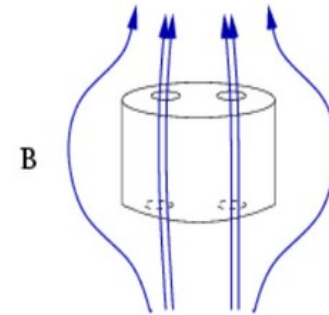
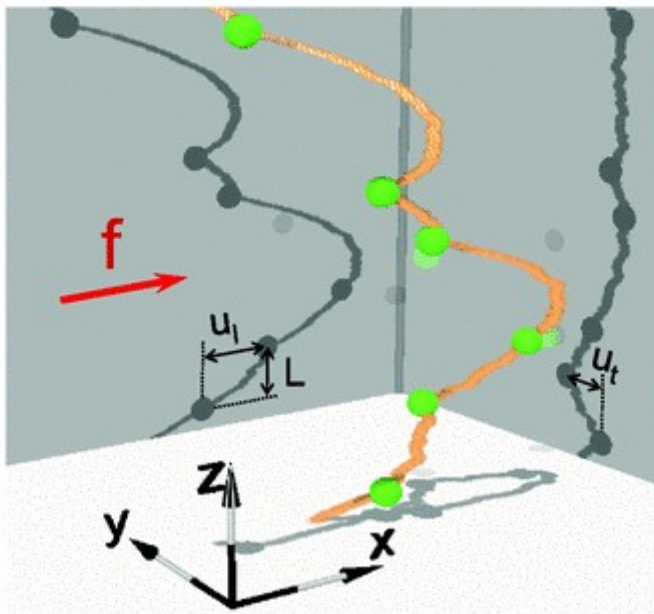
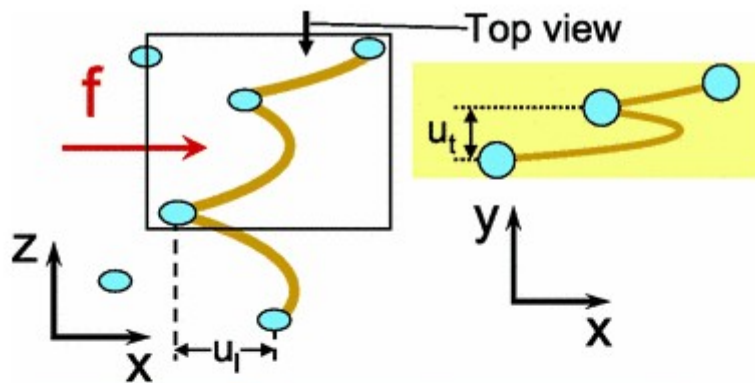


# Vortices in superconductores

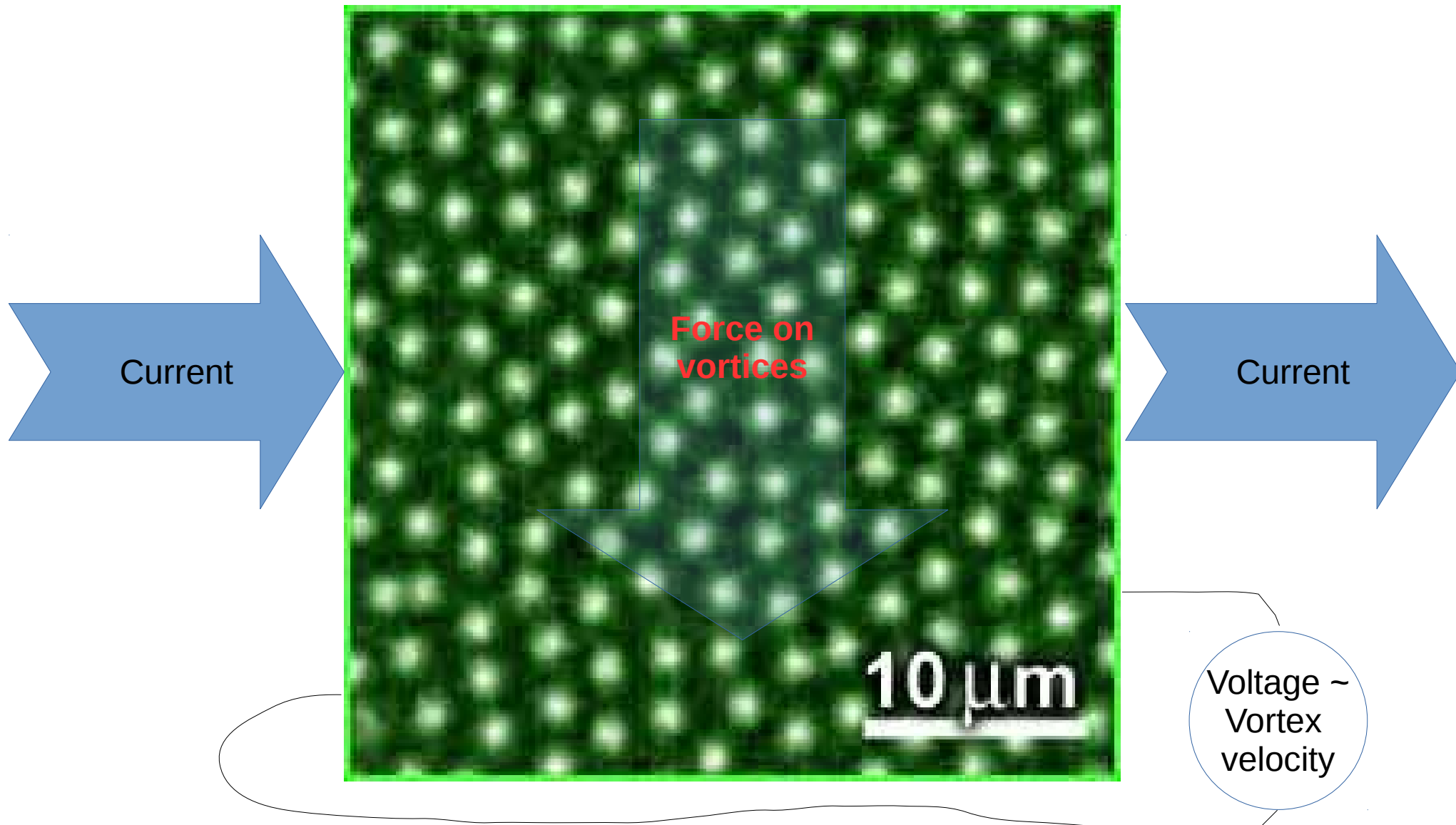


# Vortices in Superconductors

- Pinned elastic strings in 3D

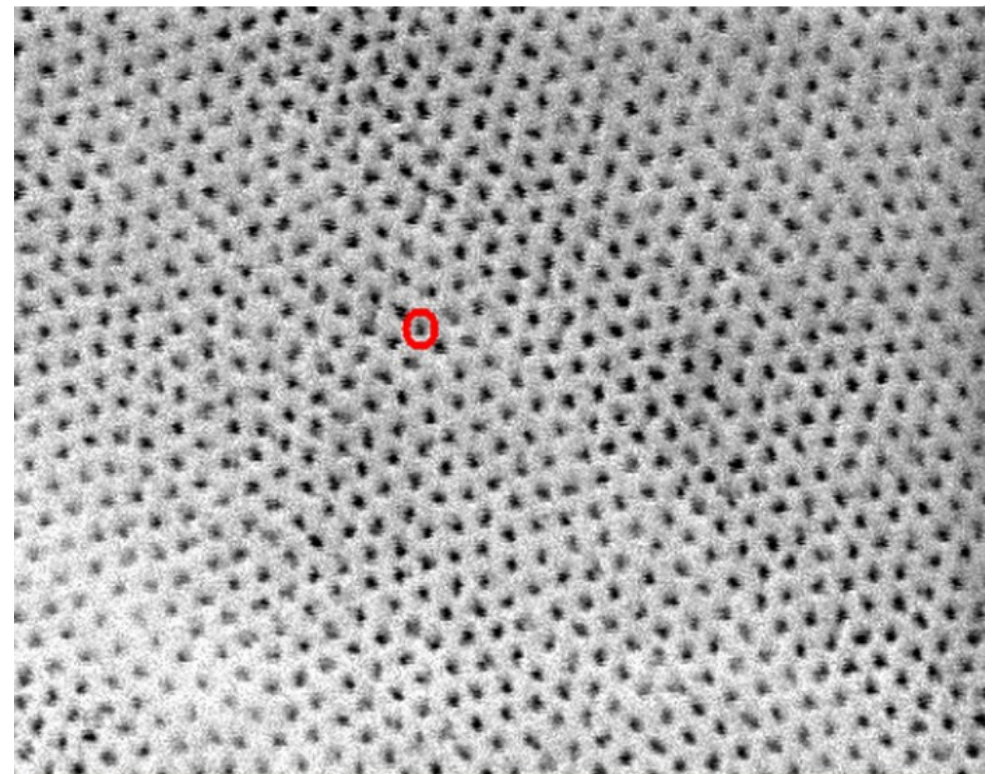
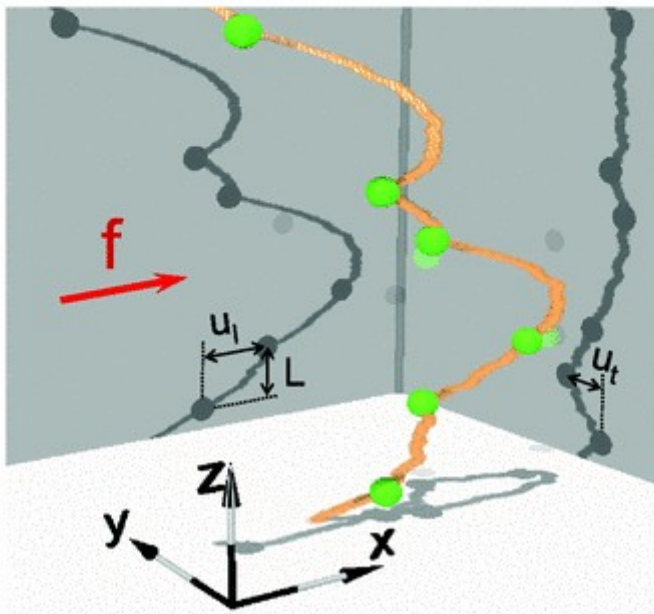
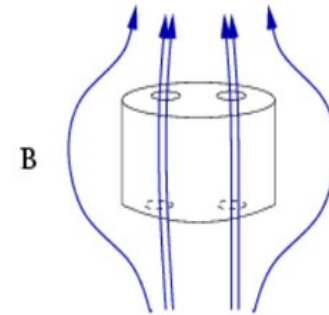
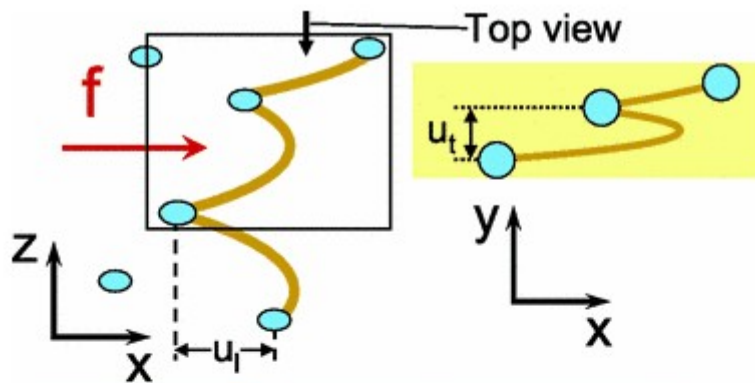


# Vortex motion produces dissipation



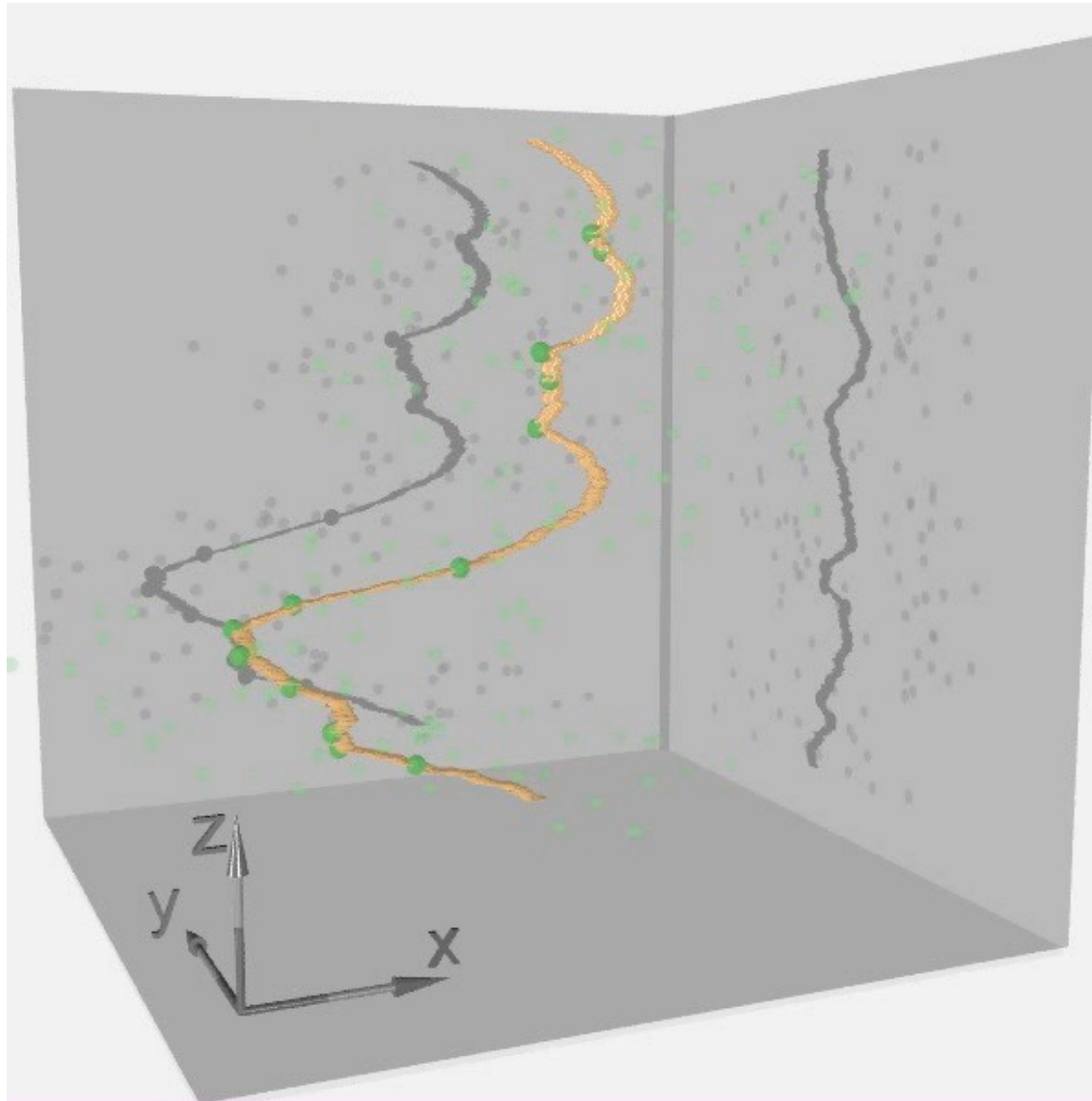
# Vortices in Superconductors

- Pinned elastic strings in 3D





# A moving Vortex simulation

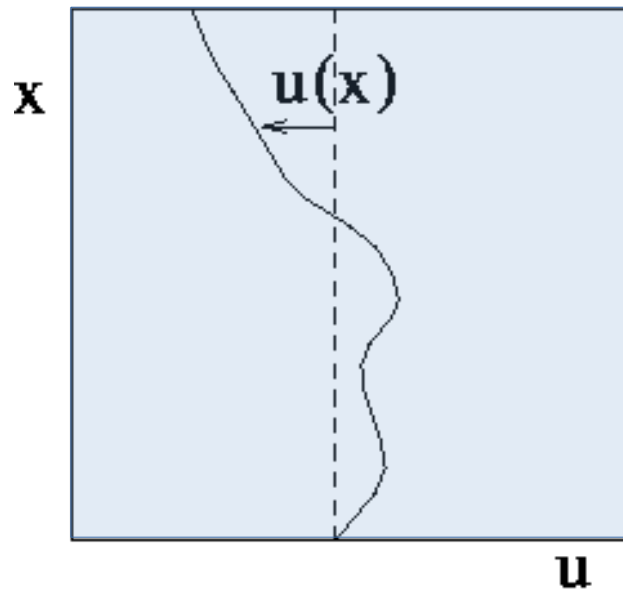


A. Koshelev,  
A. B. Kolton  
PRB 2011

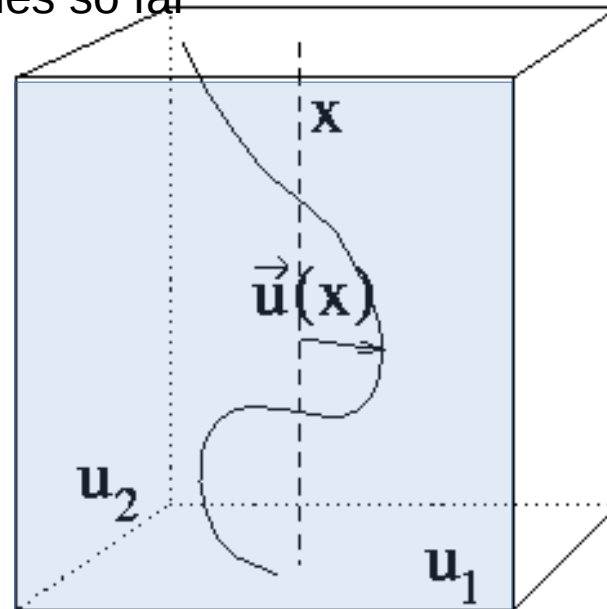
Vortex pinning reduce dissipation in High T<sub>c</sub> superconductors!

# Other Disordered Elastic Manifolds

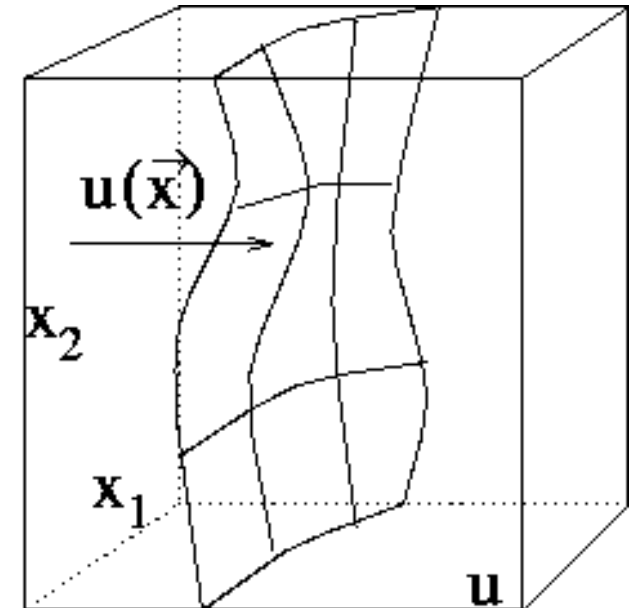
Examples so far



$N=d=1$   
(a)



$N=2 \ d=1$   
(b)



$N=1 \ d=2$   
(c)

- Ferromagnetic Domain walls
- Ferroelectric Domain walls
- Contact lines of liquids
- Etc...

- Vortices in superconductors.
- Polymers in random media.
- Etc.

- Two dimensional domain walls
- **Earthquakes?**

# Earthquakes!

# Earthquakes!

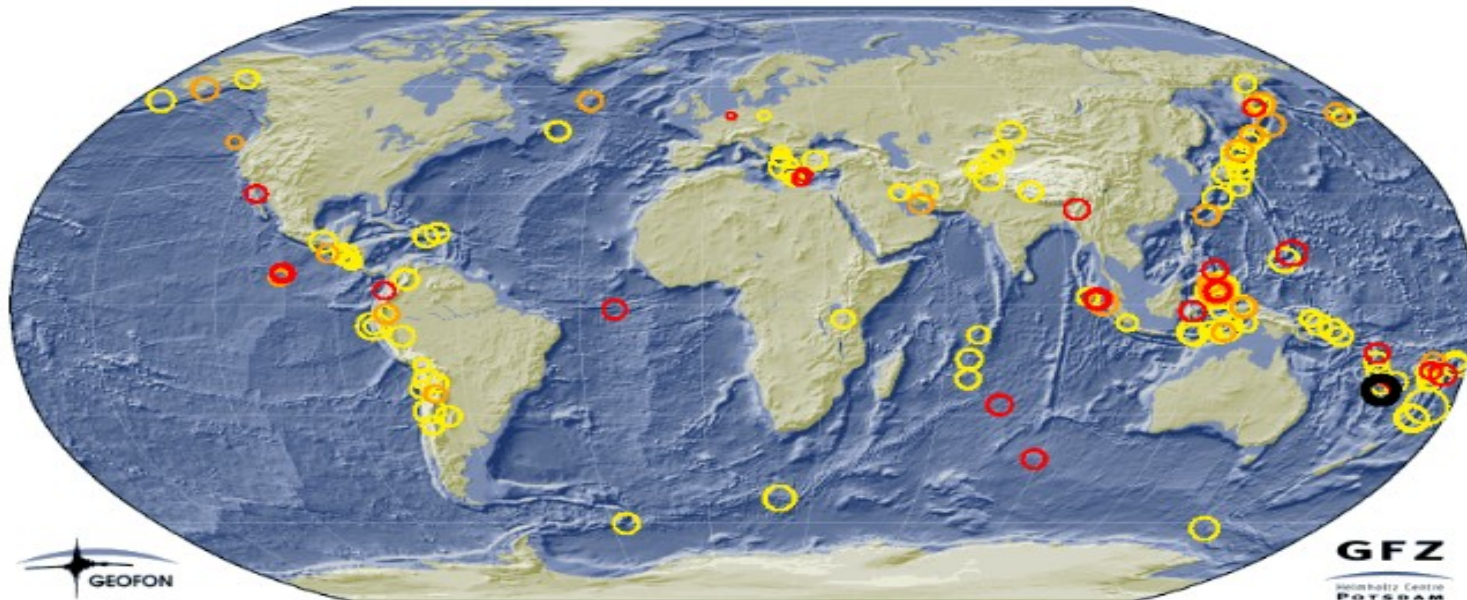


Valdivia (Chile), 22/5/1960; 9,5° Richter scale; largest registered earthquake in the world.

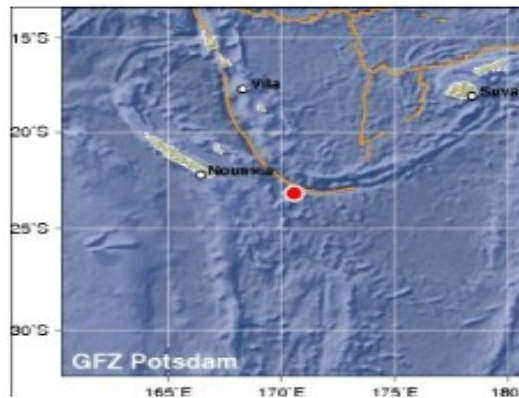


# Where?

## Automatic GEOFON Global Seismic Monitor



The displayed events are within the last **24 hours** / **1-4 days** / **4-14 days** .



### Most recent large event: **Southeast of Loyalty Islands**

Magnitude: **5.8**

Origin time: **2009-02-24 12:39:47.8 UTC**

Epicenter: **170.57°E 23.20°S**

Depth: **71 km**

Location: **automatic**

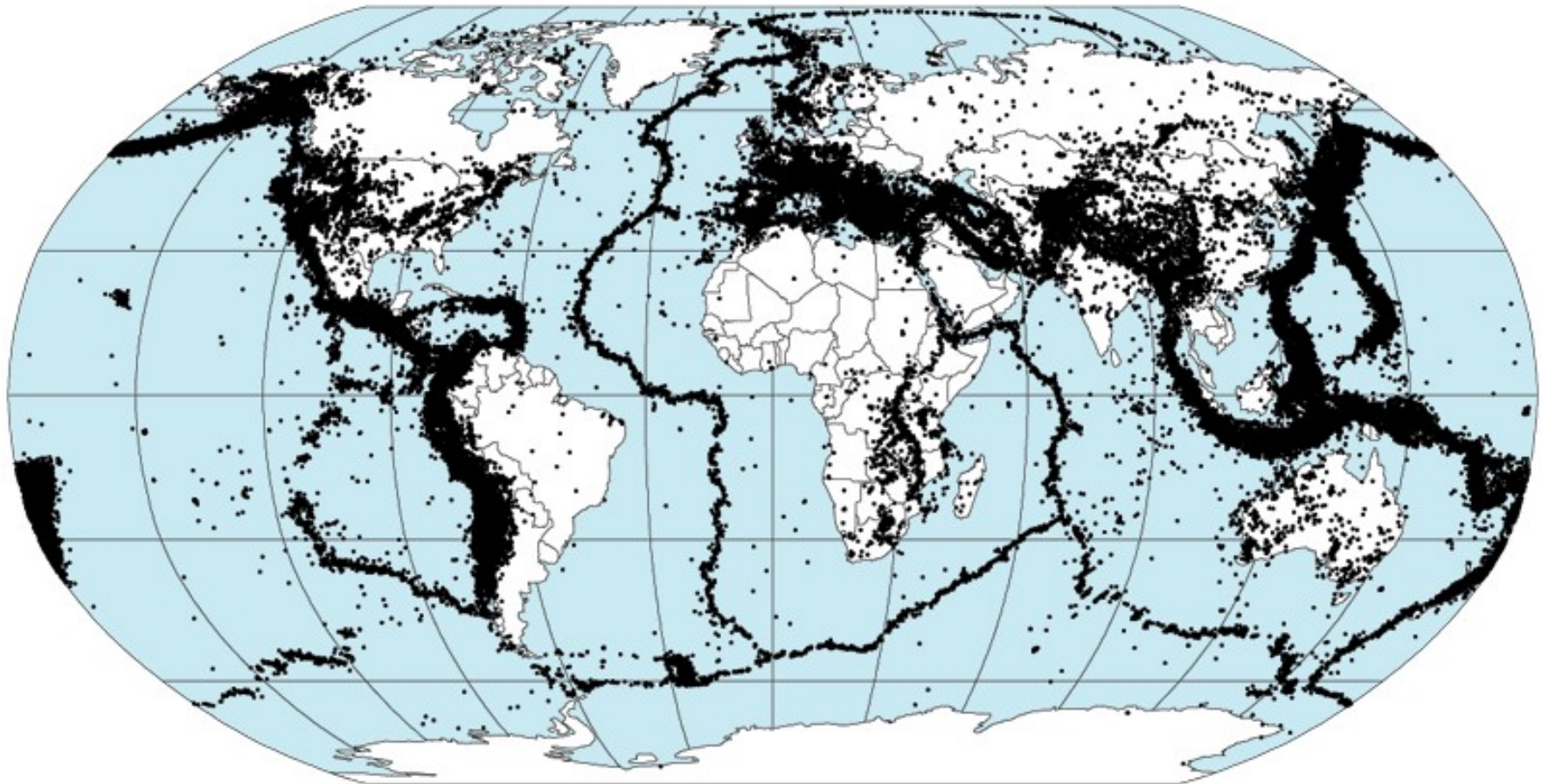
See also:

- The [specific page](#) for this event
- The [complete list](#) of automatic GEOFON alerts

# Where?

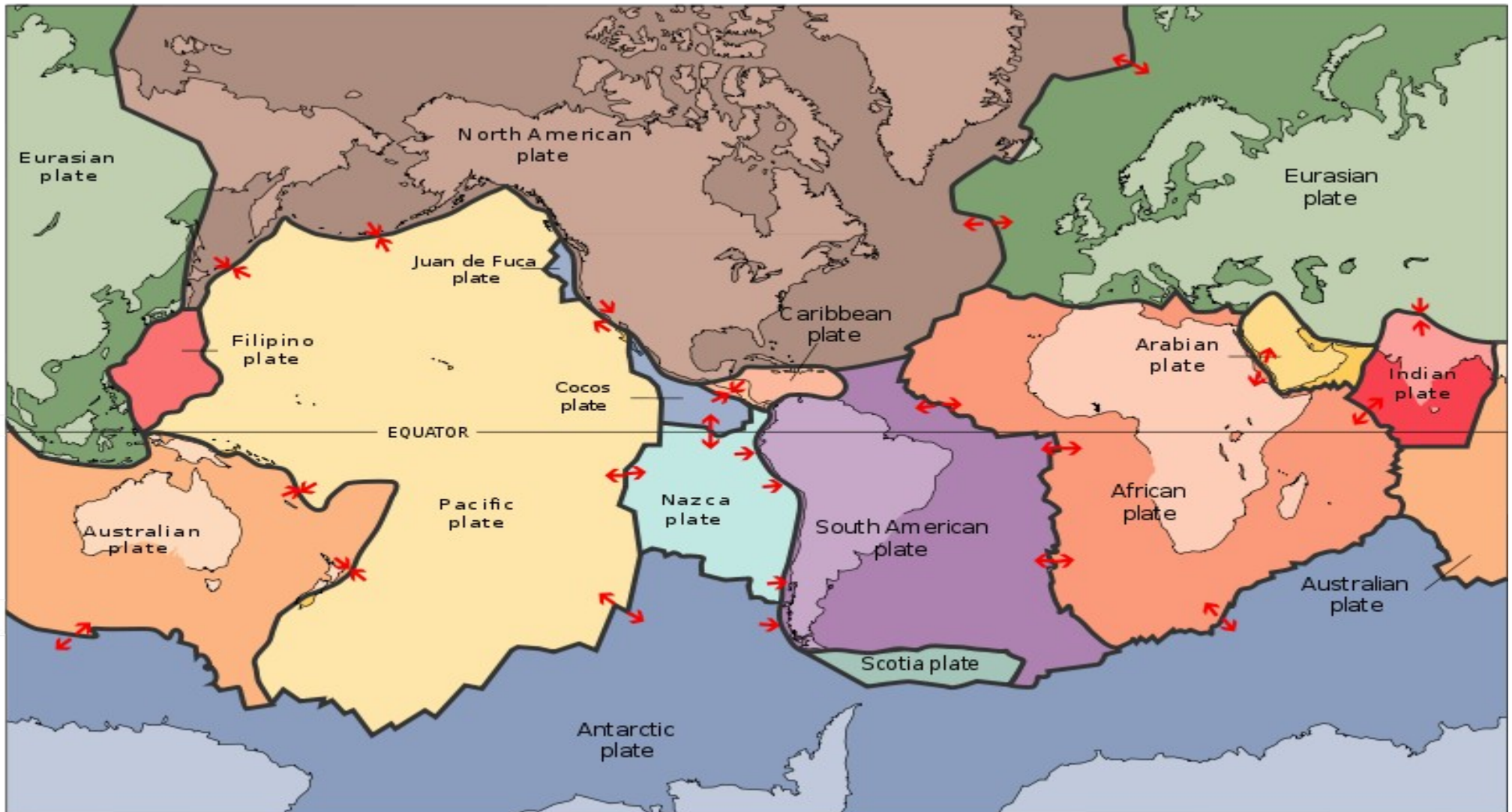
Preliminary Determination of Epicenters

358,214 Events, 1963 - 1998



<http://earthquake.usgs.gov/earthquakes/>

# Plate Tectonics (1960!)

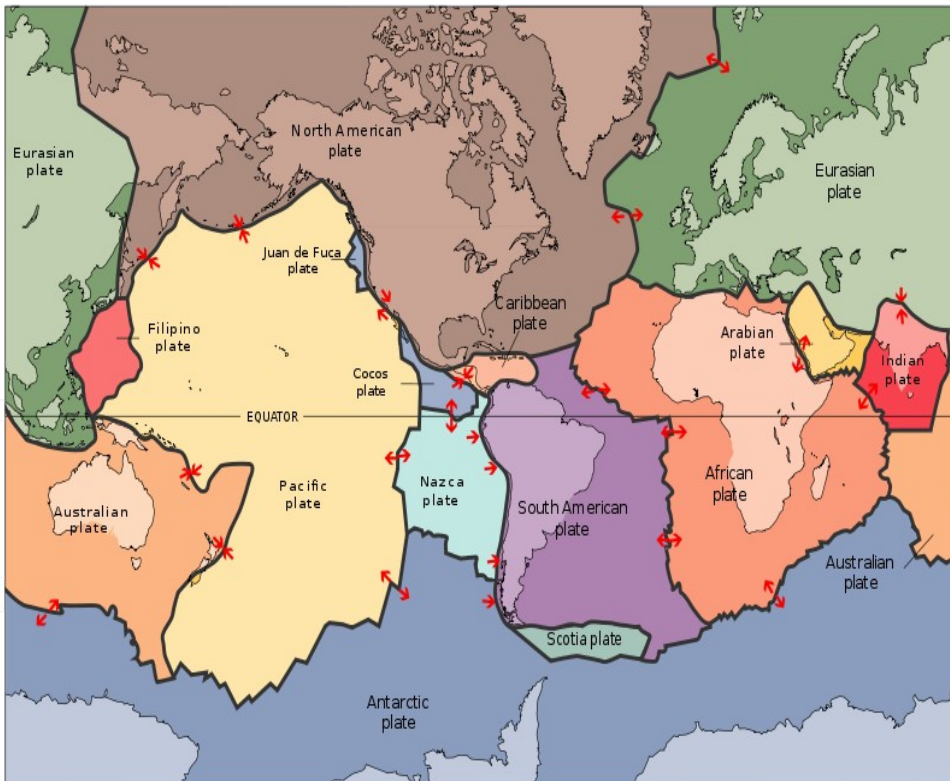


<http://earthquake.usgs.gov/earthquakes/>



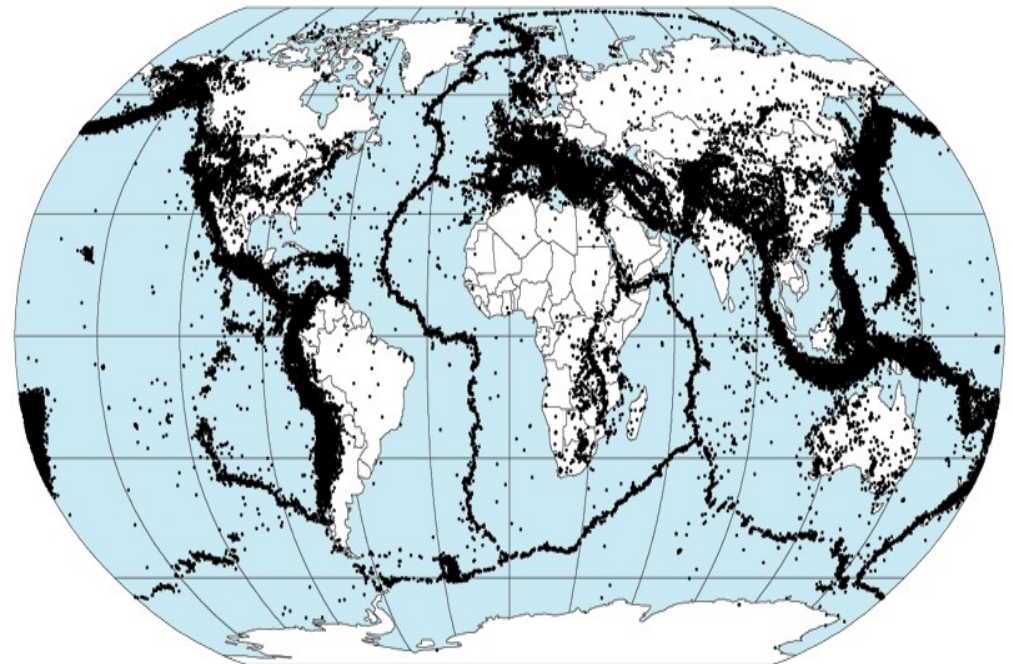
# Plate tectonics & Earthquakes

Placas

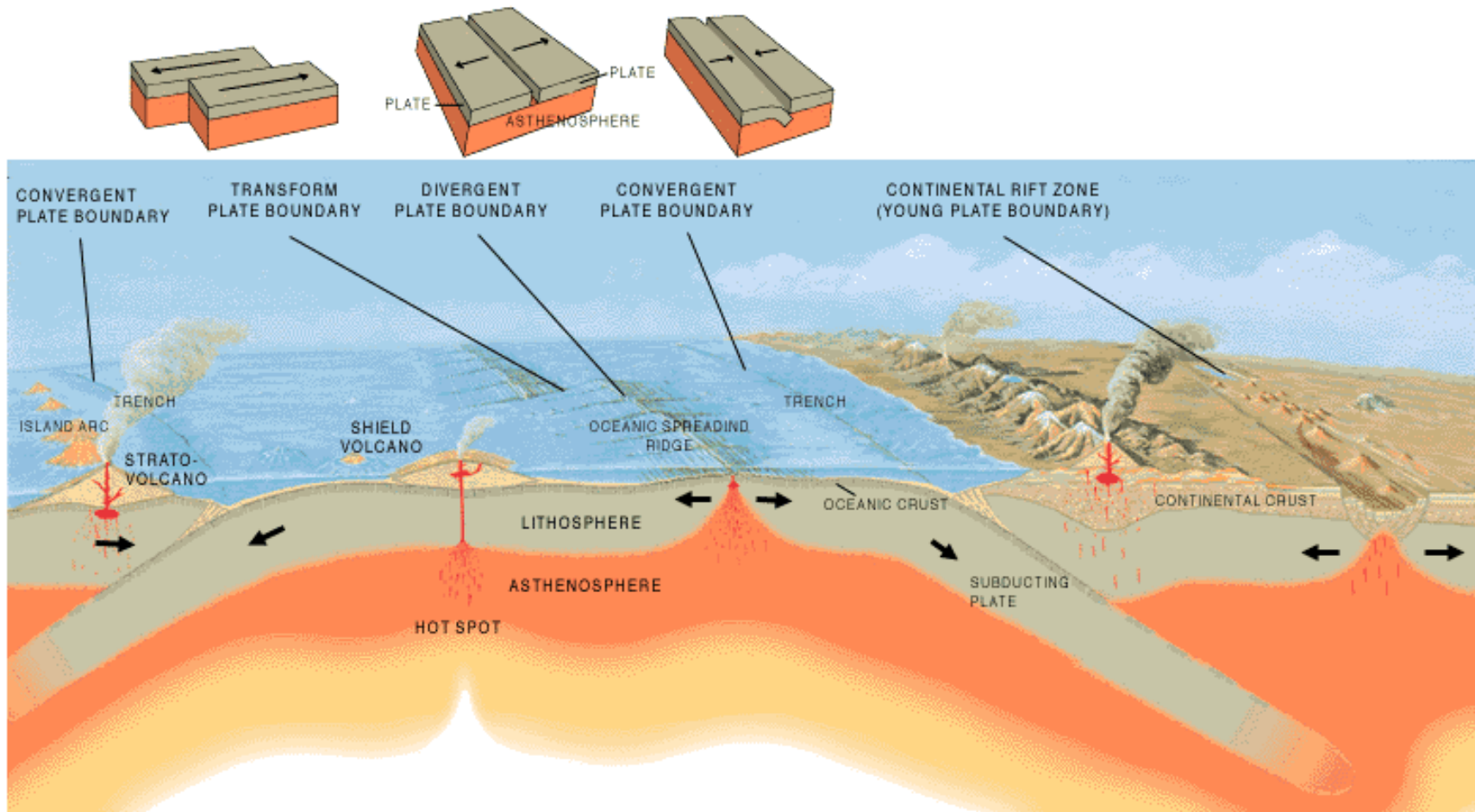


Terremotos

Preliminary Determination of Epicenters  
358,214 Events, 1963 - 1998

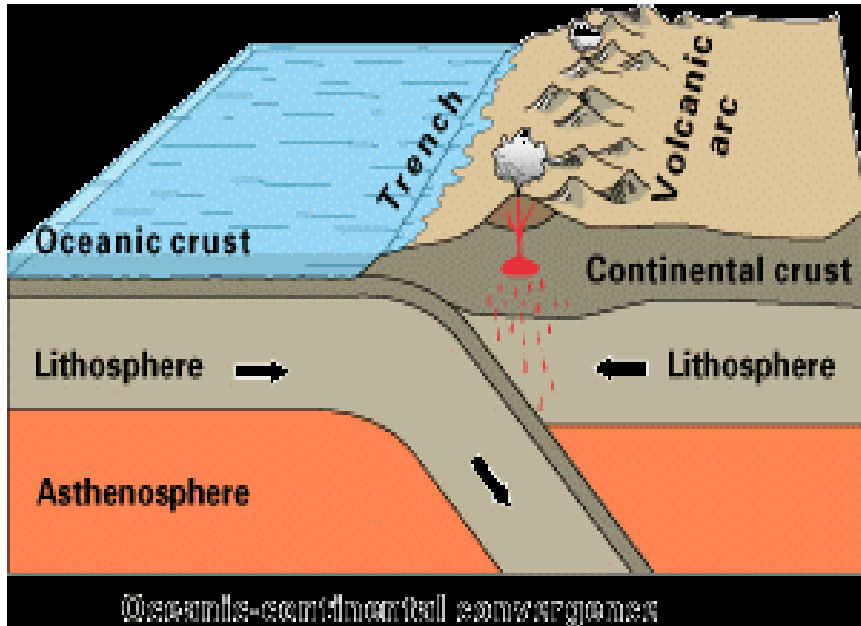


# Plates borders

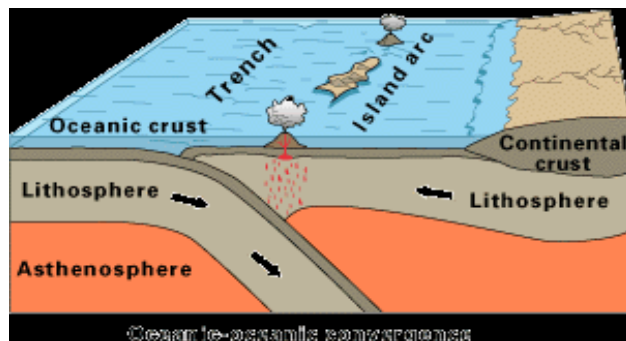


# Convergent borders

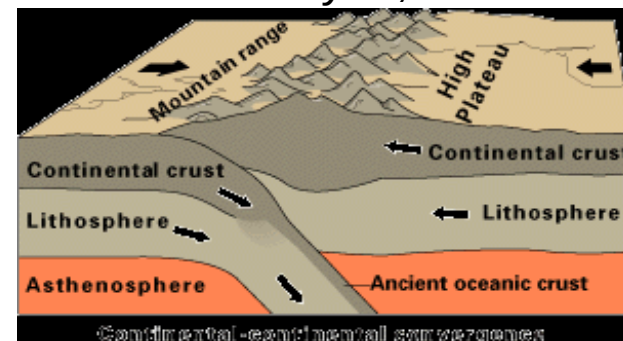
Los Andes



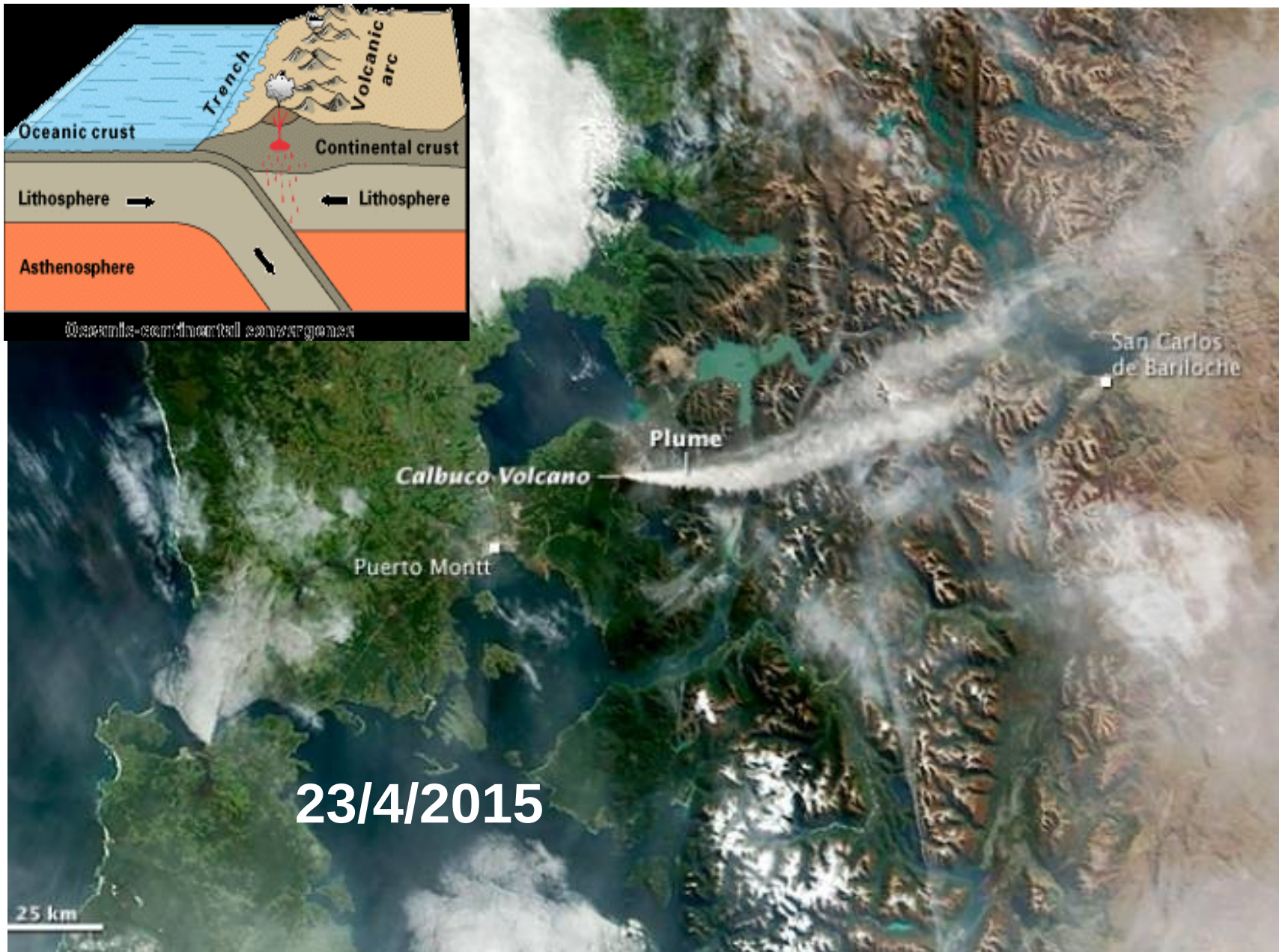
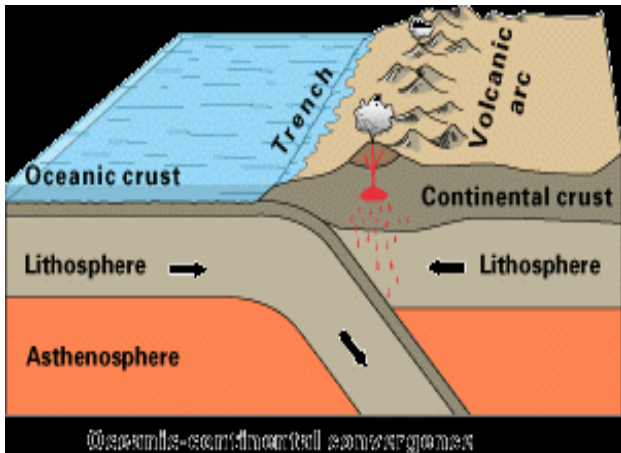
Marianas Islands, etc



Himalayas, etc



# Convergent borders



# Calbuco's volcanic plume

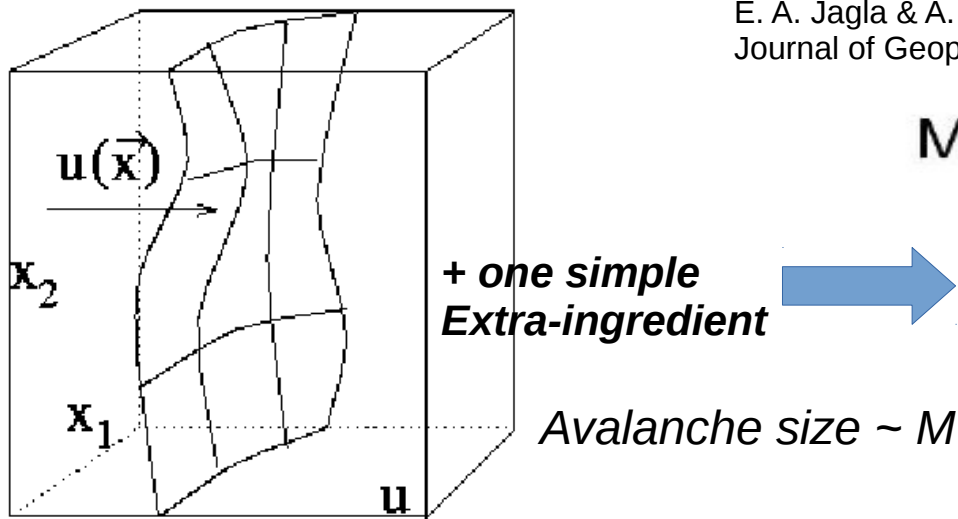


22/4/2015

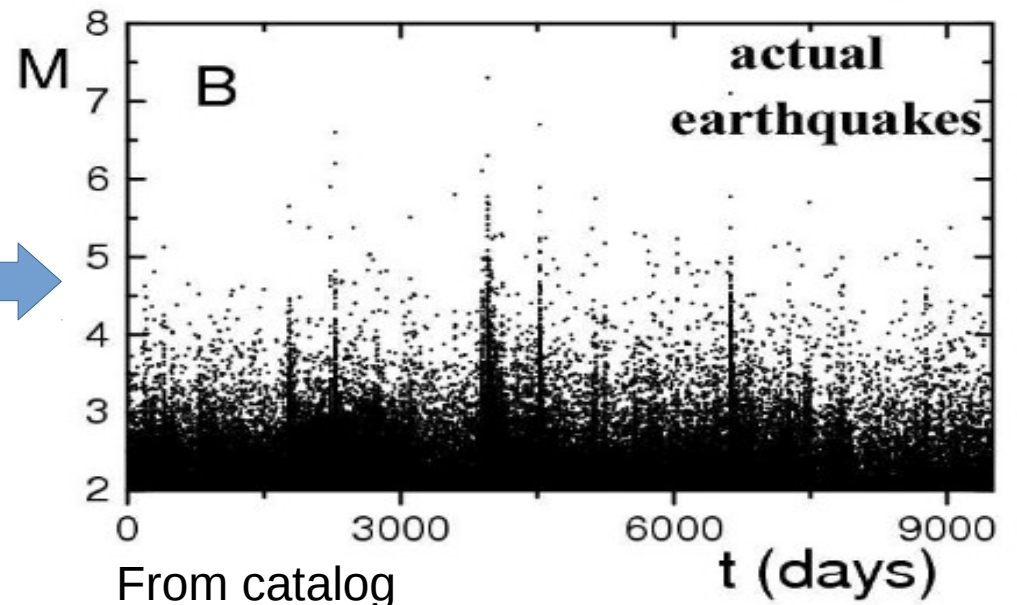
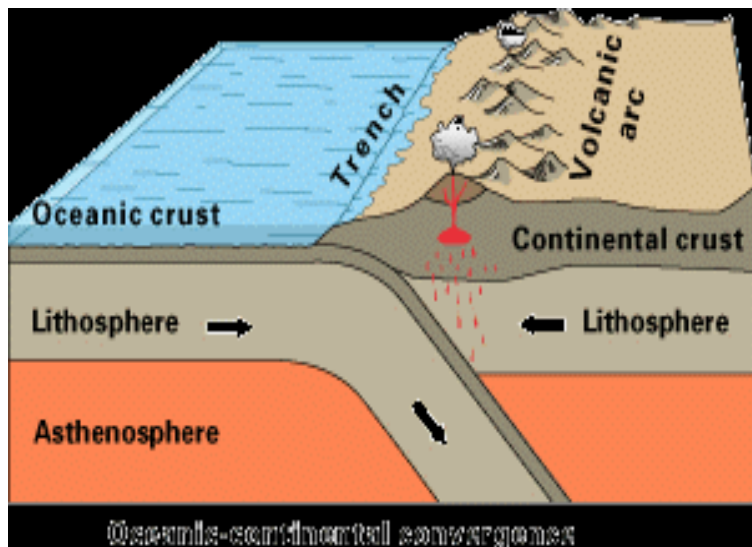
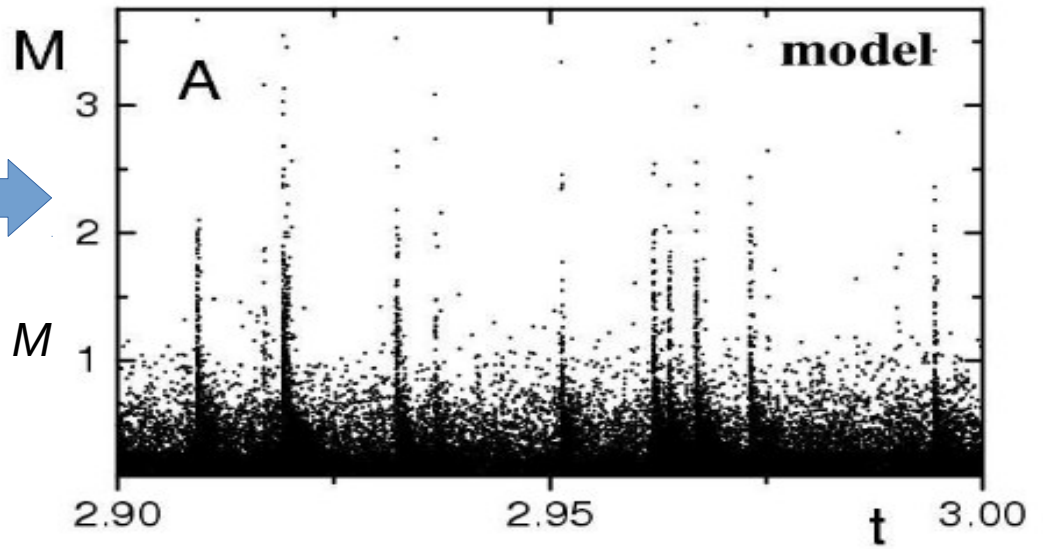
*Bariloche.Org*

# Simple model for earthquake statistics?

E. A. Jagla & A. B. Kolton, *A mechanism for spatial and temporal earthquake clustering*, Journal of Geophysical Research (2010)



$N=1$   $d=2$

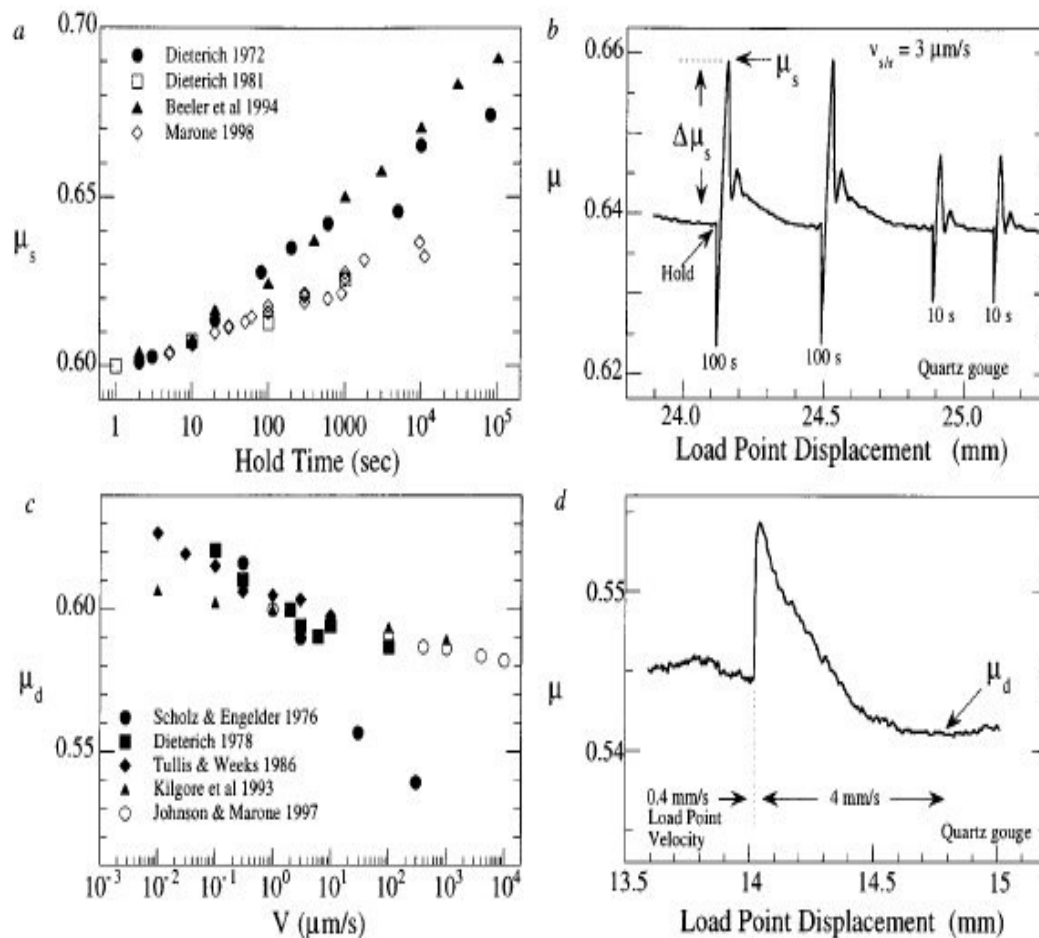


# A mechanism for spatial and temporal earthquake clustering

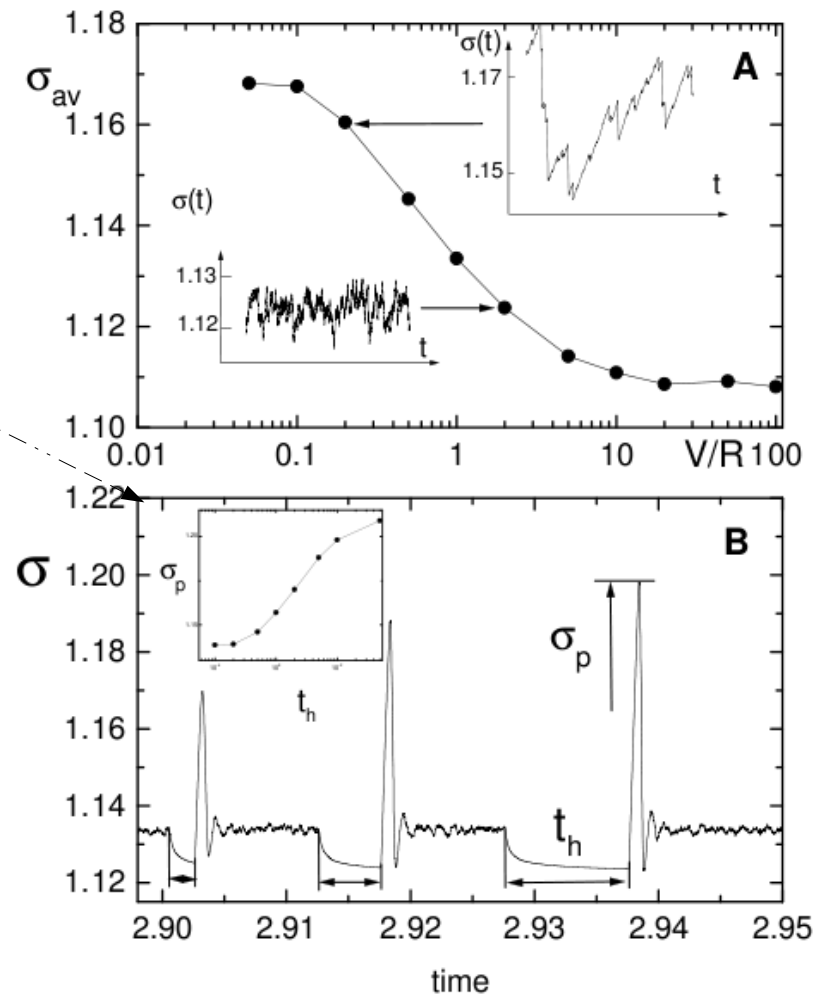
E. A. Jagla, A. B. Kolton, J. Geophys. Res. 115, B05312 (2010)

## Original Motivation: Frictional properties of the model

### Rock Friction Experiments (Lab scale)



### Modified Depinning Model



# A mechanism for spatial and temporal earthquake clustering

E. A. Jagla, A. B. Kolton, J. Geophys. Res. 115, B05312 (2010)

## EVENTS

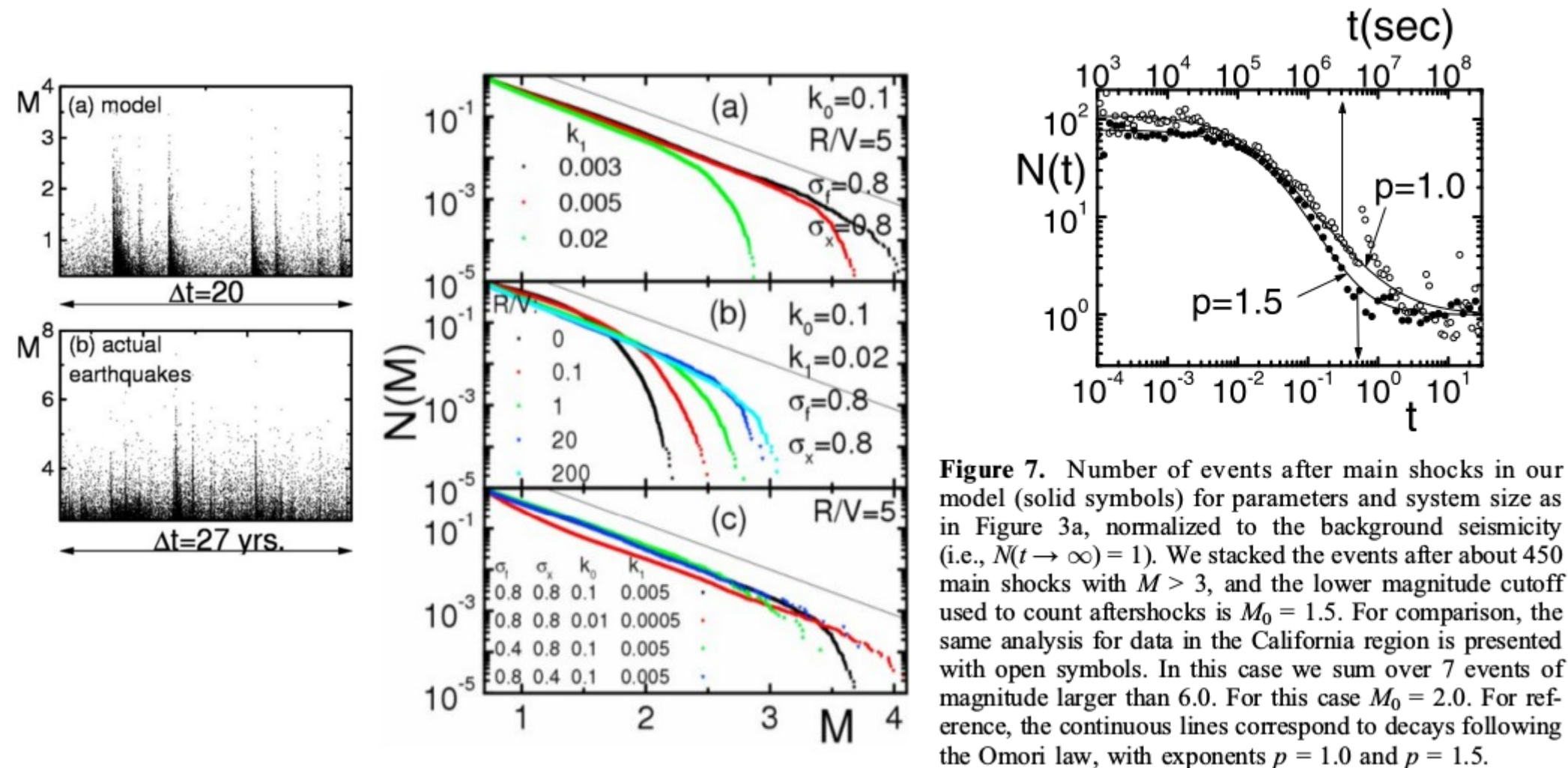
$$M = 2/3 \log_{10} S$$

## Gutenberg-Richter

$$N(M) \sim 10^{-bM}$$

## Omori

$$N(t) = A/(t + c)^p + N_0$$

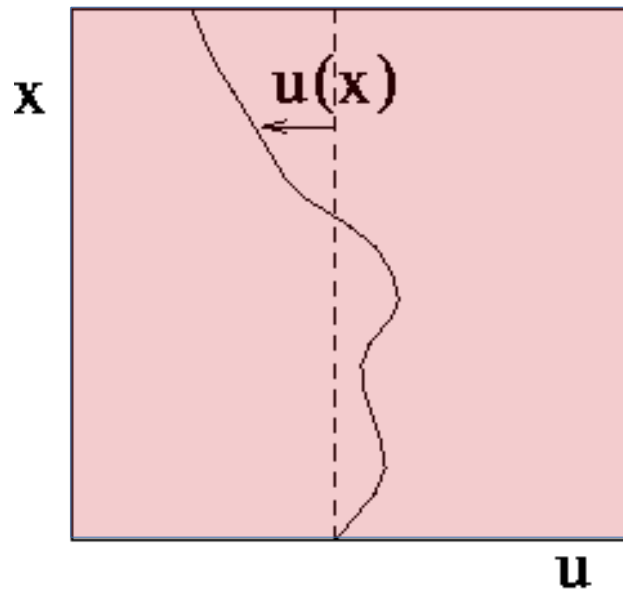


**Figure 7.** Number of events after main shocks in our model (solid symbols) for parameters and system size as in Figure 3a, normalized to the background seismicity (i.e.,  $N(t \rightarrow \infty) = 1$ ). We stacked the events after about 450 main shocks with  $M > 3$ , and the lower magnitude cutoff used to count aftershocks is  $M_0 = 1.5$ . For comparison, the same analysis for data in the California region is presented with open symbols. In this case we sum over 7 events of magnitude larger than 6.0. For this case  $M_0 = 2.0$ . For reference, the continuous lines correspond to decays following the Omori law, with exponents  $p = 1.0$  and  $p = 1.5$ .



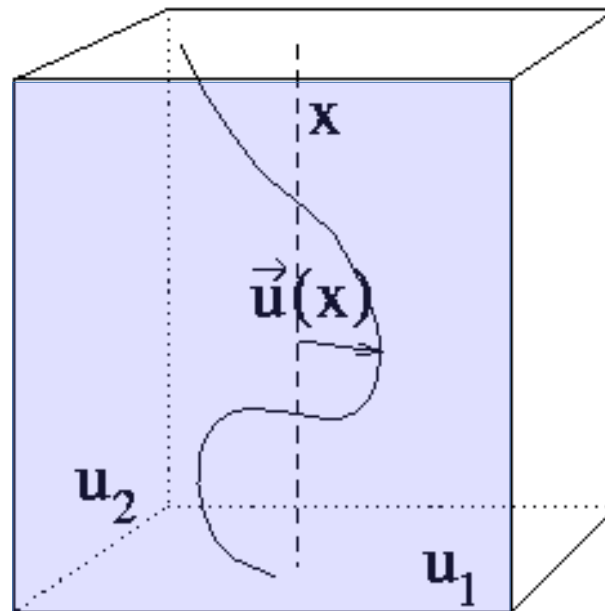
# Disordered Elastic Manifolds

Examples so far



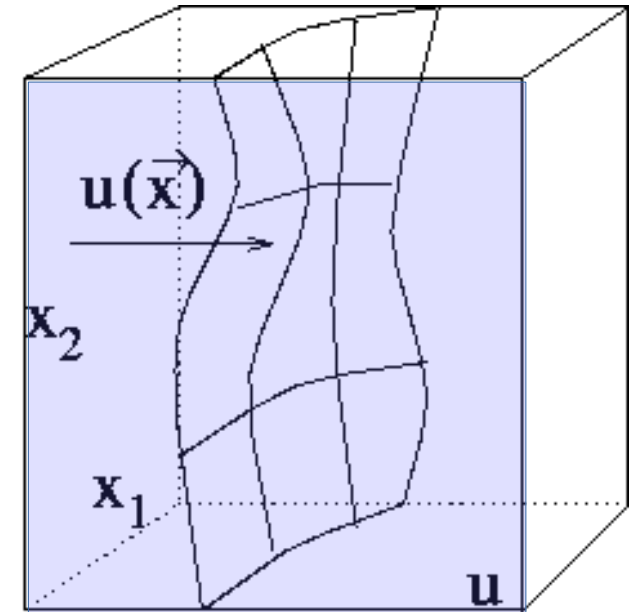
$N=d=1$

(a)



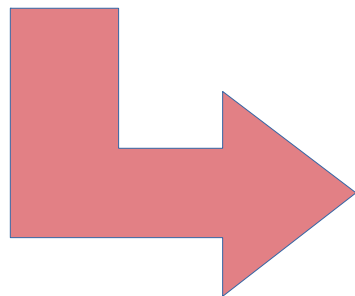
$N=2$   $d=1$

(b)



$N=1$   $d=2$

(c)

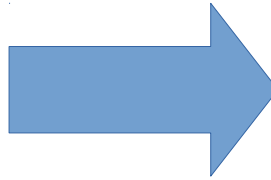


*Elastic Strings in 2d Random Media*

# Models

Main ingredients

- Elasticity
- Disorder
- Drive



Model to capture their universal interplay?

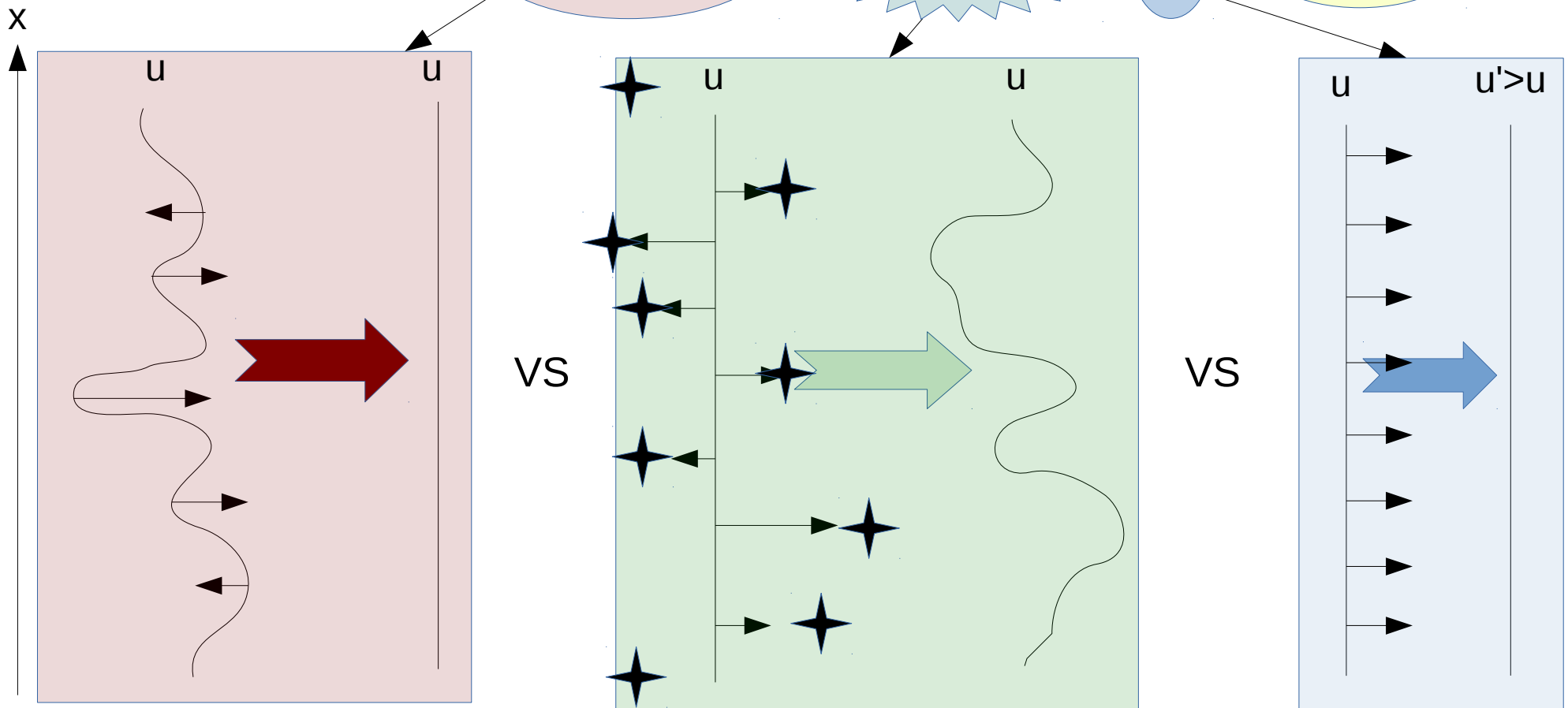
*If dynamics has universal features, we should choose **minimal models** to predict them accurately.*

# A minimal model?

... to capture the competition between disorder and elasticity  
and to predict the response to a driving field

Temperature

$$\gamma \partial_t u(x, t) = c \partial_x^2 u(x, t) + F_p(u, x) + f + \eta(x, t)$$



*smoothes*

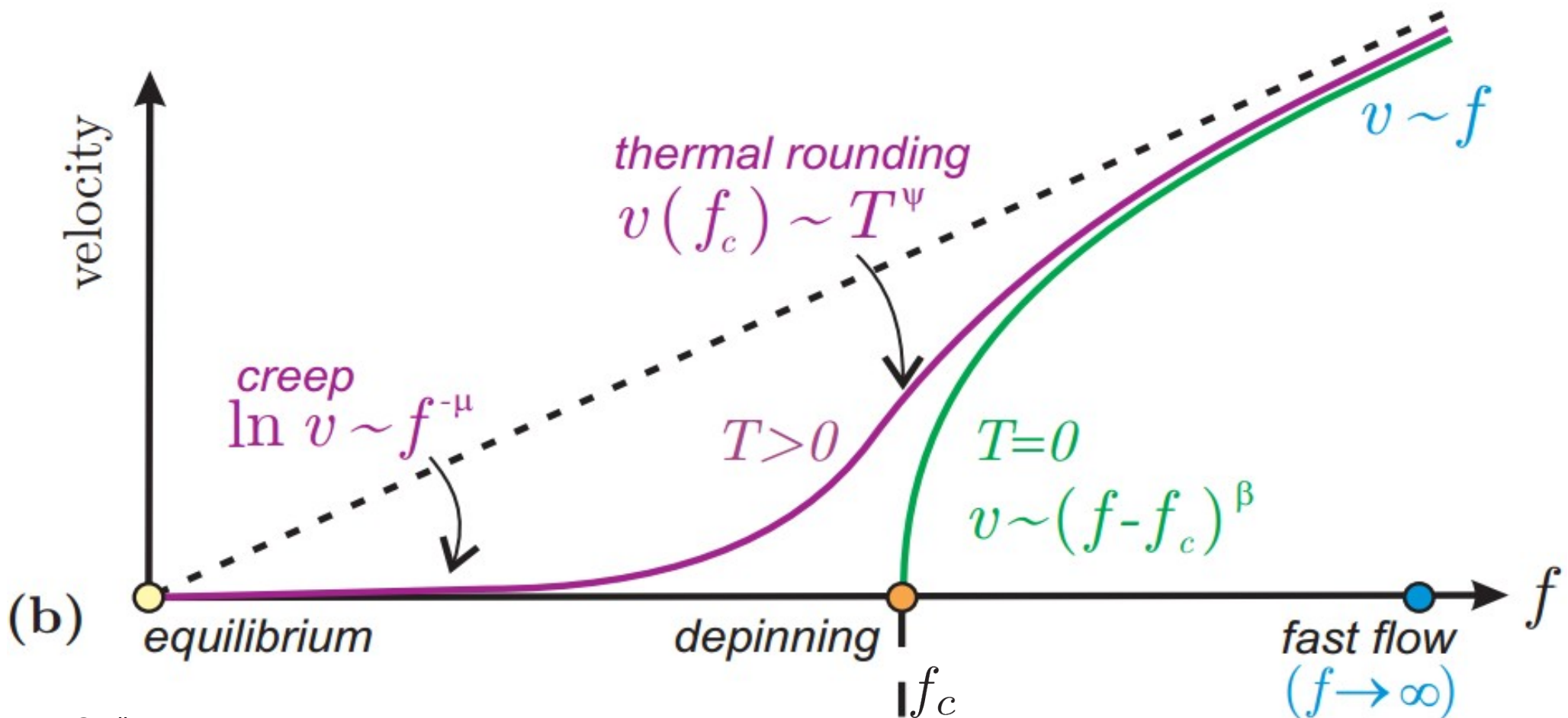
*distorts and pins*

*pushes*

[Note: far more simple than a micro-magnetic model... too simple?]

# Predictions from the minimal model: Dynamical Regimes

$$\gamma \partial_t u(x, t) = c \partial_x^2 u(x, t) + F_p(u, x) + f + \eta(x, t)$$



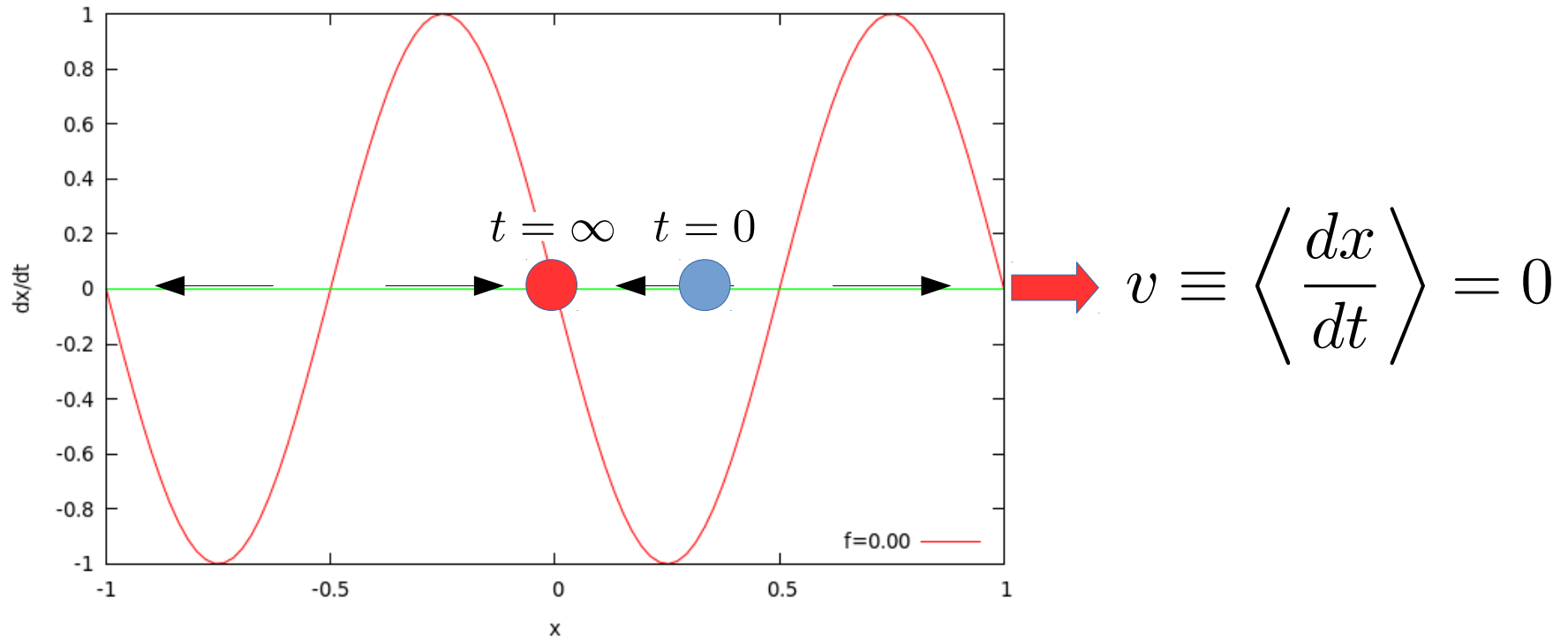
- Scaling Arguments
- Functional renormalization group
- **Numerical Simulations**

Three-reference steady states

# Depinning

- One particle in a simple periodic potential ( $T=0$ )

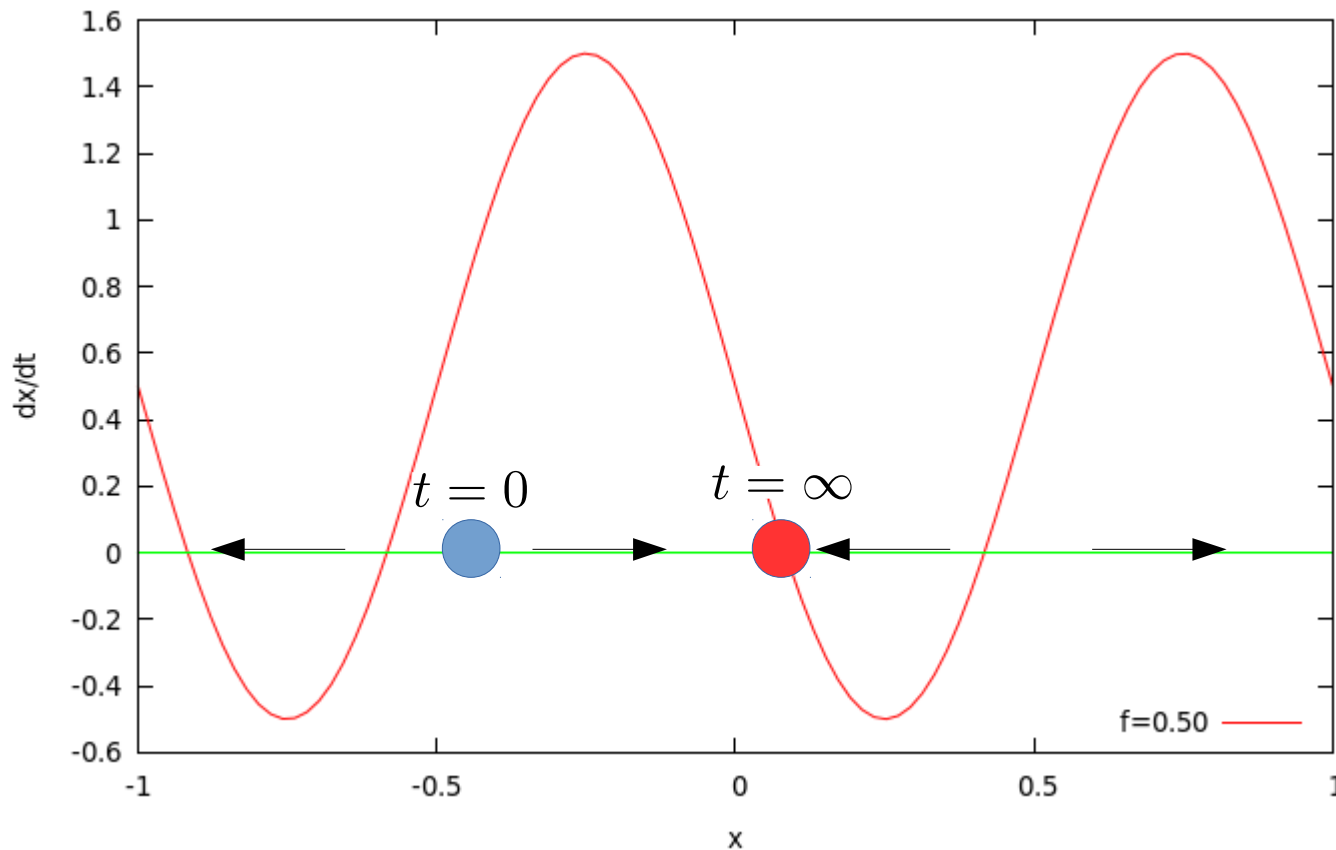
$$\frac{dx}{dt} = -\sin(2\pi x) + f$$



# Depinning

- One particle in a simple periodic potential

$$\frac{dx}{dt} = -\sin(2\pi x) + f$$

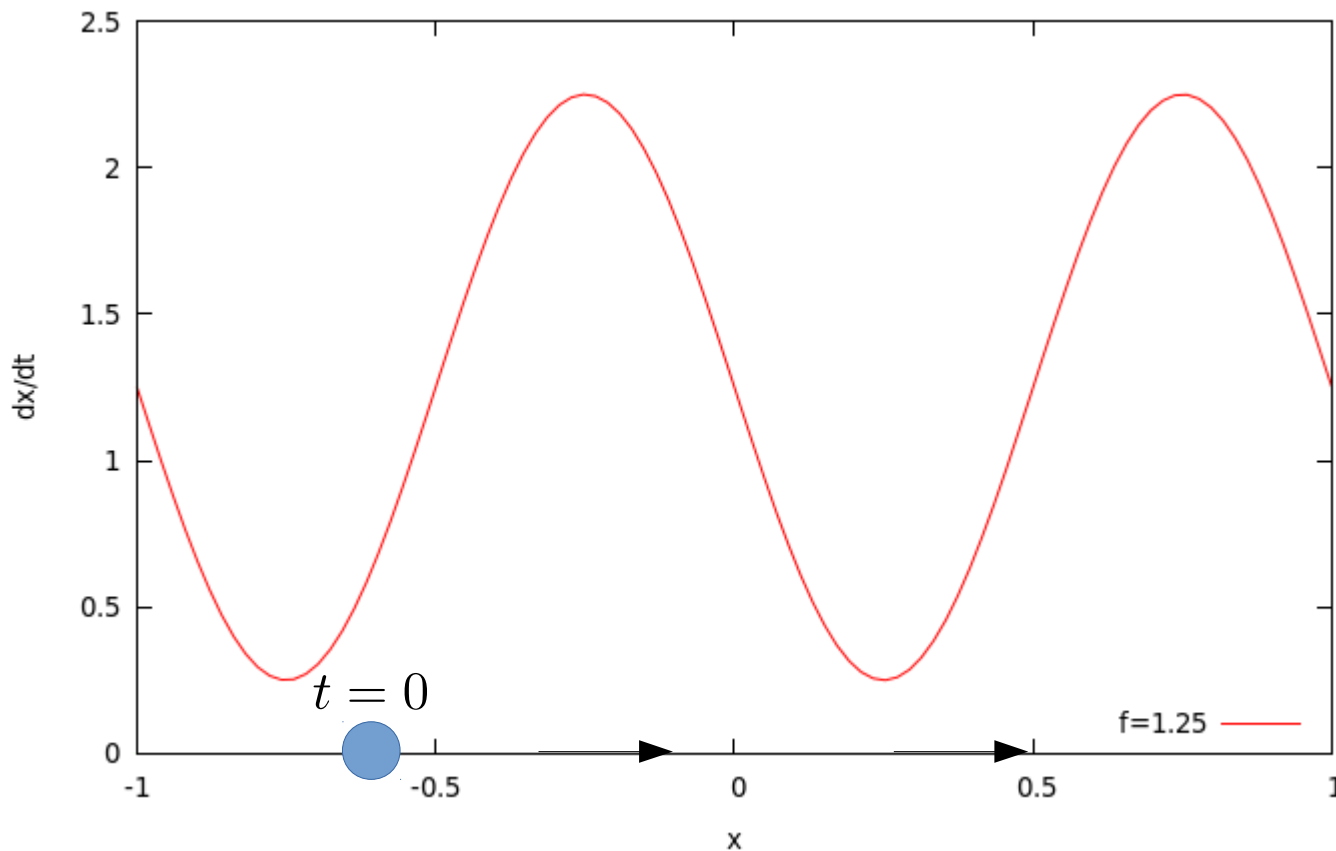


$$\Rightarrow v \equiv \left\langle \frac{dx}{dt} \right\rangle = 0$$

# Depinning

- One particle in a simple periodic potential

$$\frac{dx}{dt} = -\sin(2\pi x) + f$$



$$\rightarrow v \equiv \left\langle \frac{dx}{dt} \right\rangle > 0$$

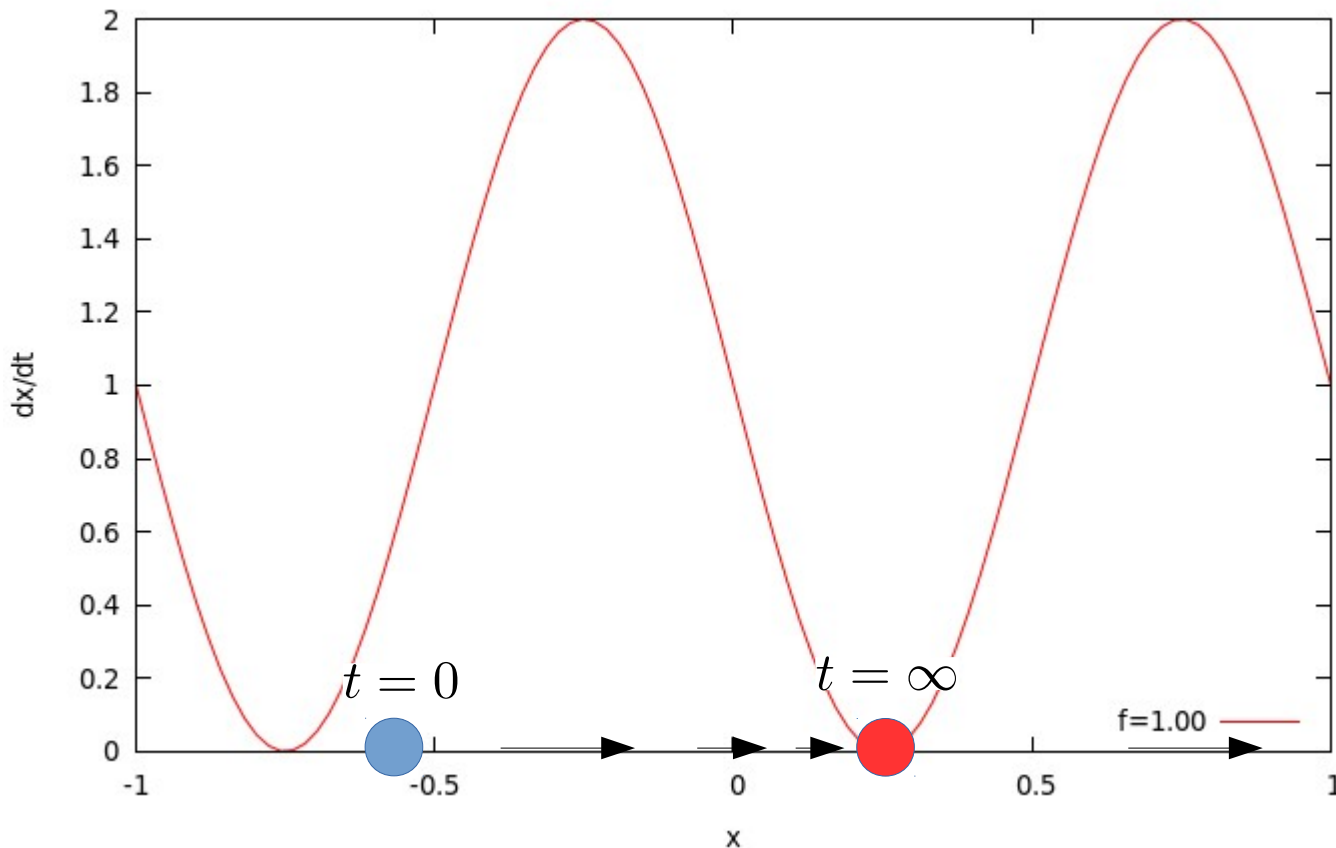
$\forall t > 0$



# Depinning

- One particle in a simple periodic potential

$$\frac{dx}{dt} = -\sin(2\pi x) + f$$



→  $v \equiv \left\langle \frac{dx}{dt} \right\rangle = 0$

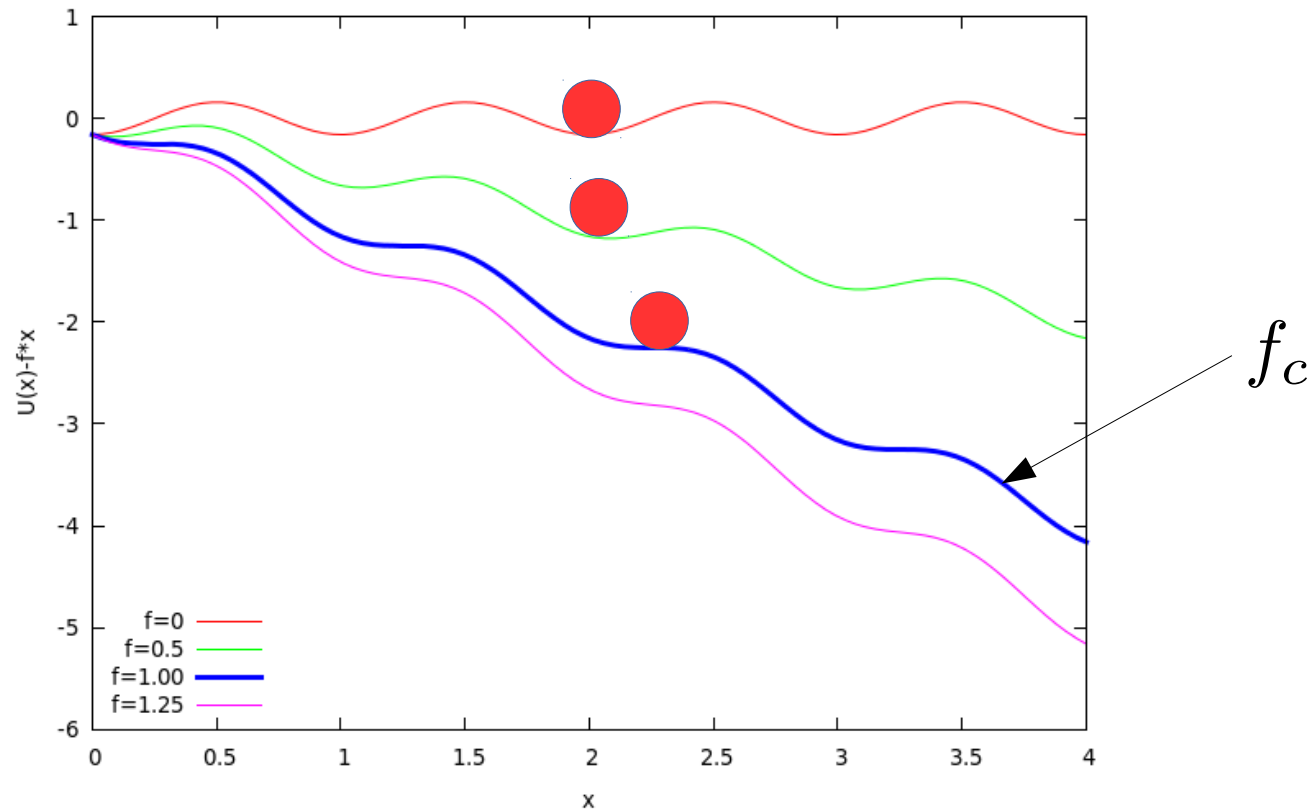
Critical Force  $f_c = 1$



# Depinning

- One particle in a simple periodic potential

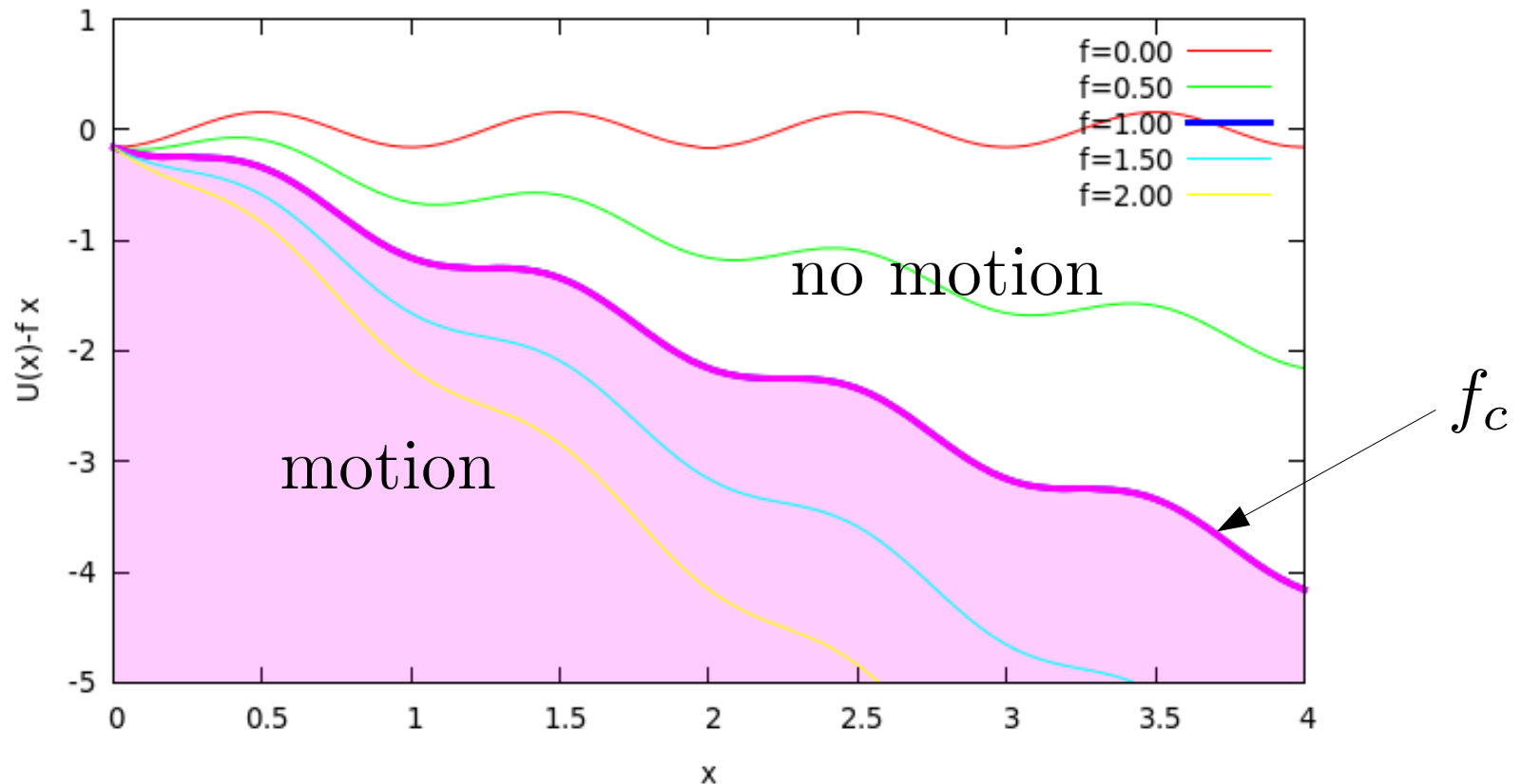
$$\frac{dx}{dt} = -\sin(2\pi x) + f = -\frac{d}{dx}U(x) - fx$$



# Depinning

- One particle in a simple periodic potential

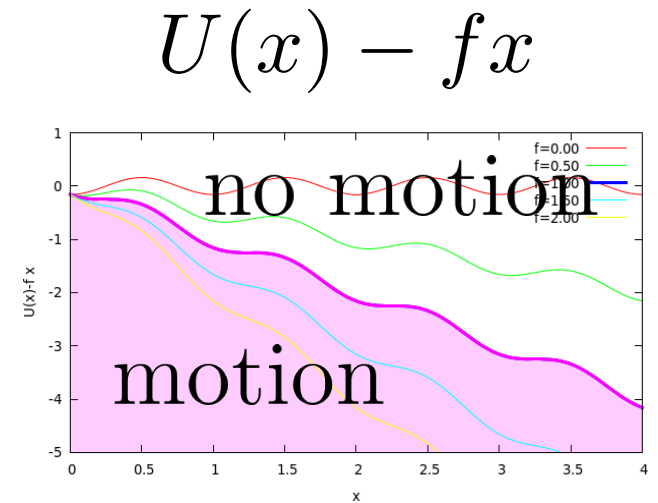
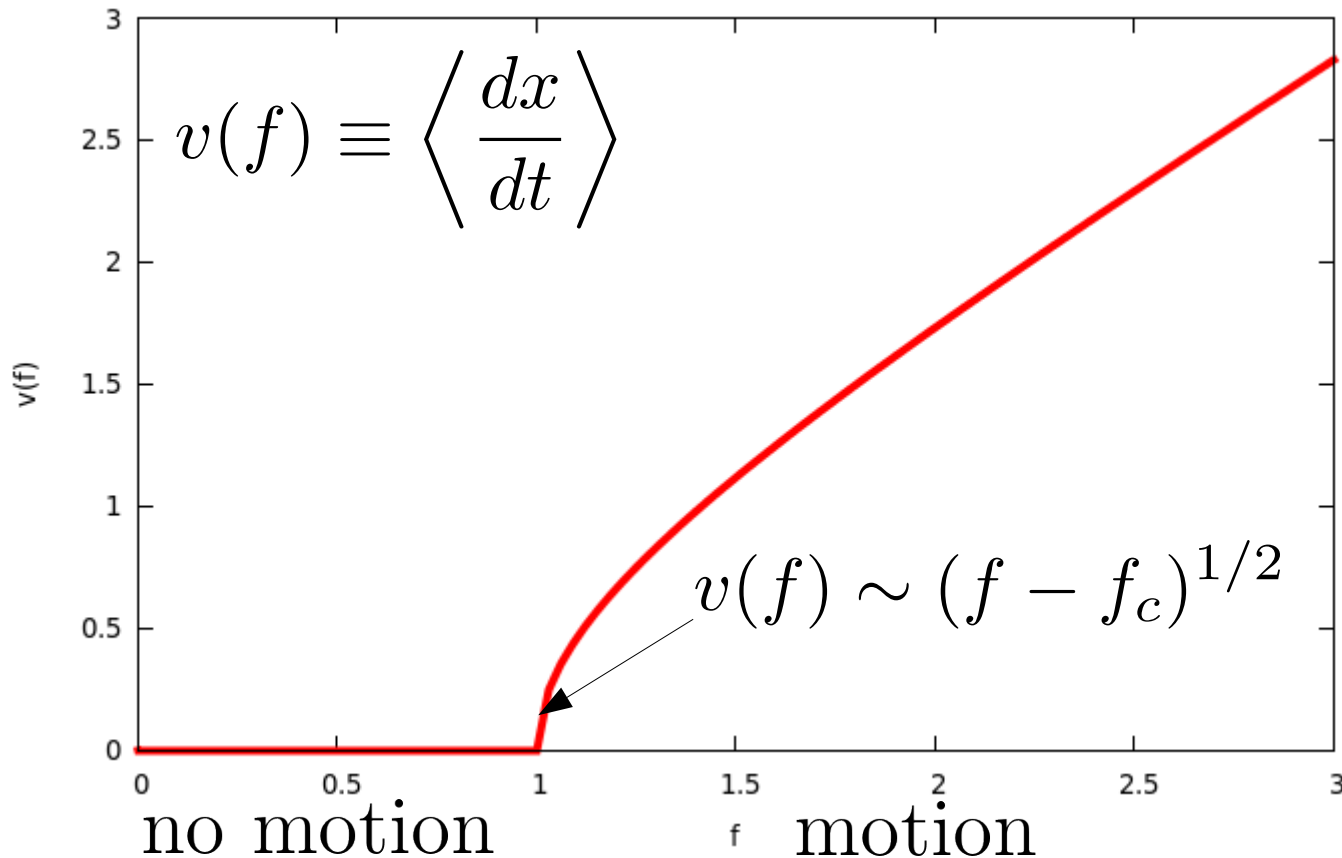
$$\frac{dx}{dt} = -\sin(2\pi x) + f = -\frac{d}{dx}[U(x) - fx]$$



# Depinning

- One particle in a simple periodic potential

$$\frac{dx}{dt} = -\sin(2\pi x) + f$$



# Depinning

- One particle in a simple periodic potential

Equation of motion

$$\frac{dx}{dt} = -\sin(2\pi x) + f$$

Time Period

$$\tau = \int_0^1 \frac{dx}{f - \sin(2\pi x)} = \frac{1}{\sqrt{f^2 - 1}}$$

Velocity-force characteristics

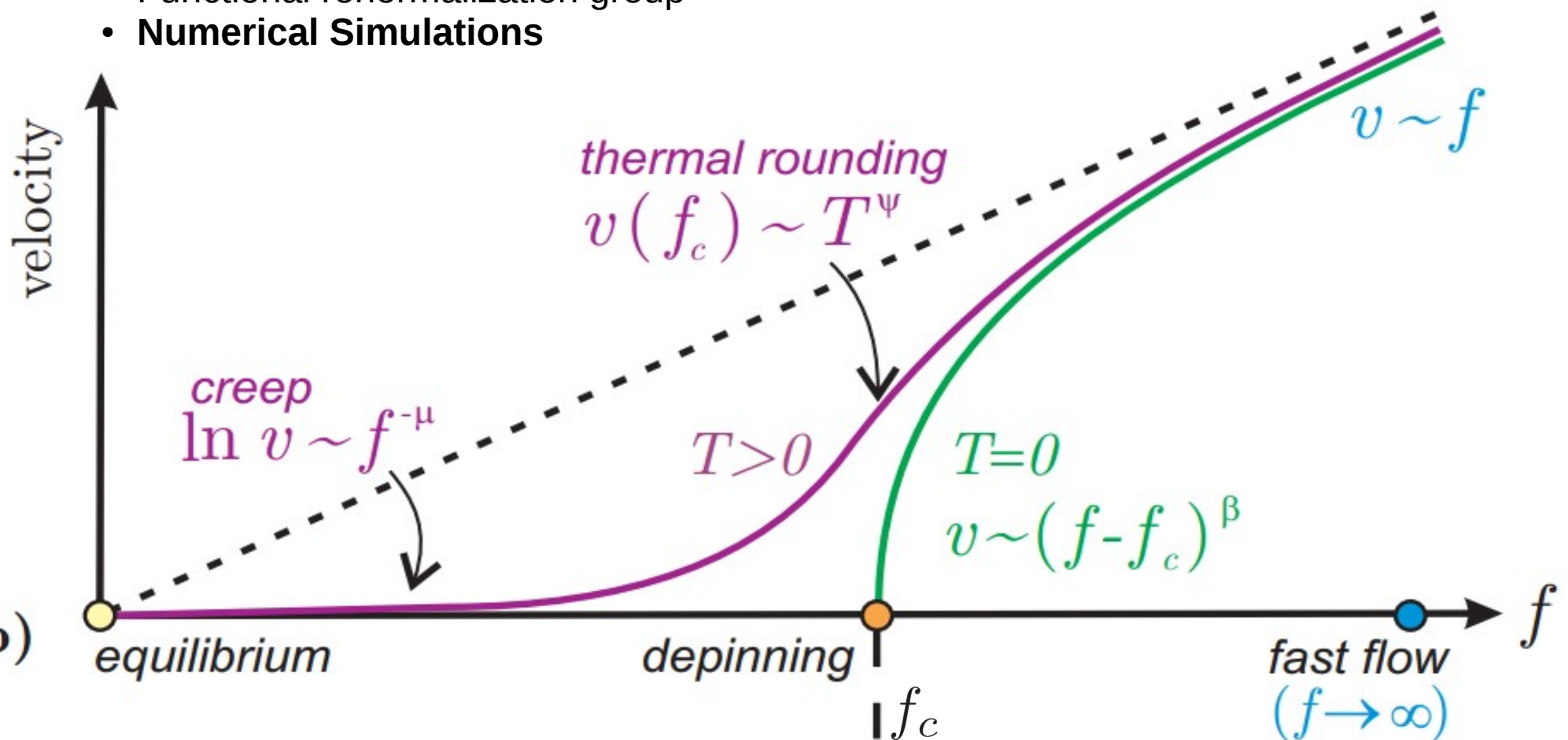
$$v = \left\langle \frac{dx}{dt} \right\rangle = \tau^{-1} = \sqrt{f^2 - 1}$$

“critical” behaviour  
Near the threshold  
 $f \rightarrow f_c^+$

$$\left\{ \begin{array}{l} v \approx 2\sqrt{f - 1} \equiv (f - f_c)^{1/2} \\ \tau \approx (f - f_c)^{-1/2} \end{array} \right.$$

# Predictions from the Elastic String model: Dynamical Regimes

- Scaling Arguments
- Functional renormalization group
- **Numerical Simulations**




Three-reference steady states

# Thermal Effects

- Particle in an arbitrary one dimensional potential

$$\frac{dx}{dt} = -\frac{dU}{dx} + f + \eta(t)$$

Thermal noise



$$\langle \eta(t)\eta(t') \rangle = 2T\delta(t - t')$$

General solution for the mean velocity

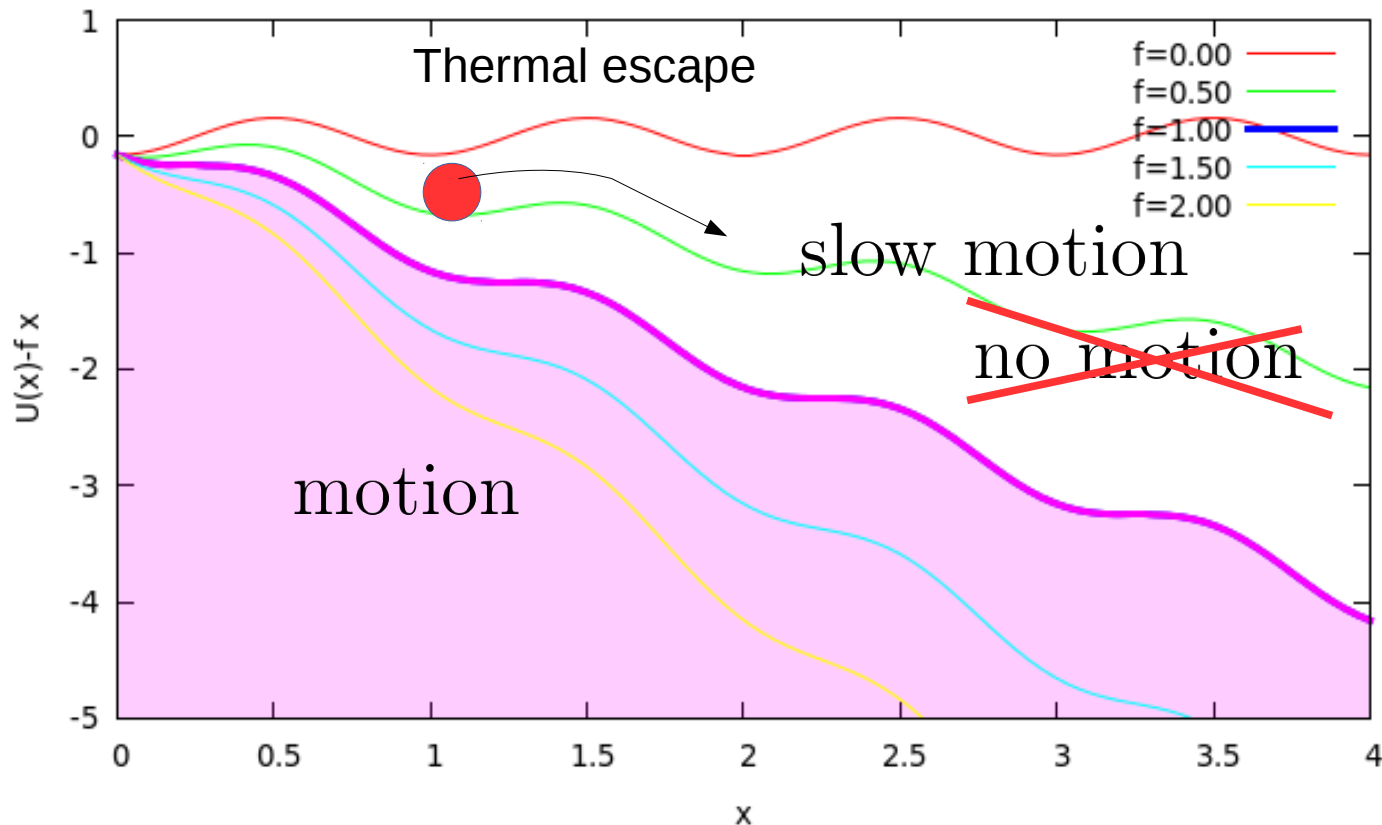
$$v = \frac{T}{\int_0^\infty dz e^{-fz/T} \langle e^{\frac{U(x+z) - U(x)}{T}} \rangle_x}$$

# Thermal Effects

- One particle in a simple periodic potential

$$\frac{dx}{dt} = -\sin(2\pi x) + f + \eta(t)$$

Thermal noise



# Thermal Effects

- One particle in a simple periodic potential

$$U(x) = -\frac{\cos(2\pi x)}{2\pi}$$

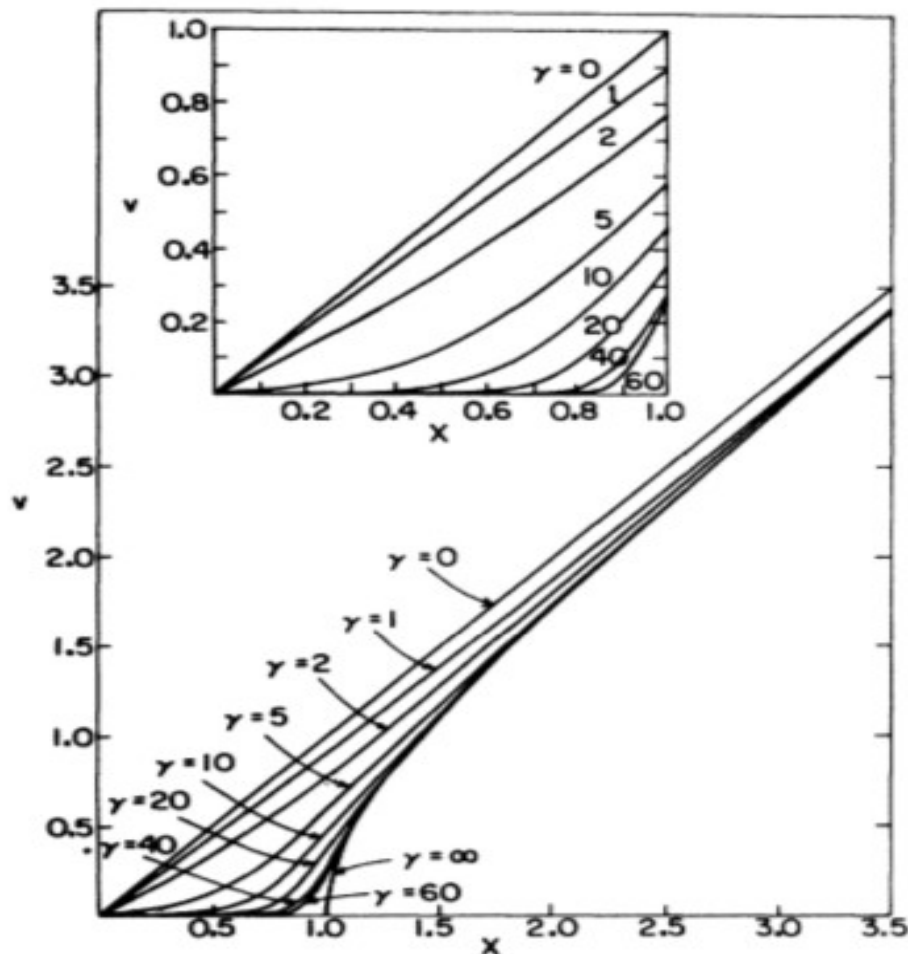
$$\frac{dx}{dt} = -\sin(2\pi x) + f + \eta(t)$$

$$v(f, T) = \frac{2T \sinh(f/2T)}{|I_{if/2T}(1/2\pi T)|^2}$$



# Thermal Effects

- One particle in a simple periodic potential



A. R. Bishop<sup>†</sup> and S. E. Trullinger<sup>‡</sup>

$$\frac{dx}{dt} = -\sin(2\pi x) + f + \eta(t)$$

$$f_c = 1$$

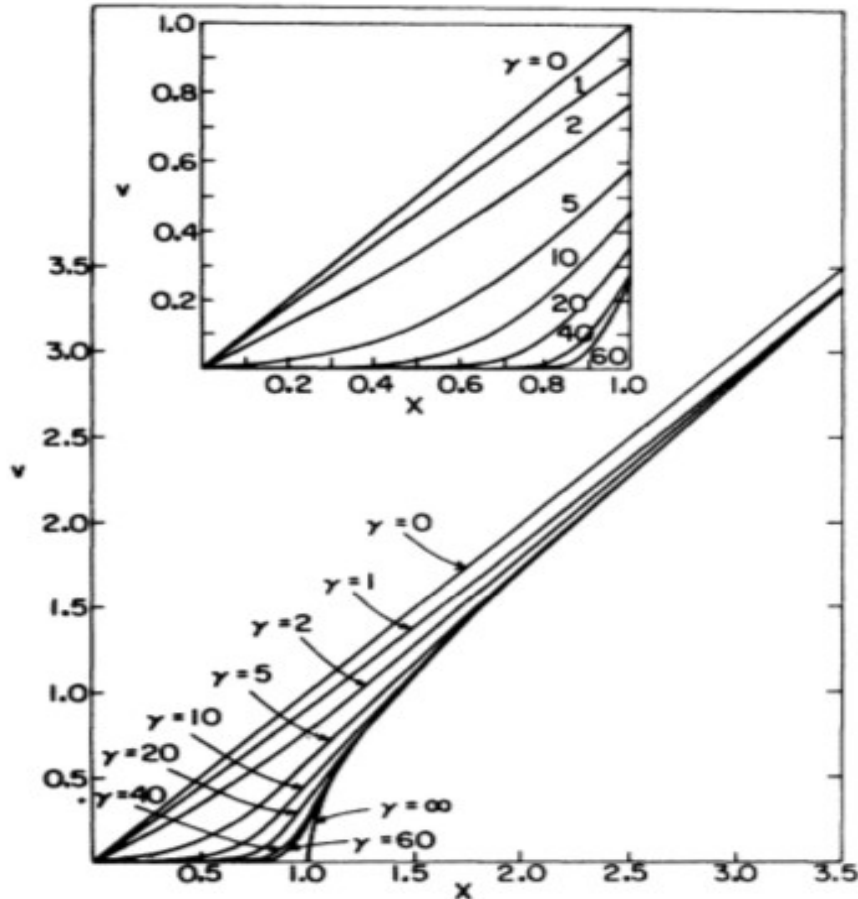
$$v(f \gtrsim f_c, T = 0) \sim (f - f_c)^{1/2}$$

$$v(f = f_c) \sim T^{1/3}$$

$$v(f \ll f_c, T) \sim f e^{-U_c/T}$$

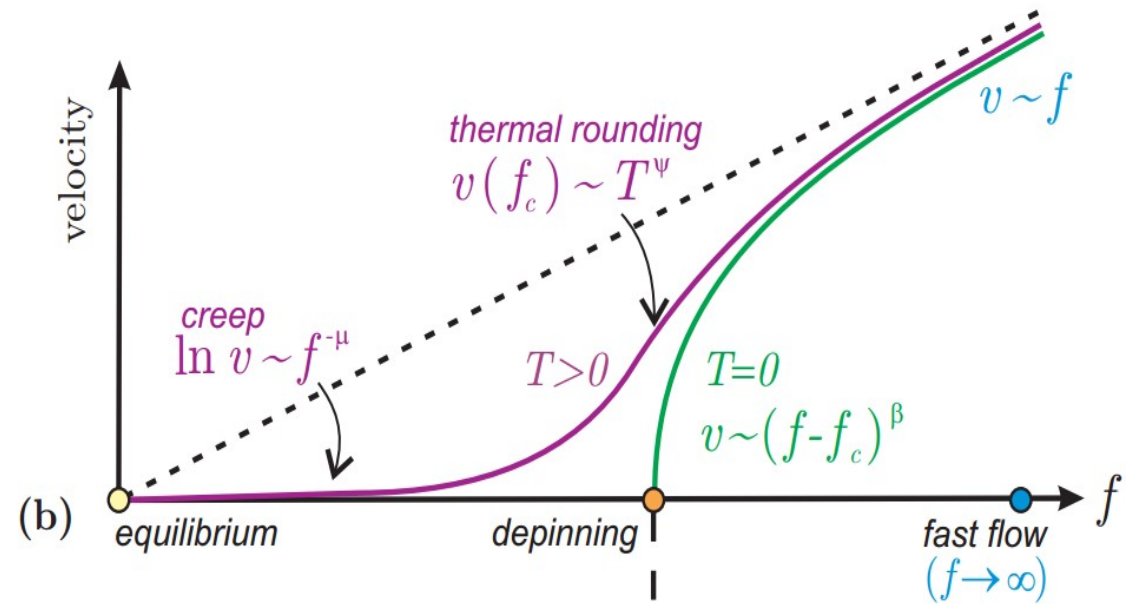
# Similar dynamical Regimes

Particle



A. R. Bishop<sup>†</sup> and S. E. Trullinger<sup>‡</sup>

Elastic String



Does this mean that the Elastic Interface Model phenomenology can be understood with a particle model?

# Particle vs String in random media

- One particle in a simple periodic or bounded random potential

$$v(f = f_c) \sim T^{1/3}$$

$$v(f \gtrsim f_c, T = 0) \sim (f - f_c)^{1/2}$$

$$v(f \ll f_c, T) \sim f e^{-U_c/T}$$

$$\psi = 0.33, \beta = 0.5, \mu = 0$$

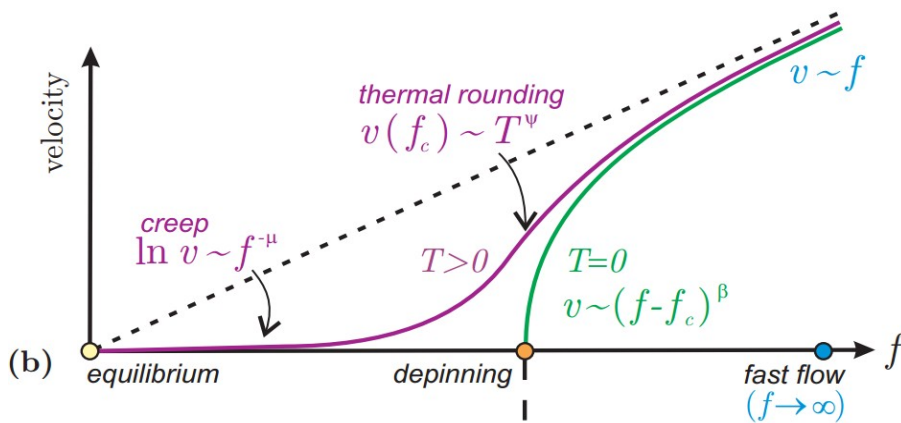
- Elastic String in a Random potential

$$v(f = f_c) \sim T^\psi$$

$$v(f \gtrsim f_c, T = 0) \sim (f - f_c)^\beta$$

$$v(f \ll f_c, T) \sim e^{-U_c(f/f_c)} T^\mu$$

$$\psi = 0.15, \beta = 0.245, \mu = 1/4$$



Qualitatively similar, but not quantitatively.  
What is producing the difference?



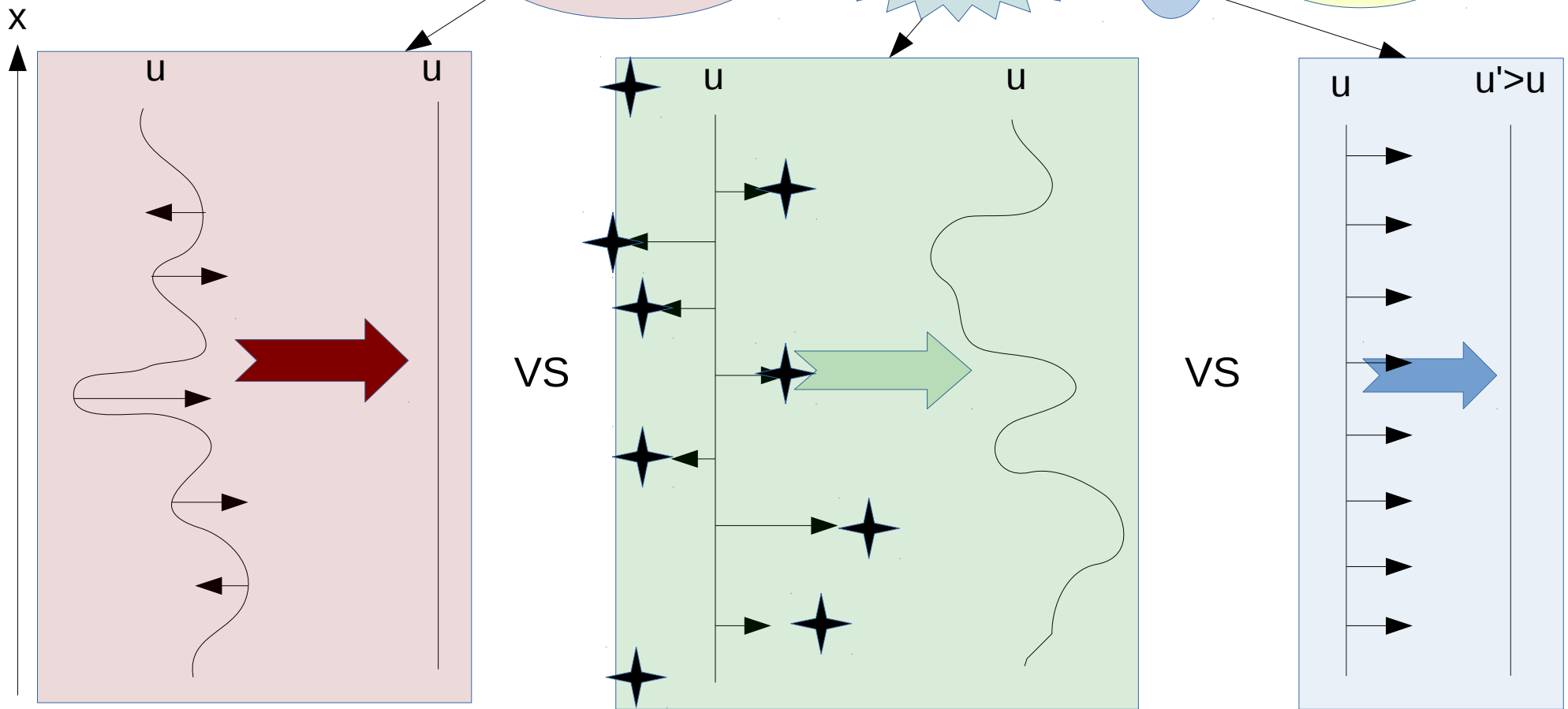
**COLLECTIVE DYNAMICS**

# A minimal non trivial model

... to capture the competition between disorder and elasticity  
and to predict the response to a driving field

Temperature

$$\gamma \partial_t u(x, t) = c \partial_x^2 u(x, t) + F_p(u, x) + f + \eta(x, t)$$



*smoothes*

*distorts and pins*

*pushes*

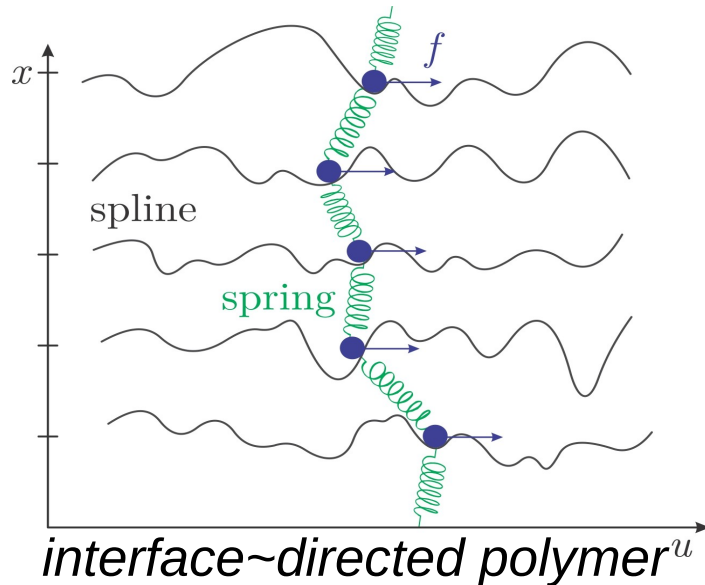
[Note: far more simple than a micro-magnetic model... too simple?]

# Collective dynamics



# Numerical Implementation

$$\gamma \partial_t u(x, t) = c \partial_x^2 u(x, t) + F_p(u, x) + f + \eta(x, t)$$



$$x = X \Delta x, \quad t = n \Delta t, \quad u \text{ continua}$$

$$F_p(u, x) \rightarrow F_p(u, X) = \text{random splines}$$

$$\eta(x, t) \rightarrow \eta(X, n) = \text{random numbers}$$

simplest: finite differences, explicit Euler method

$$F_{tot}(X, n) = C[u(X-1, n) + u(X+1, n) - 2u(X, n)] \\ + F_p[u(X, n), X] + f + \sqrt{\frac{2T}{\Delta t}} R(X, n)$$

$$u(X, n+1) = u(X, n) + F_{tot}(X, n) \Delta t$$

# Numerical Implementation

$$n = 0, 1, \dots, \infty$$

$$X = 0, 1, \dots, L - 1$$

$$F_{tot}(X, n) = C[u(X - 1, n) + u(X + 1, n) - 2u(X, n)] \\ + F_p[u(X, n), X] + f + \sqrt{\frac{2T}{\Delta t}} R(X, n)$$

$$X = 0, 1, \dots, L - 1$$

$$u(X, n + 1) = u(X, n) + F_{tot}(X, n)\Delta t$$

**Numerical challenge:**

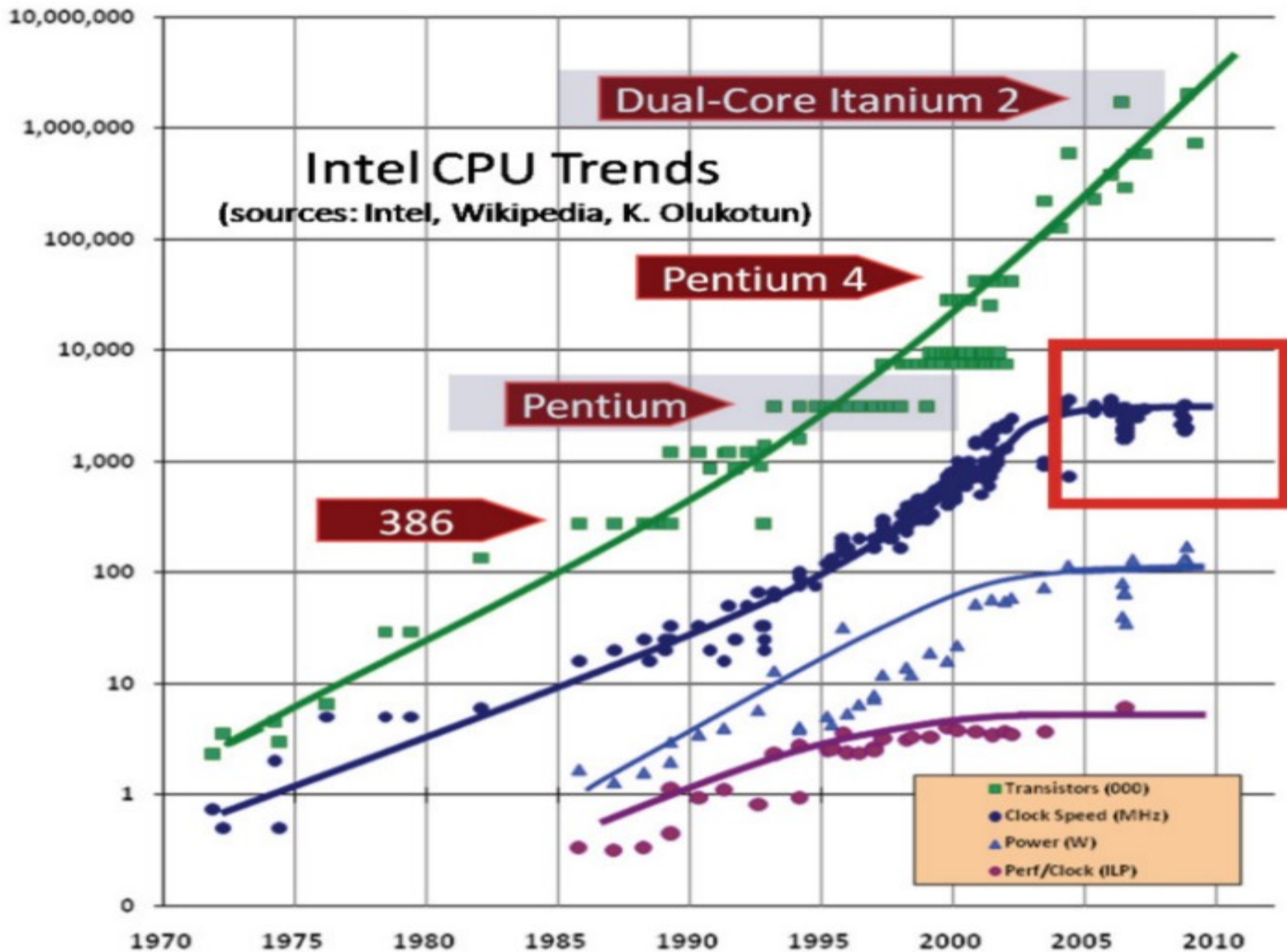
**to get accurate results we need to simulate large system sizes**

**L ~ millions** {

- Large iteration times
- Large memory

→ Buy a faster CPU?

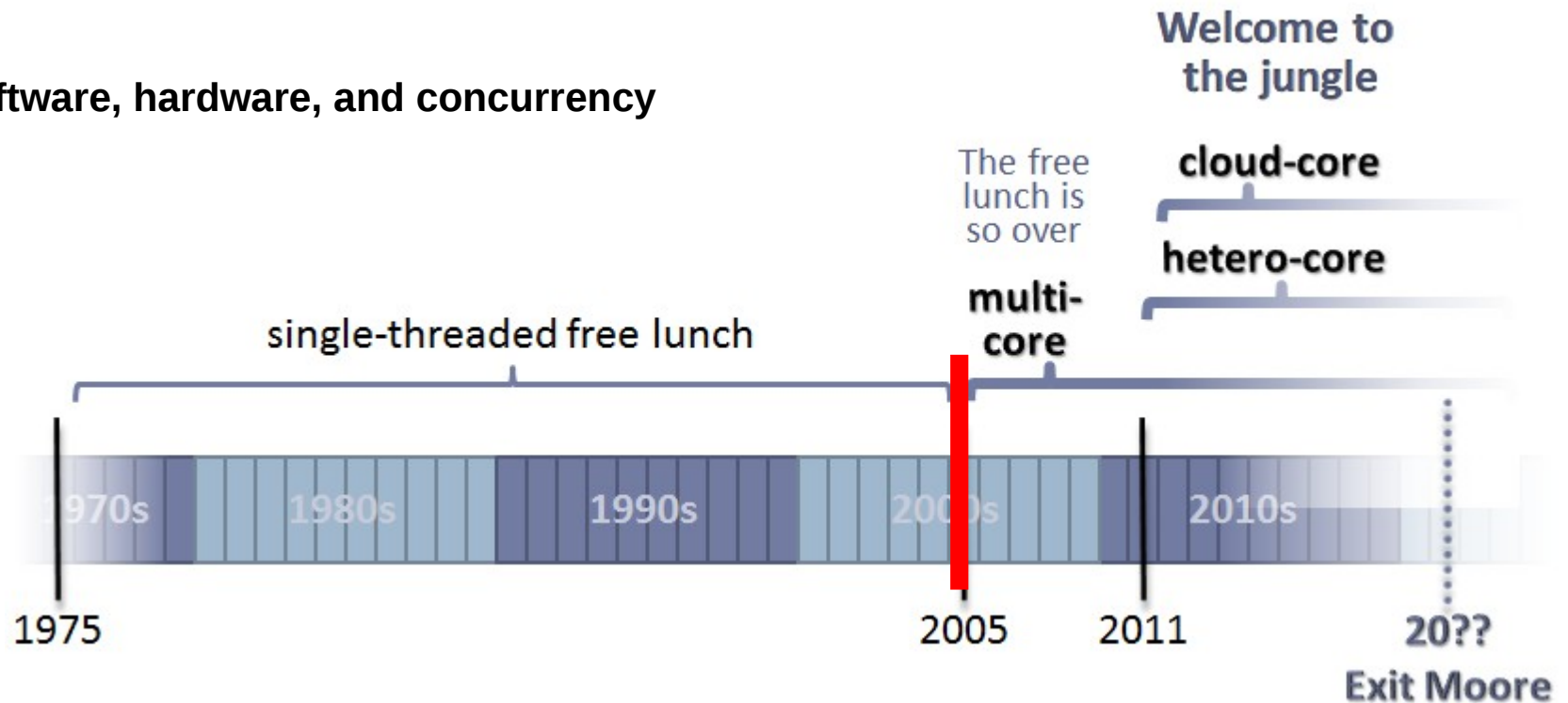
# “Computers no longer get faster, just wider”





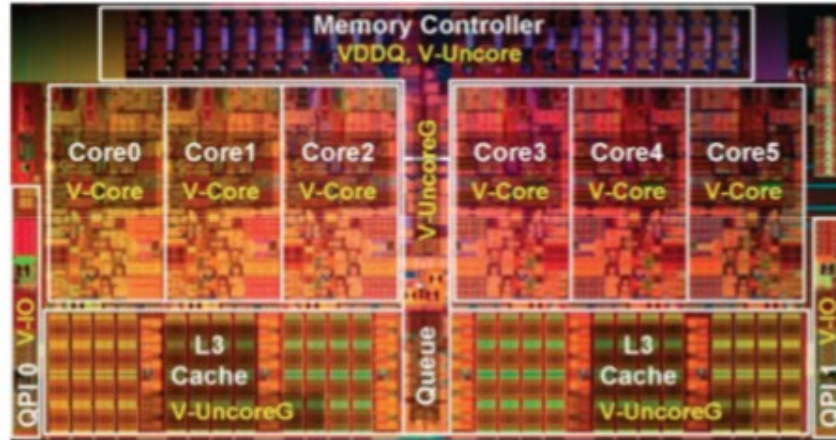
*“For the first time in the history of computing, mainstream hardware is no longer a single-processor von Neumann machine, and never will be again.”*

**Herb Sutter on software, hardware, and concurrency**



- From 1975 to 2005 our industry put a **personal computer** on every desk, in every home, and in every pocket.
- **In 2005, however, mainstream computing hit a wall....** From 2005 to 2011 our industry undertook a new mission: to put a personal parallel supercomputer on every desk, in every home, and in every pocket.
- From 2005 to 2011 we put a **personal *parallel* supercomputer** on every desk, in every home, and in every pocket (multicore tablets, smartphones, etc)

# Many cores in just one machine



## Intel Core i7-980X Extreme

6 cores

1.17B transistors

@3.33GHz, ~100 GFlops

Bandwidth 25.6Gb/s

TDP 130W

~USD 1000

[http://en.wikipedia.org/wiki/Transistor\\_count](http://en.wikipedia.org/wiki/Transistor_count)

## NVIDIA GTX 580 SC

512 cores

3B transistors

@1.54GHz, 1581 GFlops

Bandwidth 192.4Gb/s

TDP 244W

~USD 500

“green HPC”

Hybrid programming

“Task Paralellism”

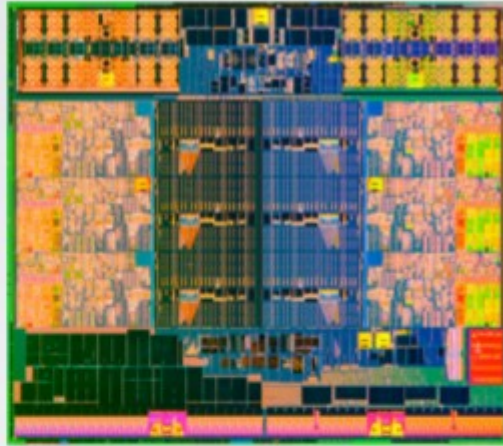
(each thread a different task)

“Data Parallelism”

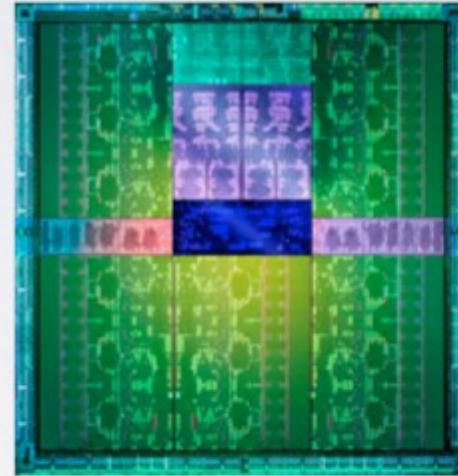
(each thread same task, different data)

# How to use this computing power?

Intel i7-4960x  
1 860 M  
6 Cores



@3.9 GHz, ~100 GFlops  
Bandwidth 25.6Gb/s  
TDP 130W  
~USD 1000



NVIDIA Kepler  
7 100 M  
2688 Cores

@1.54GHz, 1.3 TFlops  
Bandwidth 192.4Gb/s  
TDP 244W  
~USD 900

We must rethink our algorithms  
to be parallel ...

# Parallel Computing

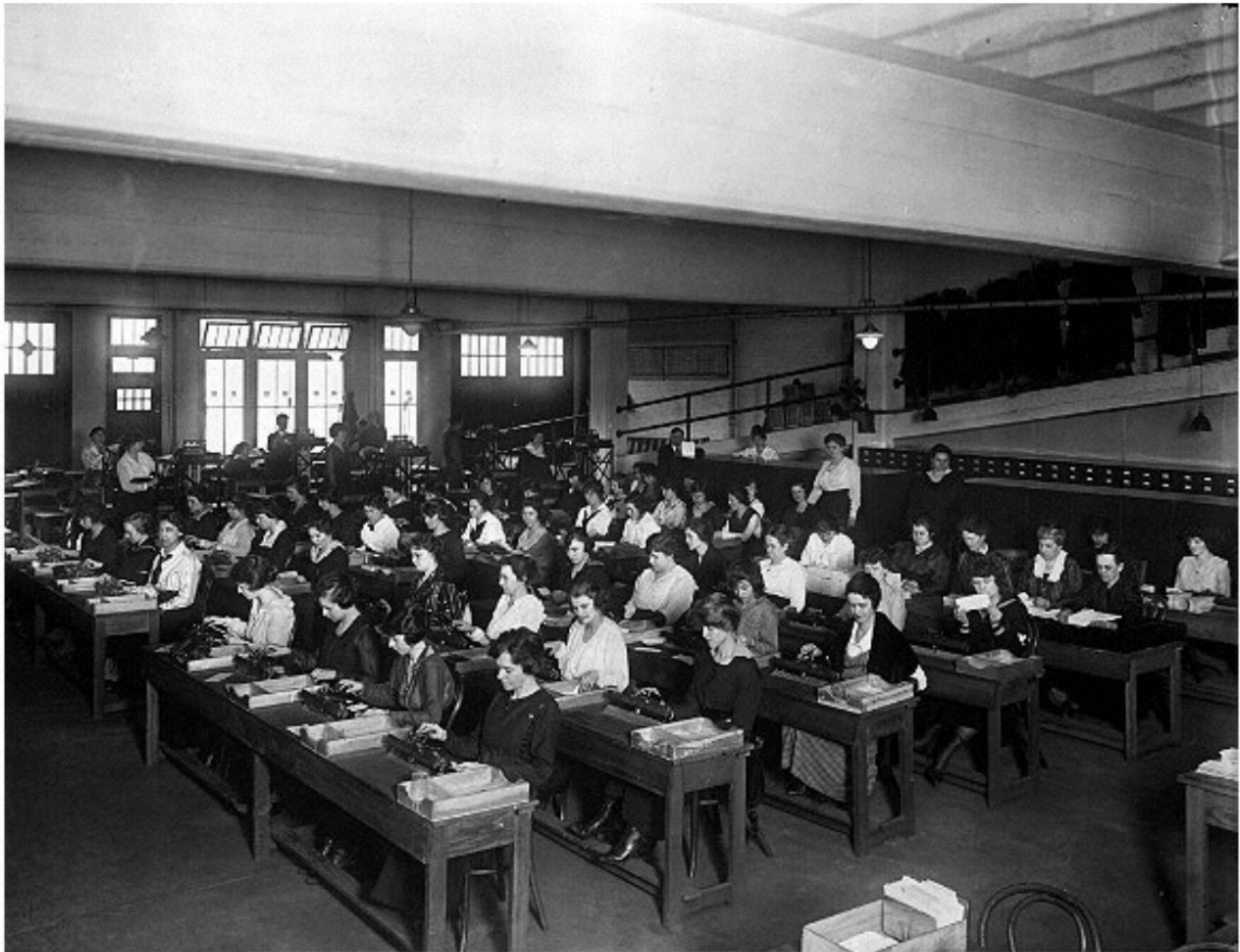
## One common definition

A parallel computer is a **collection of processing elements** that cooperate to solve problems **quickly**

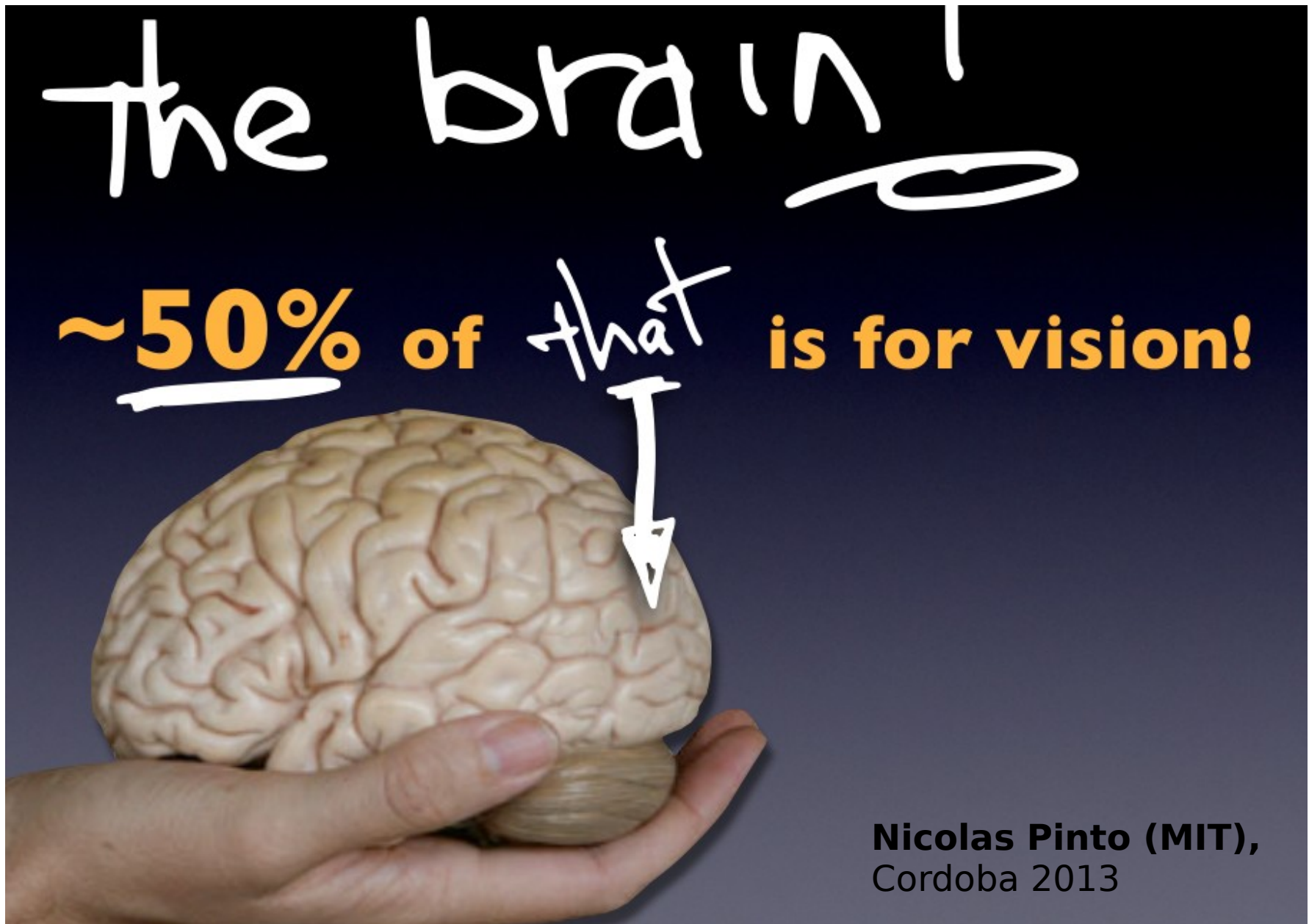
**We care about performance \***  
**We care about efficiency**

**We're going to use multiple  
processors to get it**

# Parallel Computing, socially...



# Parallel Computing, biologically...



The brain

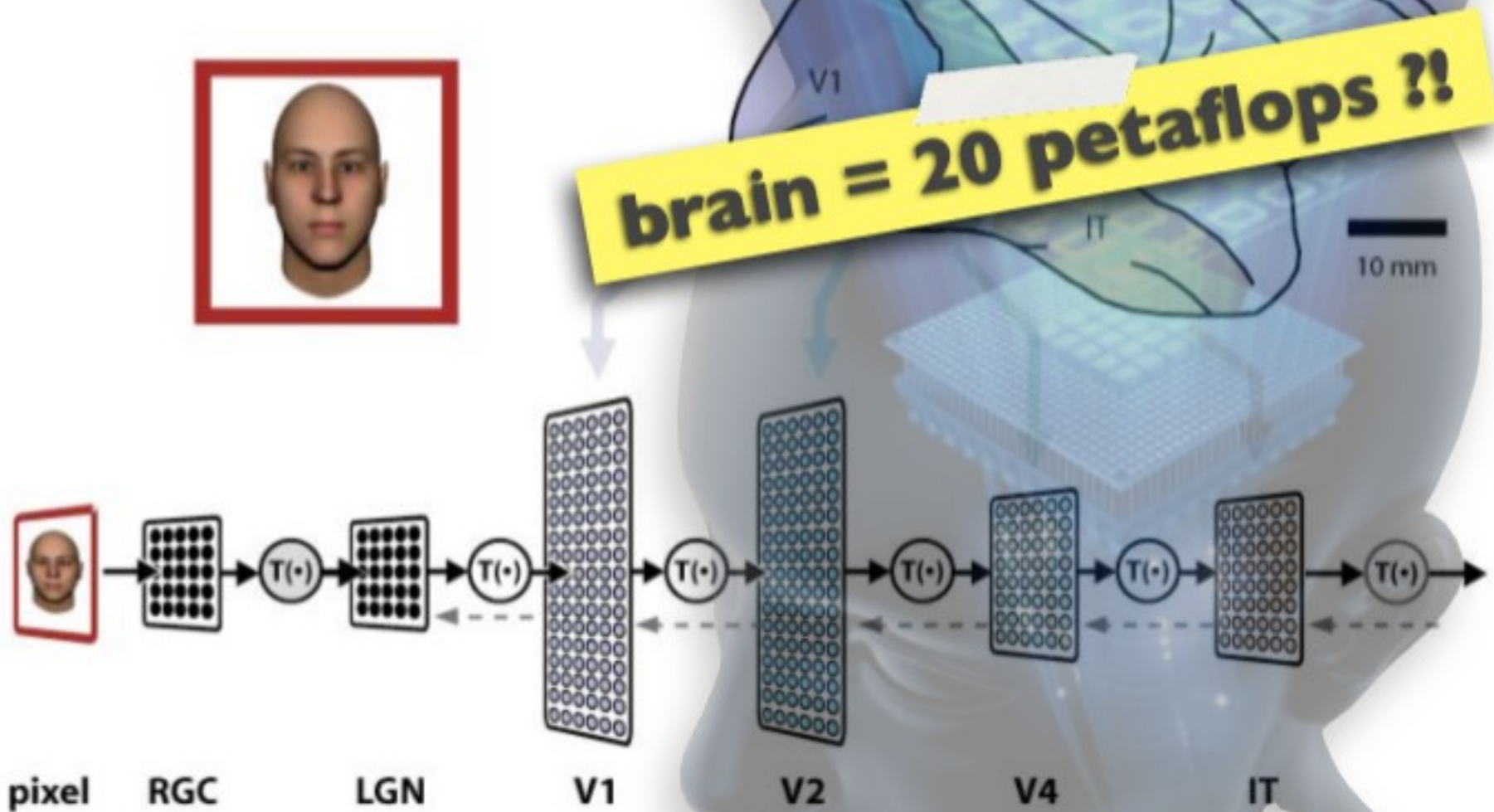
~50% of that is for vision!

Nicolas Pinto (MIT),  
Cordoba 2013

The image features a hand holding a human brain against a dark background. Handwritten white text at the top reads 'The brain'. Below it, '~50%' is underlined in white, followed by 'of that' in white, and 'is for vision!' in orange. A white arrow points from the underlined '50%' down to the brain. In the bottom right corner, the text 'Nicolas Pinto (MIT), Cordoba 2013' is displayed in white.

# Reverse Engineering

The Ventral Visual Stream





# Parallel processing performance vs power



PRESENTED BY  
UNIVERSITY OF  
MANNHEIM

ICL  
INNOVATIVE  
COMPUTING LABORATORY  
UNIVERSITY OF TENNESSEE

BERKELEY LAB  
Lawrence Berkeley  
National Laboratory

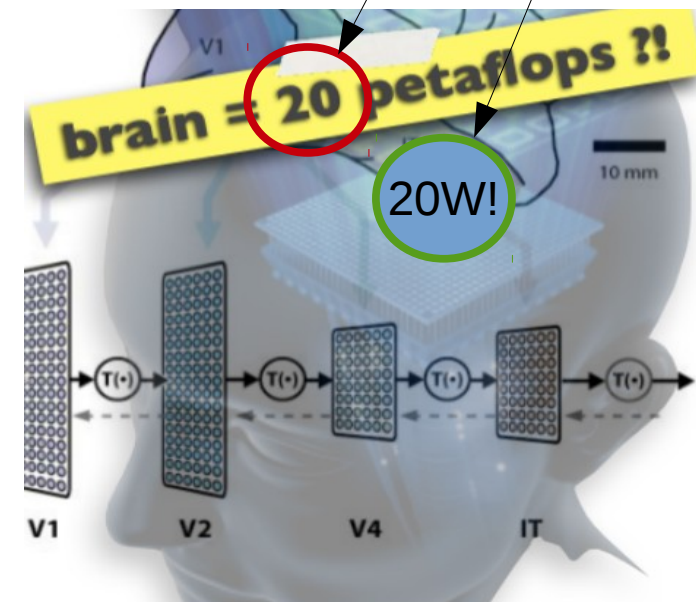
FIND OUT MORE AT  
[www.top500.org](http://www.top500.org)

	NAME	SPECS	SITE	COUNTRY	CORES	R <sub>MAX</sub> P <sub>FLOP/S</sub>	POWER <sub>MW</sub>
1	<b>Tianhe-2 (Milkyway-2)</b>	NUDT, Intel Ivy Bridge (12C, 2.2 GHz) & Xeon Phi (57C, 1.1 GHz), Custom interconnect	NUDT	China	3,120,000	<b>33.9</b>	<b>17.8</b>
2	<b>Titan</b>	Cray XK7, Opteron 6274 (16C, 2.2 GHz) + Nvidia Kepler (14C, .732 GHz), Custom interconnect	DOE/SC/ORNL	USA	560,640	17.6	8.3
3	<b>Sequoia</b>	IBM BlueGene/Q, Power BQC (16C, 1.60 GHz), Custom interconnect	DOE/NNSA/LLNL	USA	1,572,864	17.2	7.9
4	<b>K computer</b>	Fujitsu SPARC64 VIIIfx (8C, 2.0GHz), Custom interconnect	RIKEN AICS	Japan	705,024	10.5	12.7
5	<b>Mira</b>	IBM BlueGene/Q, Power BQC (16C, 1.60 GHz), Custom interconnect	DOE/SC/ANL	USA	786,432	8.16	3.95

	Petaflops	Watts
<b>Tianhe-2</b>	33.9	<b>17800000</b>
<b>Brain (Visual)</b>	20	<b>20</b>

**Computer performance**

Name	FLOPS
yottaFLOPS	10 <sup>24</sup>
zettaFLOPS	10 <sup>21</sup>
exaFLOPS	10 <sup>18</sup>
petaFLOPS	10 <sup>15</sup>
teraFLOPS	10 <sup>12</sup>
gigaFLOPS	10 <sup>9</sup>
megaFLOPS	10 <sup>6</sup>
kiloFLOPS	10 <sup>3</sup>



GPUs: cheap and massively parallel!



# Computer games

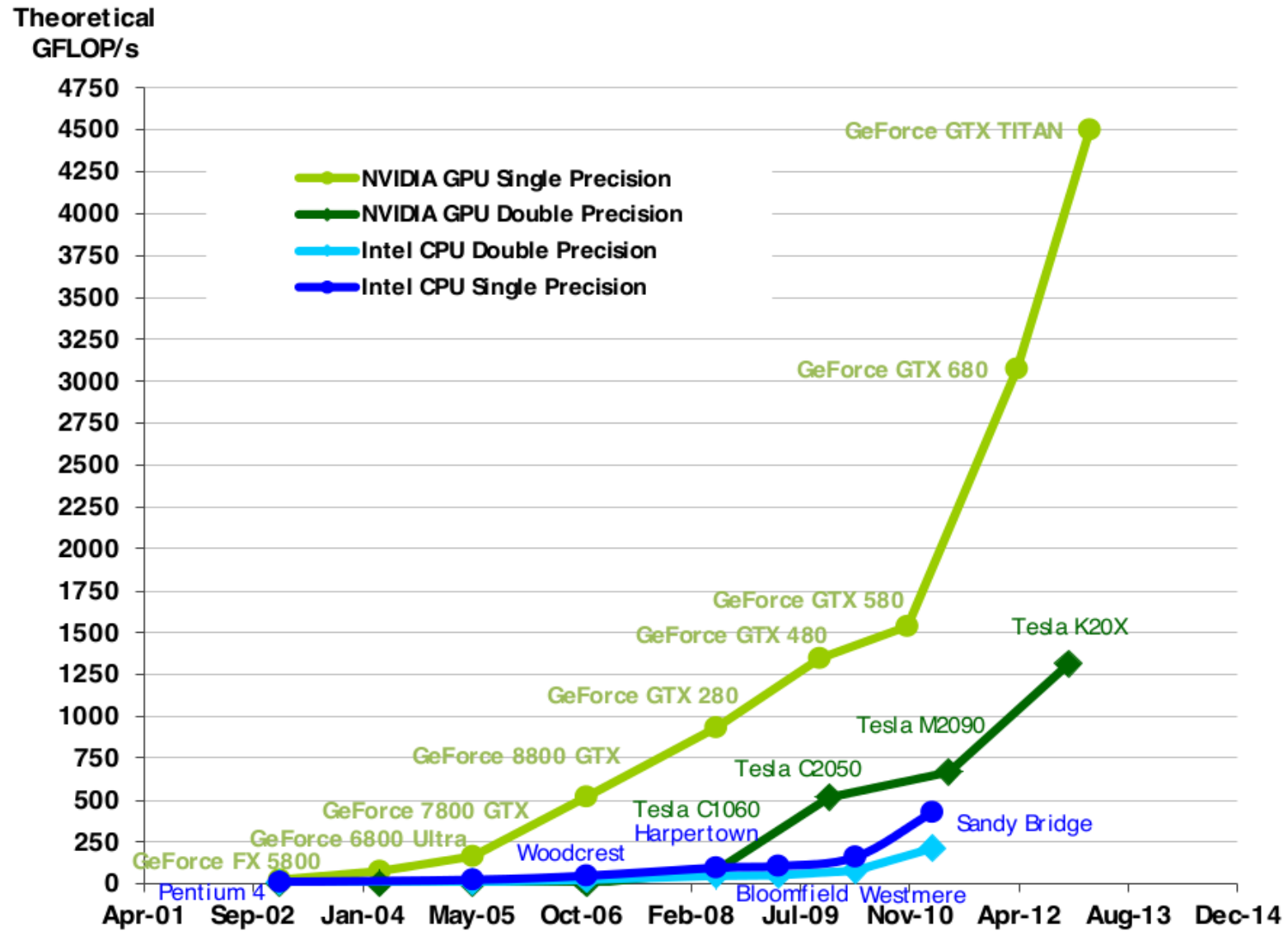


*Elite*  
(1984)

# Parallel computing for accelerating computer games



# GPUs



If you write a scalable parallel program, next year will run faster...

# How do we use this for science?



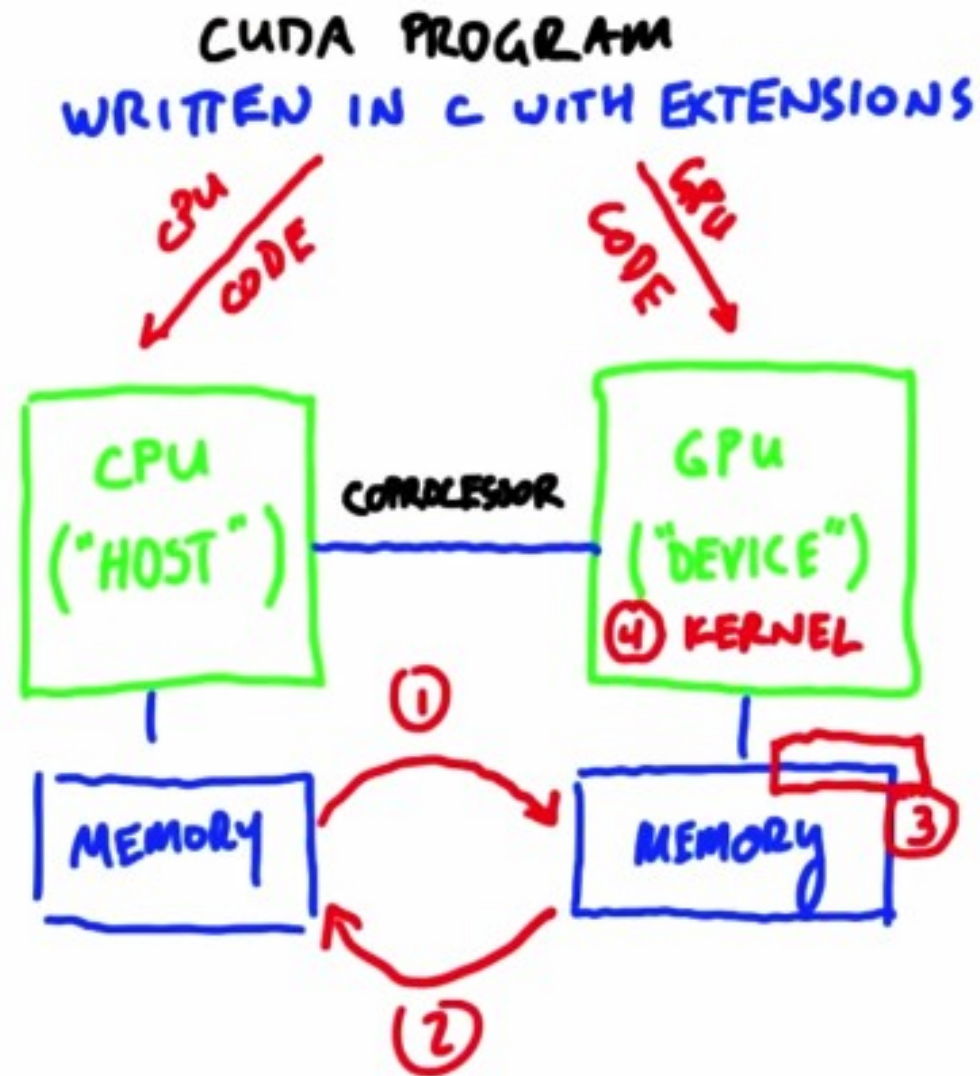
# GPGPU for scientific applications

*General-purpose computing on graphics processing units (GPGPU) is the utilization of a **graphics processing unit** (GPU), which typically handles computation only for **computer graphics**, to perform computation in applications traditionally handled by the **central processing unit** (CPU).*

# CUDA for scientific applications

- Before 2007 it was difficult... (only rendering oriented languages).
- After 2007:

**CUDA**, which stands for Compute Unified Device Architecture, is a parallel computing platform and programming model created by NVIDIA and implemented by the graphics processing units (GPUs) that they produce. CUDA gives developers direct access to the virtual instruction set and memory of the parallel computational elements in CUDA GPUs.





# CUDA for scientific applications

DEFINING THE GPU COMPUTATION

BIG IDEA



KERNELS LOOK LIKE SERIAL PROGRAMS

WRITE YOUR PROGRAM AS IF IT WILL RUN ON ONE THREAD

THE GPU WILL RUN THAT PROGRAM ON MANY THREADS

# CUDA for scientific applications

- **Throughput vs Latency**
  - Slow but many “workers” or threads...
- **Data-parallel computing is the most scalable solution**
  - Instead of modifying the code when we change the number of cores (2, 4, 8, 16...)
  - We build the computation around the data, as we will always have more data than cores → *Many threads doing the same computation on different data.*
- **Scalability !**
  - from cell phones to supercomputers.
  - from current to future GPUs.

# Very lucky problem for GPGPU!

Simple numerical implementation: explicit finite-difference

$$u(X, n + 1) = u(X, n) + F_{tot}(X, n)\Delta t$$

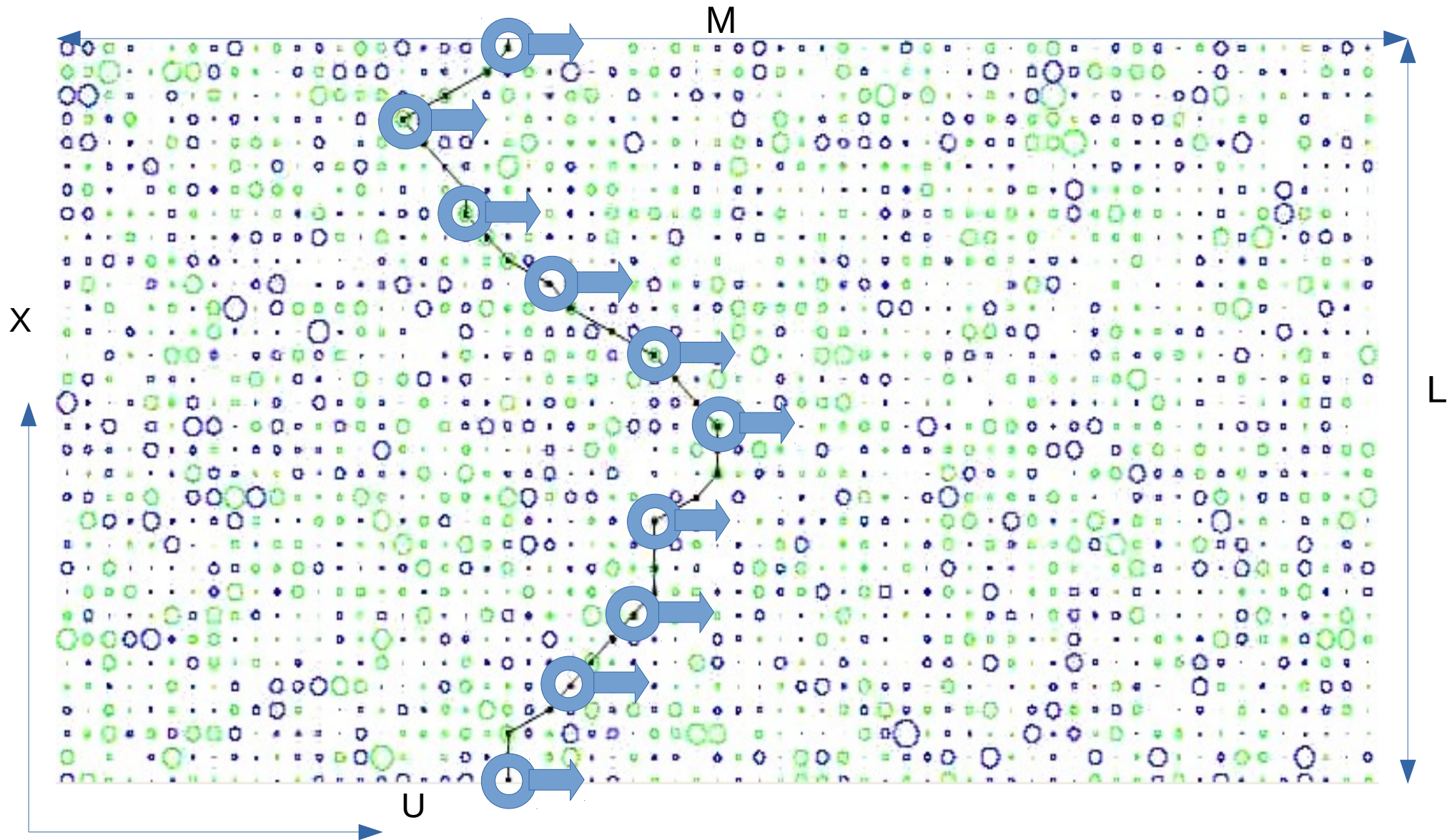
$$F_{tot}(X, n) = C[u(X - 1, n) + u(X + 1, n) - 2u(X, n)] \\ + F_p[u(X, n), X] + f + \sqrt{\frac{2T}{\Delta t}}R(X, n)$$

---

## ***“Embarrassingly parallel problem”***

- 1) ***“Parallel For” or “transform”*** (X=0,...,L-1)  
Thread “X” calculates force in X,  $F_{tot}(X)$  → Array “Ftot” (device memory)
- 2) ***“Parallel For” or “transform”*** (X=0,...,L-1)  
Thread “X” calculates  $u(X, n+1)$  → New “u” array (device memory)
- 3) ***“parallel reductions”*** : physical properties (or **visualization...**)
- 4) Time = Time + Dt ; Go to (1)

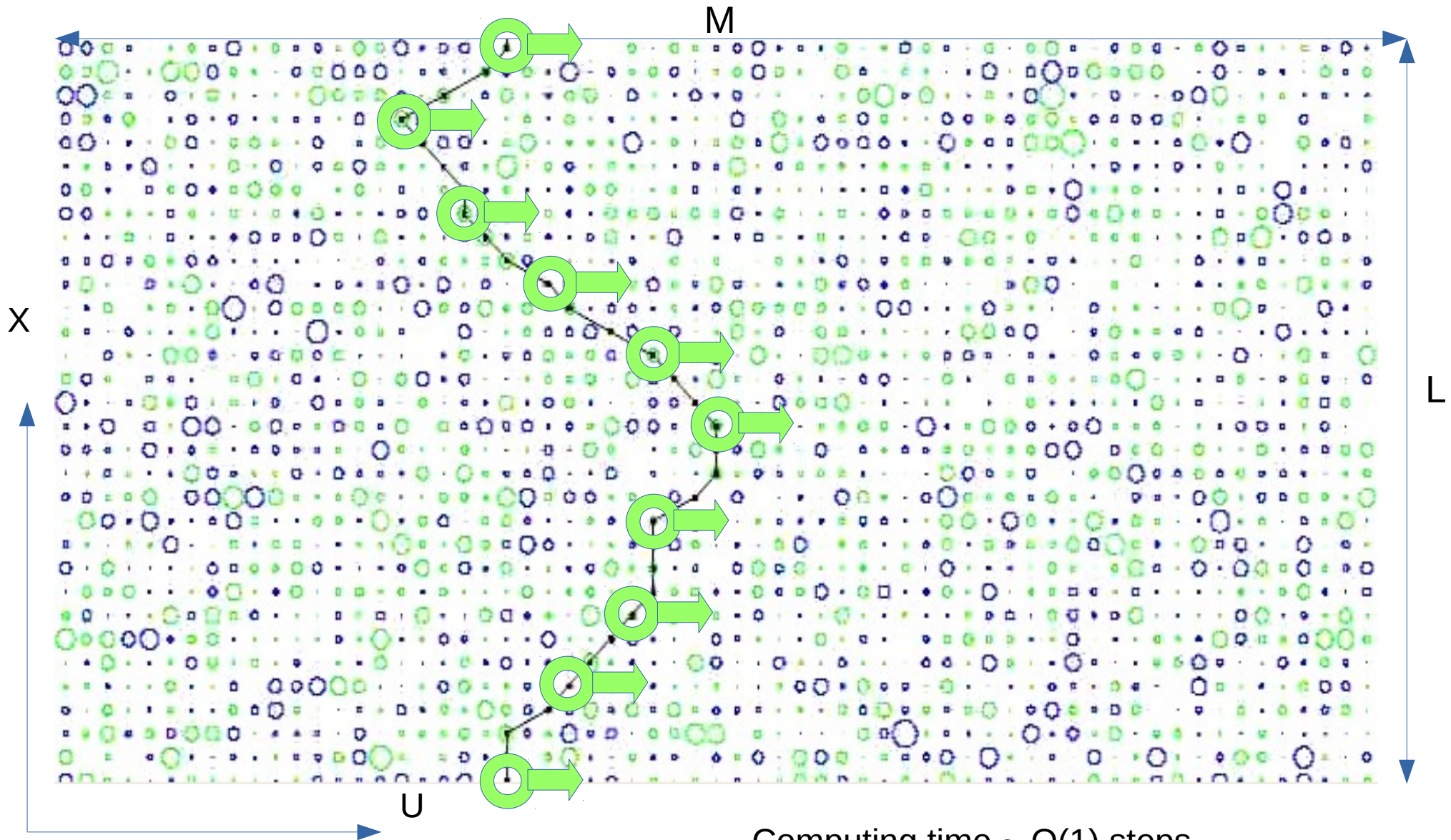
Serial time step:  
*one "thread" in charge of many operations*



Computing time  $\sim O(L)$  steps

# Parallel time step:

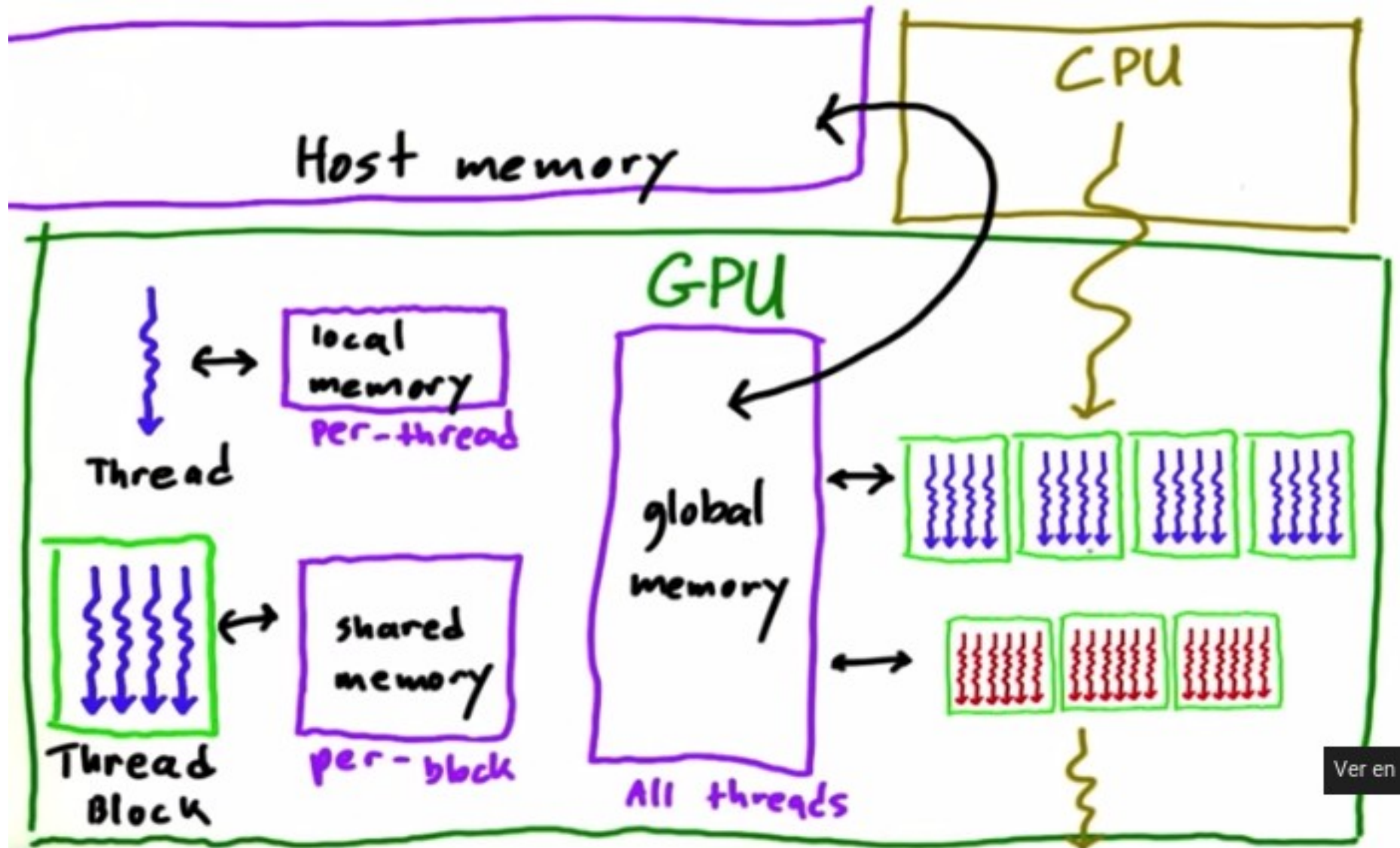
*Many threads in charge of many operations*



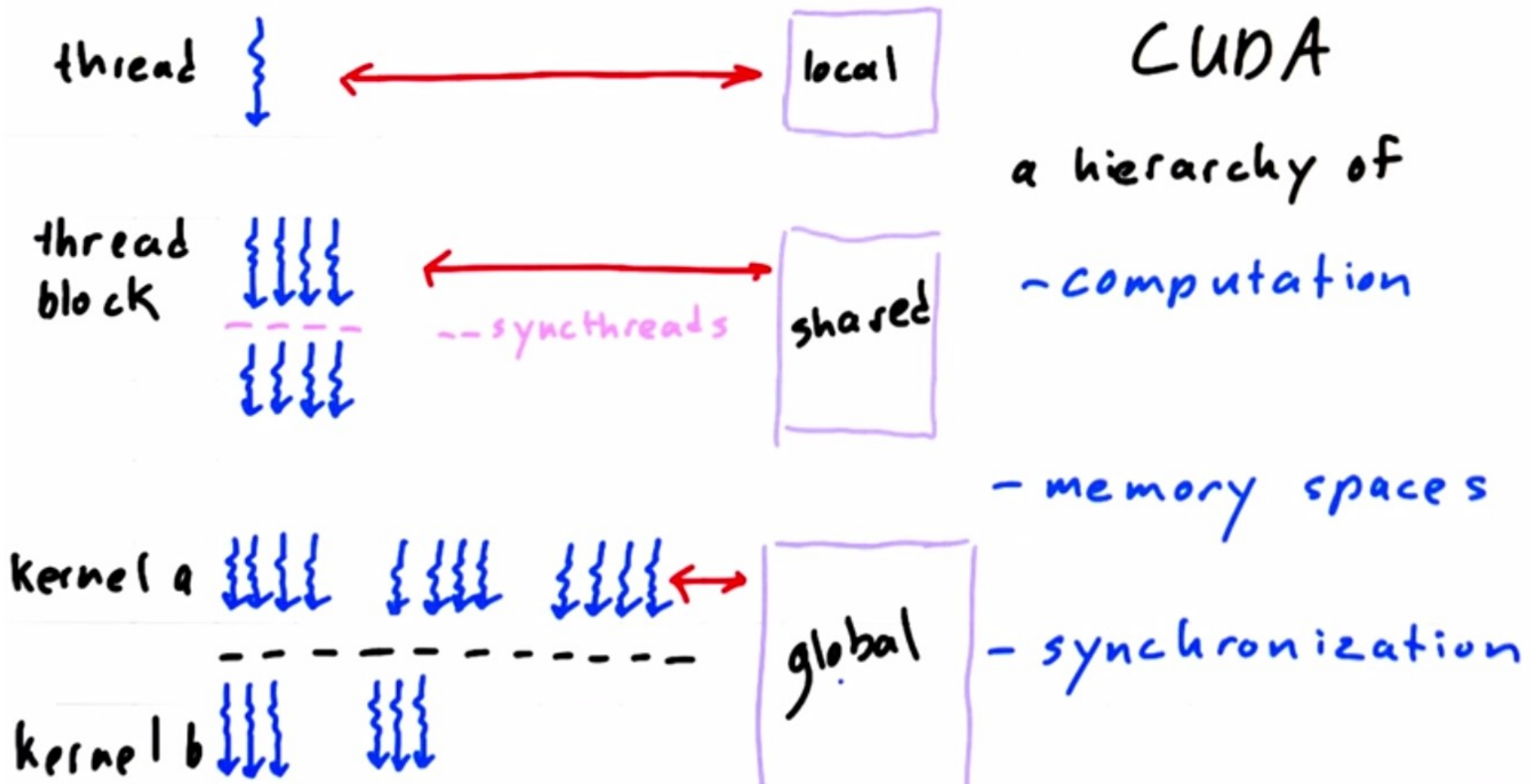
Computing time  $\sim O(1)$  steps  
 $\sim O(L)/N_{\text{eff}}$  (If  $L$  large)

How do we code for many parallel threads in the GPU?

# A low level interface: CUDA C/C++



# Programming in CUDA C/C++





# Be careful with threads and communication patterns!

Example:

- Two threads have to increment by 1 an integer, which is zero initially
- Result must be 2, right?

Thread 1	Thread 2		Integer value
			0
read value		←	0
increase value			0
write back		→	1
	read value	←	1
	increase value		1
	write back	→	2

Thread 1	Thread 2		Integer value
			0
read value		←	0
	read value	←	0
increase value			0
	increase value		0
write back		→	1
	write back	→	1

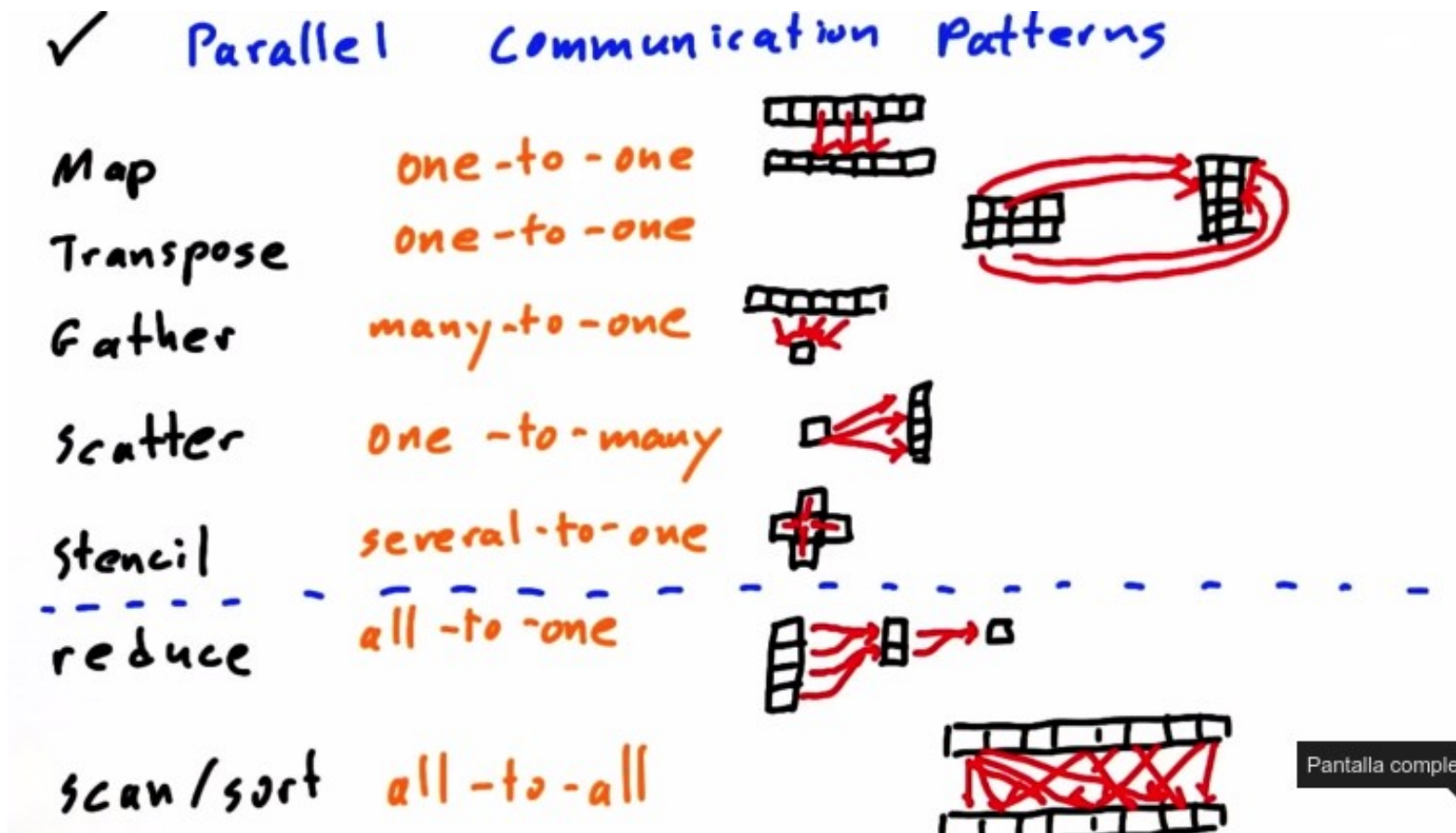
RACE CONDITIONS, and other problems we did not think about !!

How to solve them, without killing too much the performance gain by parallelism?

**Do not be scared!**

there is a way to quickly start programming GPUs,  
and accelerate your old sequential codes :-)

# Identify common parallel patterns in your code....



# Look for a flexible, portable and expressive parallel library

Ex: Thrust library

- Algorithms
  - Searching
    - Binary Search
    - Vectorized Searches
  - Copying
    - Gathering
    - Scattering
  - Reductions
    - Counting
    - Comparisons
    - Extrema
    - Transformed Reductions
    - Logical
    - Predicates
  - Merging
  - Reordering
    - Partitioning
    - Stream Compaction
  - Prefix Sums
    - Segmented Prefix Sums
    - Transformed Prefix Sums
  - Set Operations
  - Sorting
  - Transformations
    - Filling
    - Modifying
    - Replacing

✓ Parallel Communication Patterns

Map

one-to-one



Transpose

one-to-one



Gather

many-to-one



Scatter

one-to-many



Stencil

several-to-one



reduce

all-to-one



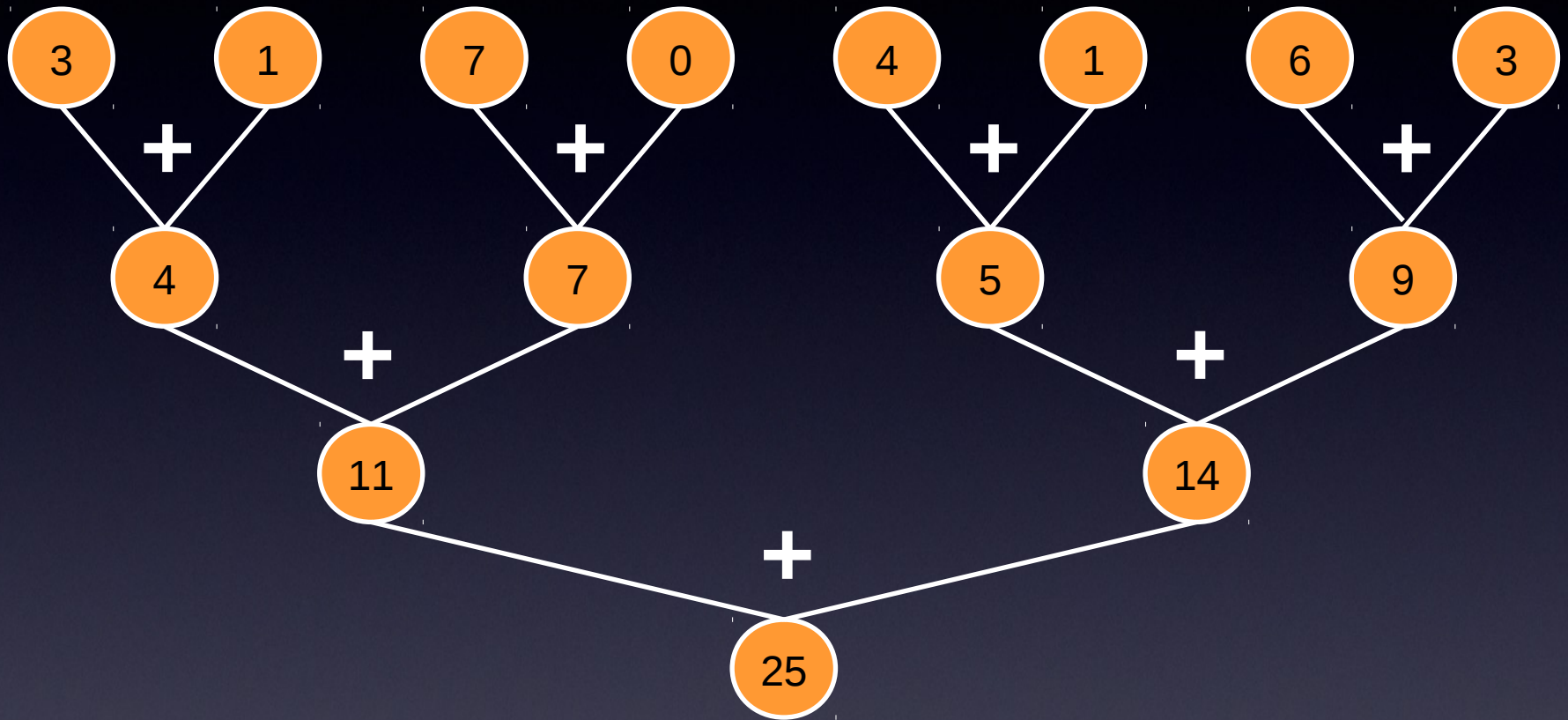
scan/sort

all-to-all

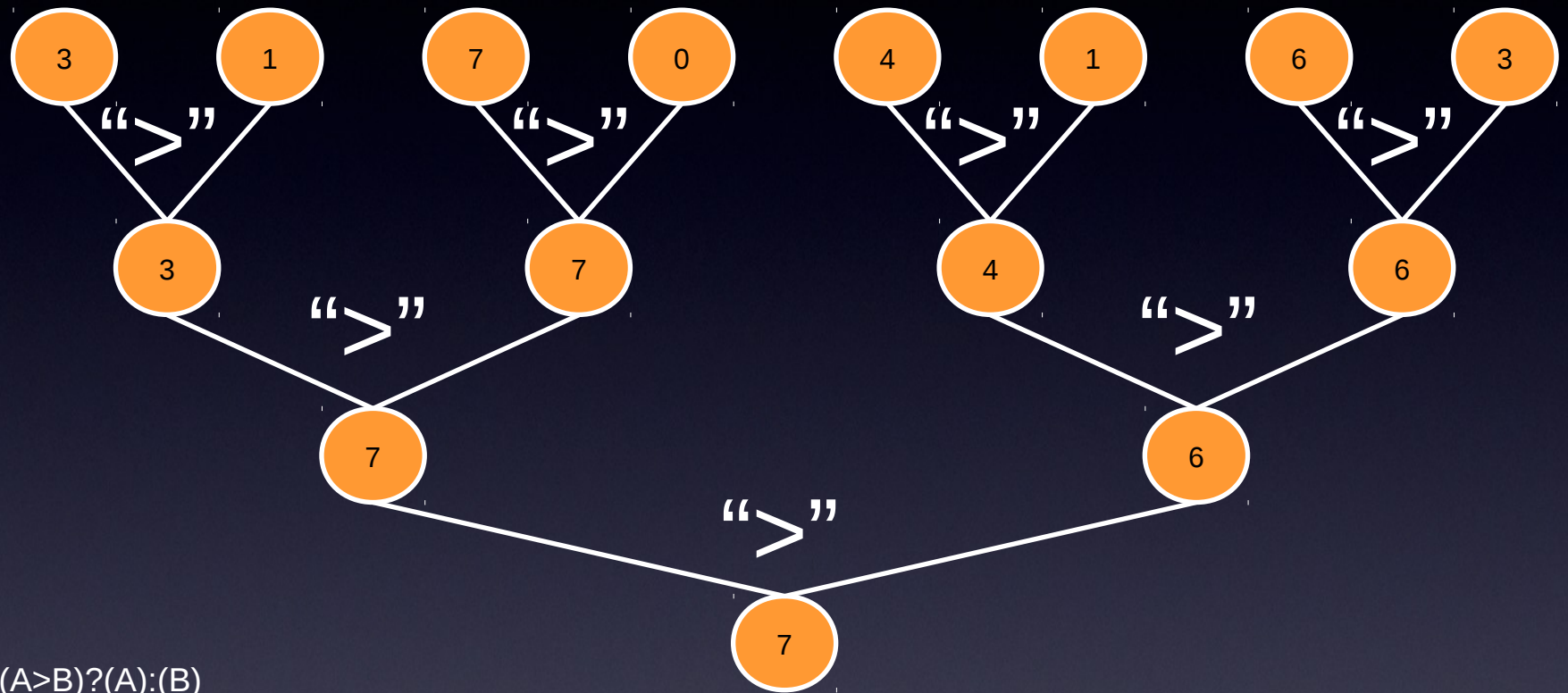


Pantalla completa

Intro to Parallel Programming, Using CUDA to Harness the Power of GPUs  
[UDACITY]

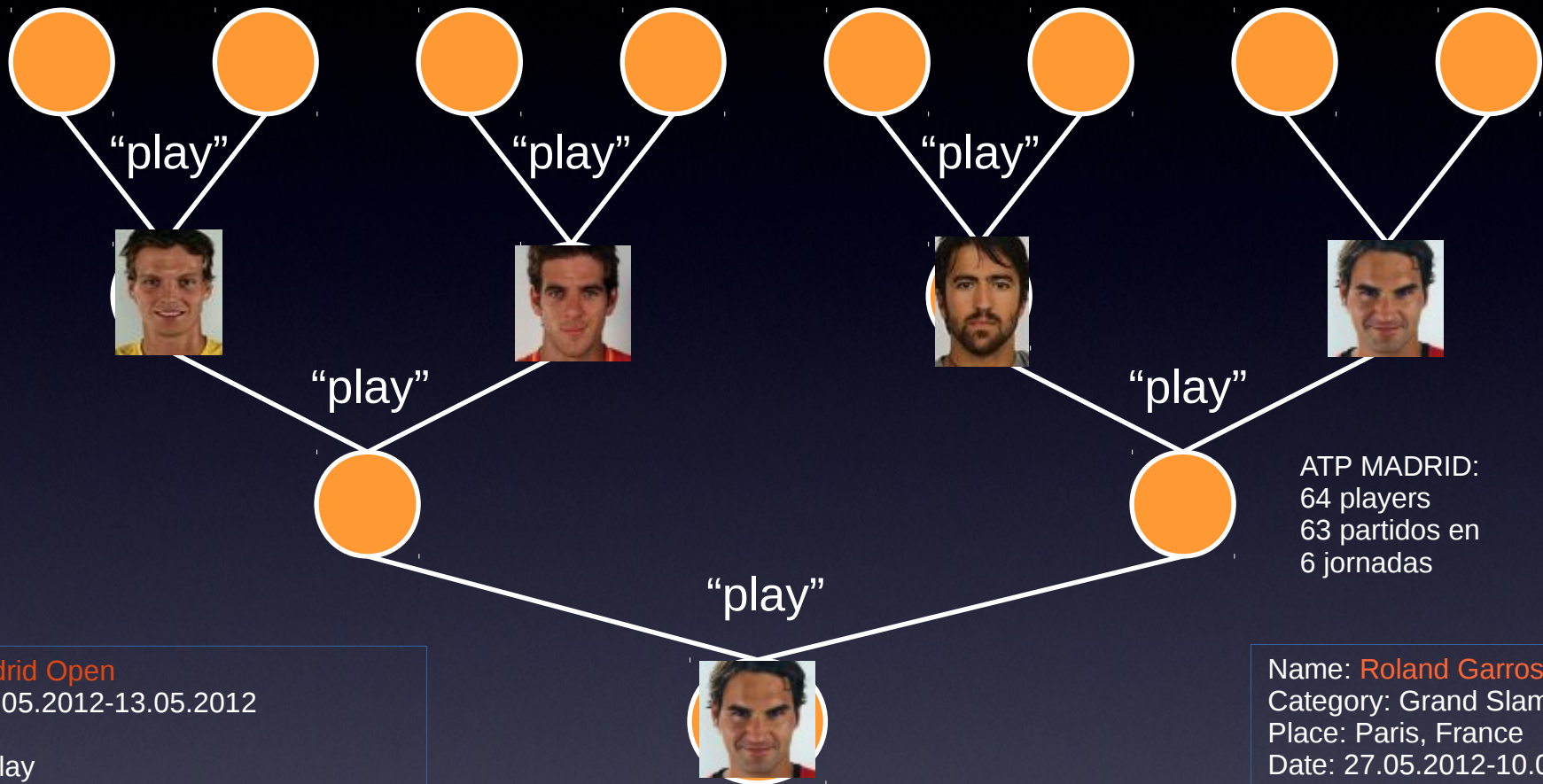


Parallel Reduction  $\sim O(\log_2 N)$  steps  
Serial Reduction  $\sim O(N)$  steps (!)



A ">" B = (A>B)?(A):(B)

Finds the maximum



ATP MADRID:  
 64 players  
 63 partidos en  
 6 jornadas

Mutua Madrid Open  
 Spain - 07.05.2012-13.05.2012  
 Draw: 64  
 Surface: Clay  
 Prize Money: €3,090,150  
 63 partidos en 6 jornadas

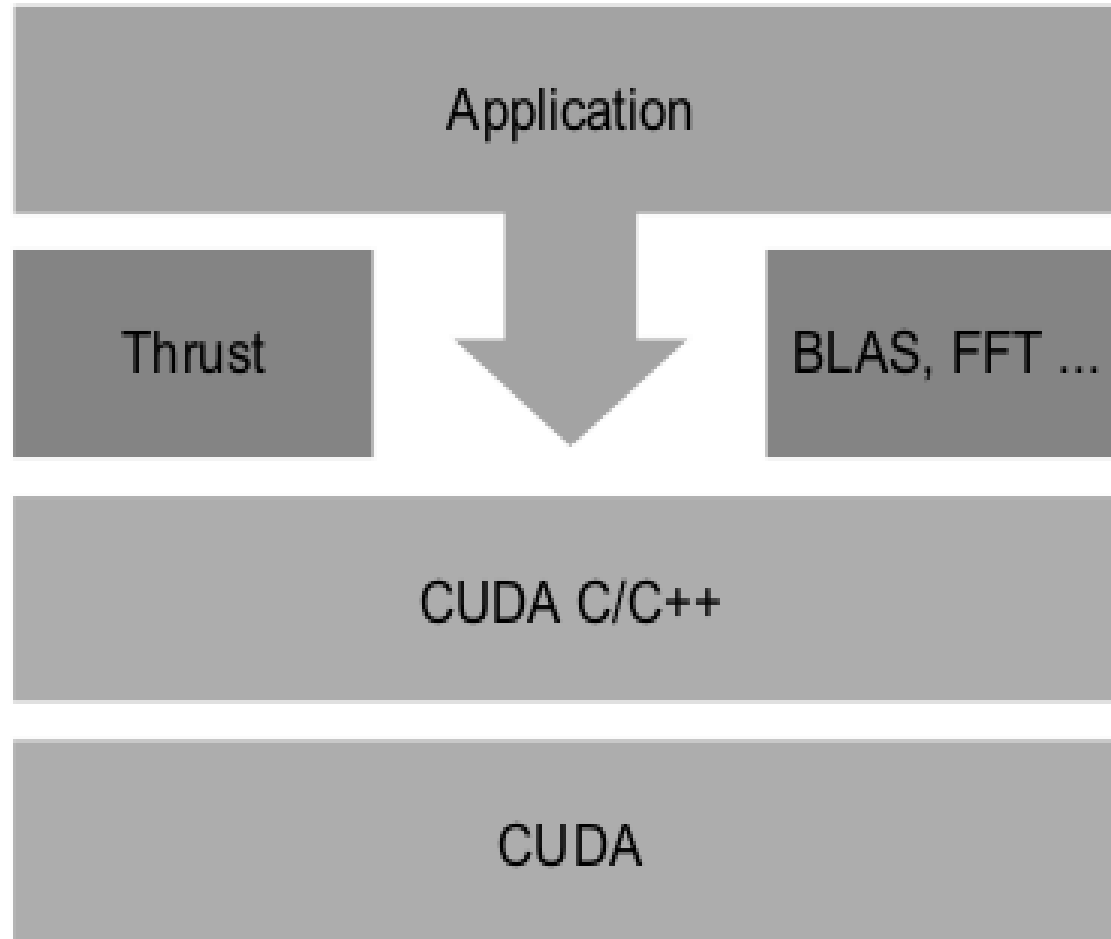
Name: Roland Garros  
 Category: Grand Slam  
 Place: Paris, France  
 Date: 27.05.2012-10.06.2012  
 Draw Size: S-128 D-64

Computes the champion

log<sub>2</sub>(N) operations:

Grand-Grand-Slam: 2<sup>20</sup> = 1048576 players → only 20 stages !!!

# High Level Interface



<http://thrust.github.io/>

# Fundamental Parallel Algorithms Libraries for scientists

## C++ template library



GPU y CPU multicore

## Random Numbers

**Random123:**

a Library of Counter-Based  
Random Number Generators

GPU y CPU multicore

+ several others!

<http://thrust.github.io/>

[https://www.deshawresearch.com/resources\\_random123.html](https://www.deshawresearch.com/resources_random123.html)



# Thrust

Thrust Content from GTC 2012: Nathan Bell

GPU  
TECHNOLOGY  
CONFERENCE

## What is Thrust?

- High-Level Parallel Algorithms Library
- Parallel Analog of the C++ Standard Template Library (STL)
- Performance-Portable Abstraction Layer
- Productive way to program CUDA

Main developers: [Jared Hoberock](#) y [Nathan Bell](#)

# Thrust

Thrust Content from GTC 2012: Nathan Bell

GPU  
TECHNOLOGY  
CONFERENCE

## Easy to Use

- Distributed with CUDA Toolkit
- Header-only library
- Architecture agnostic
- Just compile and run!

```
$ nvcc -O2 -arch=sm_20 program.cu -o program
```

# Thrust

Thrust Content from GTC 2012: Nathan Bell

TECHNOLOGY  
CONFERENCE

GPU

## Productivity

### ■ Containers

- `host_vector`
- `device_vector`

### ■ Memory Management

- Allocation
- Transfers

### ■ Algorithm Selection

- Location is implicit

```
// allocate host vector with two elements
thrust::host_vector<int> h_vec(2);

// copy host data to device memory
thrust::device_vector<int> d_vec = h_vec;

// write device values from the host
d_vec[0] = 27;
d_vec[1] = 13;

// read device values from the host
int sum = d_vec[0] + d_vec[1];

// invoke algorithm on device
thrust::sort(d_vec.begin(), d_vec.end());

// memory automatically released
```

# Thrust High Level Interface

```
#include <thrust/host_vector.h>
#include <thrust/device_vector.h>
#include <thrust/generate.h>
#include <thrust/sort.h>
#include <thrust/copy.h>
#include <algorithm>
#include <cstdlib>

int main(void)
{
    // generate 32M random numbers serially
    thrust::host_vector<int> h_vec(32 << 20);
    std::generate(h_vec.begin(), h_vec.end(), rand);

    // transfer data to the device
    thrust::device_vector<int> d_vec = h_vec;

    // sort data on the device (846M keys per second on GeForce GTX 480)
    thrust::sort(d_vec.begin(), d_vec.end());

    // transfer data back to host
    thrust::copy(d_vec.begin(), d_vec.end(), h_vec.begin());

    return 0;
}
```

<http://thrust.github.io/>

# Simulating a large elastic String in a Random Medium using the GPU

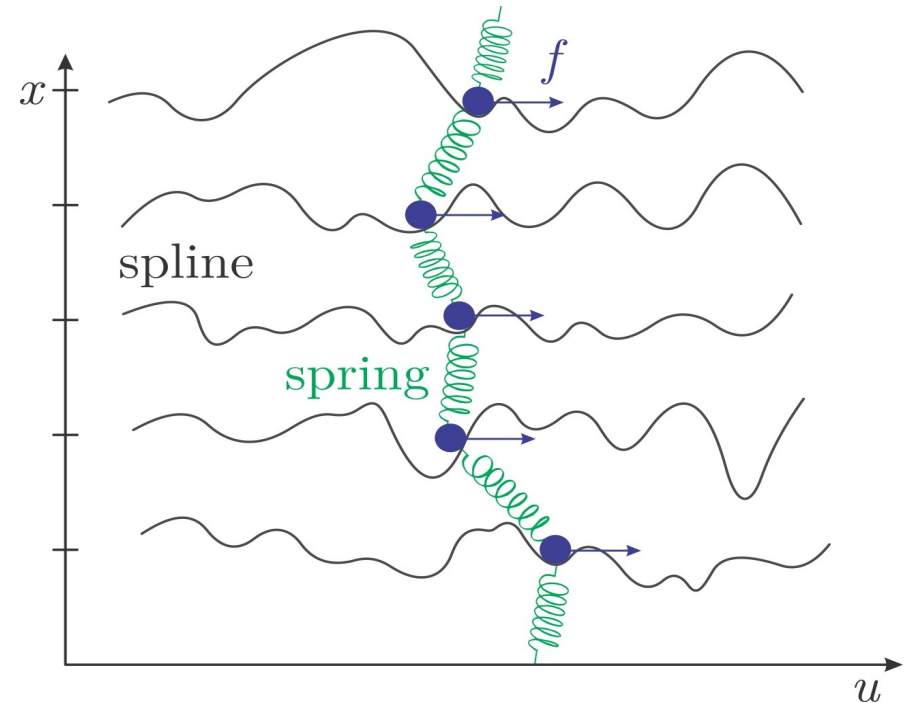
```

// #include<...>
using namespace thrust;
int main()
{

// vectors in GPU global memory
// declaration and allocation:
device_vector<REAL> u(L); // mi pared (discretizada)
device_vector<REAL> Ftot(L); // la fuerza sobre mi pared

// iterators on GPU global memory: they define ranges
// necessary for applying parallel algorithms
device_vector<REAL>::iterator u_begin = u.begin();
device_vector<REAL>::iterator Ftot_begin = Ftot.begin();

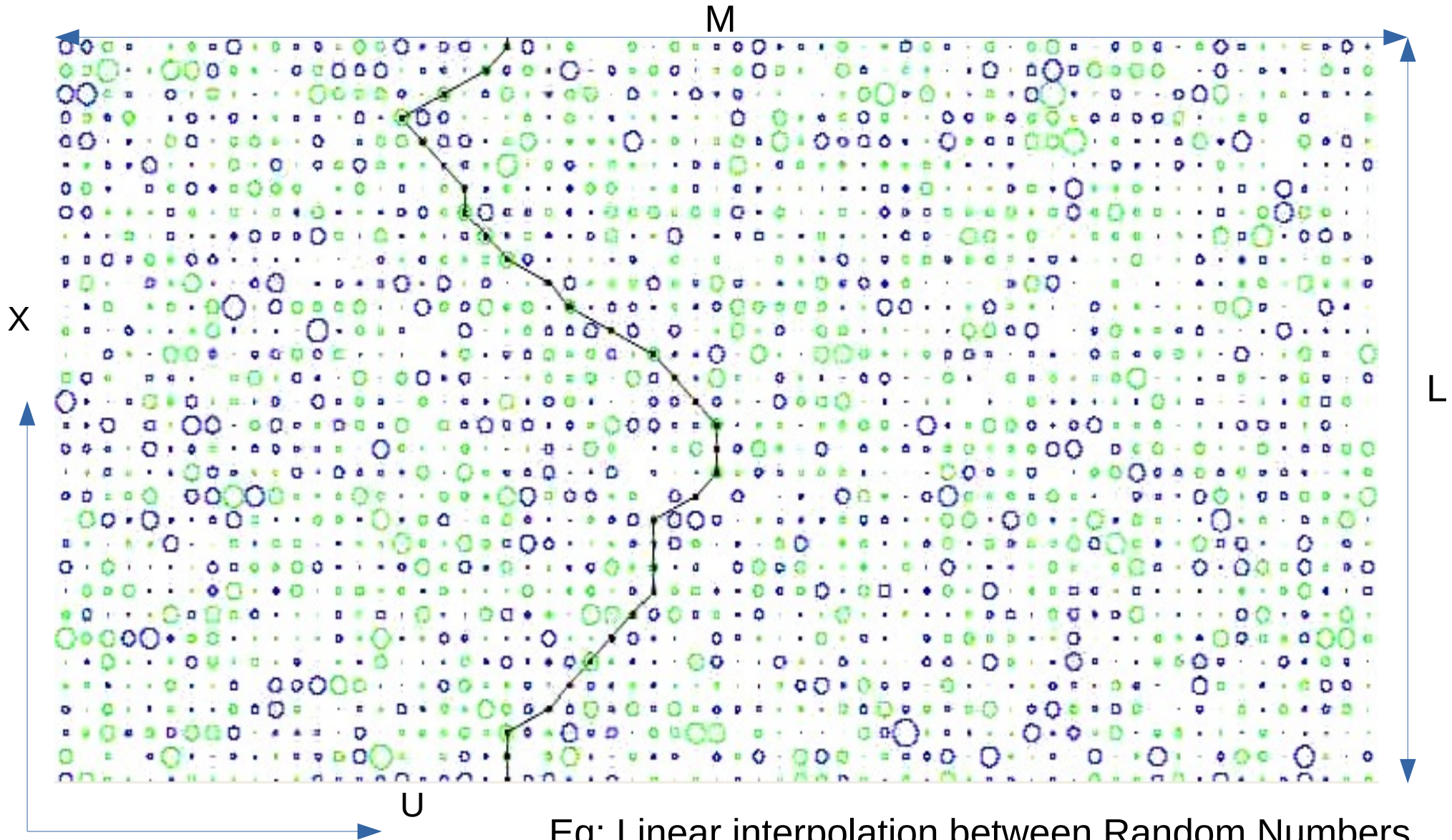
// temporal evolution
for(long n=0;n<Trun;n++)
{
    ....
    // (1) parallel algorithm in the GPU → Ftot(X,n)
    ....
    // (2) parallel algorithm in the GPU → u(X,n+1) en la GPU
    ....
    // (3) parallel algorithm in the GPU → properties
    ....
    // (4) Copies GPU -> CPU (reduced to a minimum)
}
}
    
```



$$F_{tot}(X, n) = C[u(X - 1, n) + u(X + 1, n) - 2u(X, n)] + F_p[u(X, n), X] + f + \sqrt{\frac{2T}{\Delta t}} R(X, n)$$

$$u(X, n + 1) = u(X, n) + F_{tot}(X, n)\Delta t$$

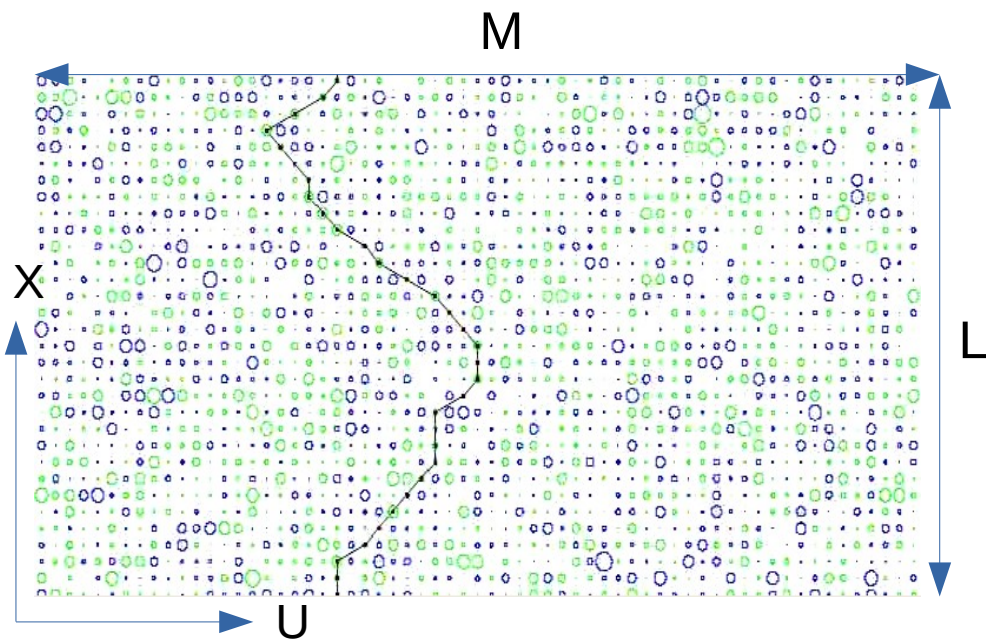
# Disordered landscapes



Eg: Linear interpolation between Random Numbers

$$F_p[u(X, t), X] = f_p[U, X] + (f_p[U + 1, X] - f_p[U, X])(u - U)$$

# The problem with the disorder...




- Need  $L \times M$  samples, with:  
 $L > 2^{16} = 65536$  y  $M \sim L^{\{5/4\}} \sim 2^{20}$ .
- Disorder Matrix  $> 2^{\{16+20\}}$  floats  
→ **256 Gb** (!!).

```
koltona@mostro:/opt/cudasdk/C/bin/linux/release$ ./deviceQuery
```

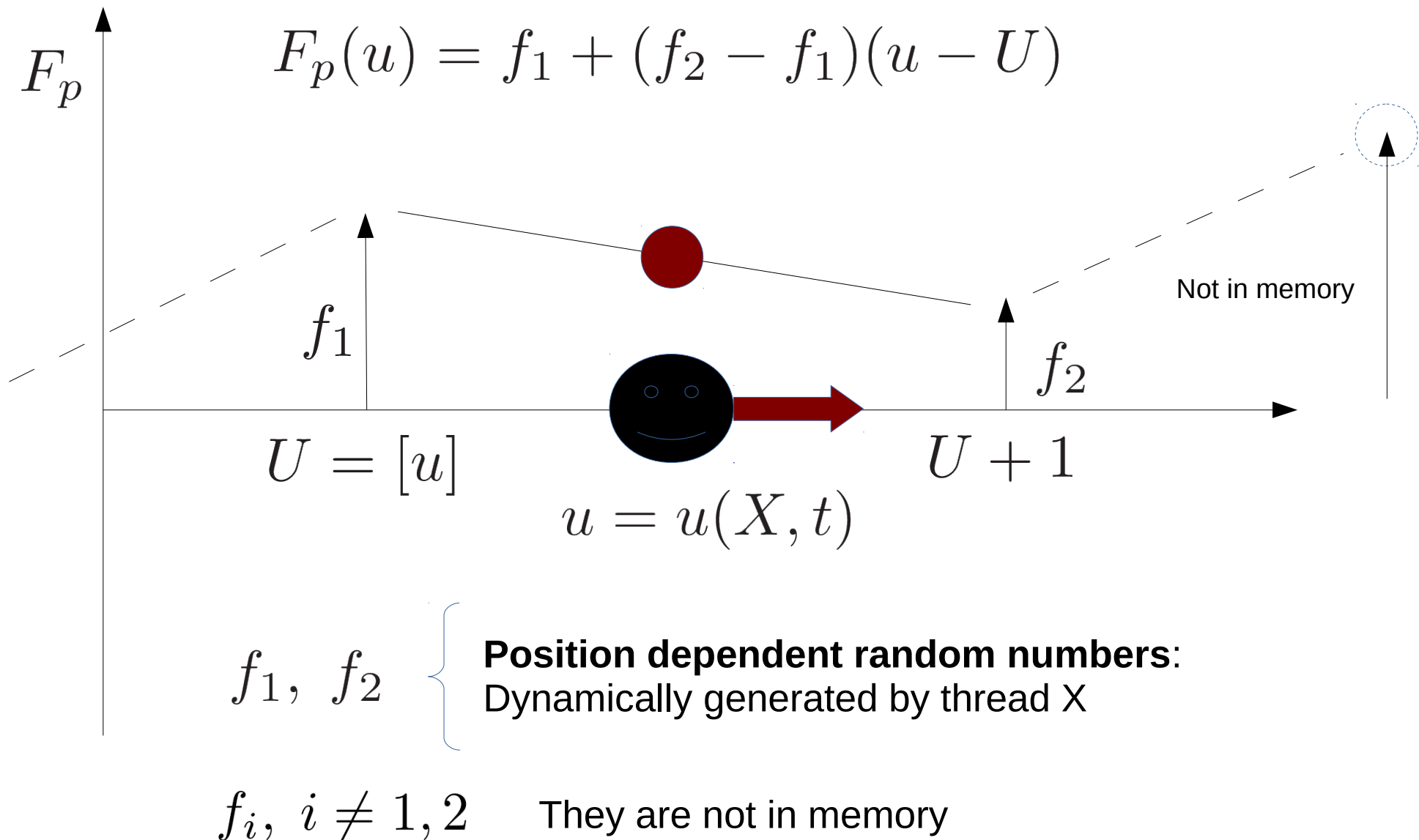
## Device 1: "Tesla C2075"

CUDA Driver Version / Runtime Version    4.2 / 4.1  
CUDA Capability Major/Minor version number:  2.0  
Total amount of global memory:            **5375 MBytes** (5636554752 bytes)  
(14) Multiprocessors x (32) CUDA Cores/MP:  448 CUDA Cores

- 
- Save disorder in device memory is not an option!
  - Simplest solution → ***Dynamically Generate*** using a parallel RNG.

# Dynamically generated disorder

Example: Linearly interpolated quenched random force





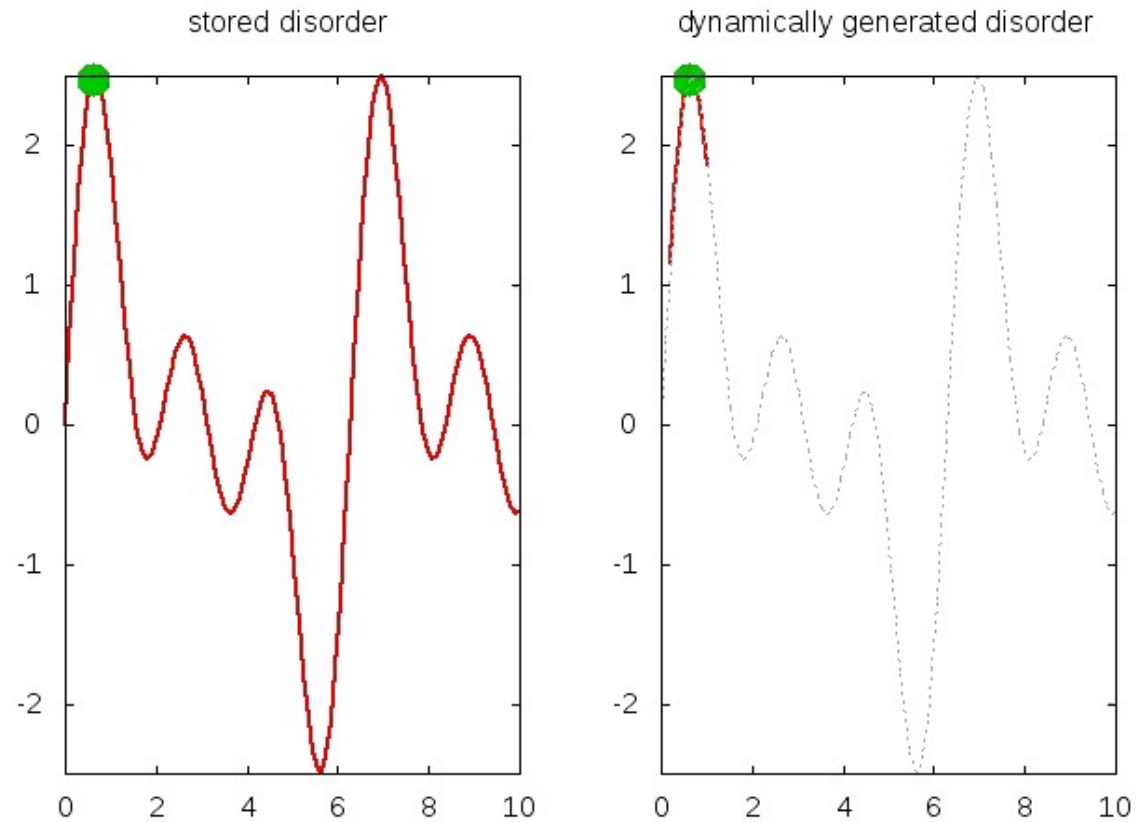
# Random number generators for massively parallel simulations on GPU

M. Manssen, M. Weigel, A. K. Hartmann

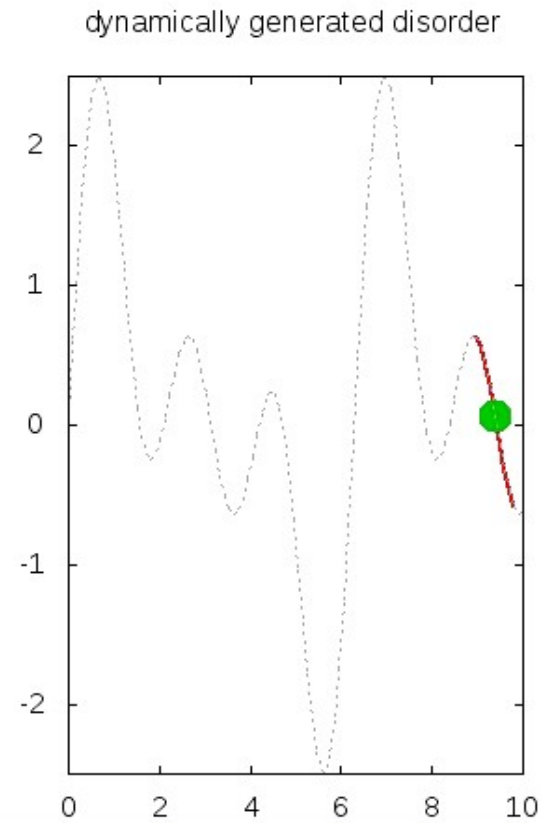
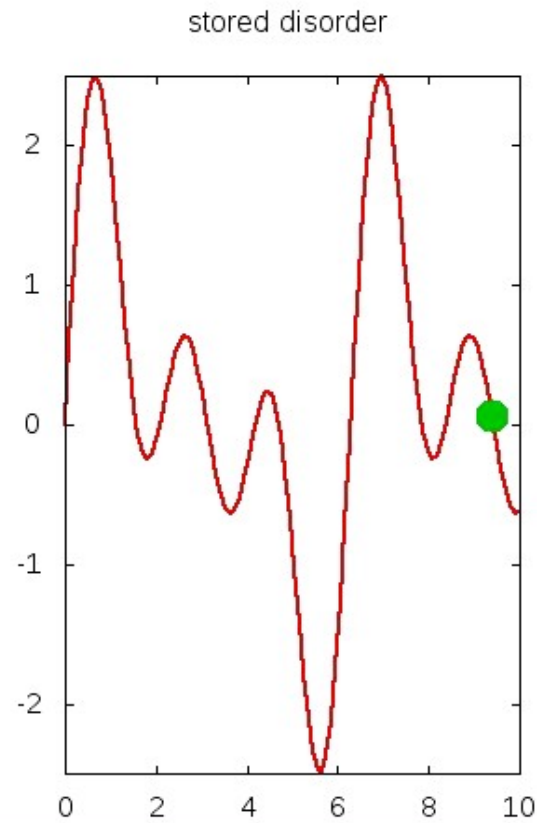
generator	bits/thread	failures in TestU01			Ising test	perf. $\times 10^9/s$
		SmallCrush	Crush	BigCrush		
LCG32	32	12	—	—	failed	58
LCG32, random	32	3	14	—	passed	58
LCG64	64	None	6	—	failed	46
LCG64, random	64	None	2	8	passed	46
MWC	64 + 32	1	29	—	passed	44
Fibonacci, $r = 521$	$\geq 80$	None	2	—	failed	23
Fibonacci, $r = 1279$	$\geq 80$	None	(1)	2	passed	23
XORWOW (cuRAND)	192	None	None	1/3	failed	19
MTGP (cuRAND)	$\geq 44$	None	2	2	—	18
XORShift/Weyl	32	None	None	None	passed	18
Philox4x32_7	(128)	None	None	None	passed	41
Philox4x32_10	(128)	None	None	None	passed	30

**Which one?**

# Dynamically Generated quenched disorder



# Dynamically Generated quenched disorder



# Need for a reversible random generator

Climate (thermal noise) might not,  
but the landscape (quenched disorder) must be reversible!



# Random number generators for massively parallel simulations on GPU

M. Manssen, M. Weigel, A. K. Hartmann

*Regeneración sencilla de números ya generados?*

generator	bits/thread	failures in TestU01			Ising test	perf. $\times 10^9/s$
		SmallCrush	Crush	BigCrush		
LCG32	32	12	—	—	failed	58
LCG32, random	32	3	14	—	passed	58
LCG64	64	None	6	—	failed	46
LCG64, random	64	None	2	8	passed	46
MWC	4 + 32	1	29	—	passed	44
Fibonacci, $r = 521$	$\geq 80$	None	2	—	failed	23
Fibonacci, $r = 1279$	$\geq 80$	None	(1)	2	passed	23
XORWOW (cuRAND)	192	None	None	1/3	failed	19
MTGP (cuRAND)	$\geq 44$	None	2	2	—	18
XORShift/Weyl	32	None	None	None	passed	18
Philox4x32_7	(128)	None	None	None	passed	41
Philox4x32_10	(128)	None	None	None	passed	30

DIFFICULT

Counter-based RNGs

*Which one?*

# Counter-Based RNGs



## Parallel Random Numbers: As Easy as 1, 2, 3

John K. Salmon, Mark A. Moraes, Ron O. Dror, David E. Shaw

“.. We demonstrate that independent, **keyed transformations of counters** produce a large alternative class of PRNGs with **excellent statistical properties**... are ideally suited to modern multicore CPUs, GPUs, clusters, and special-purpose hardware because they vectorize and parallelize well, and **require little or no memory for state**. ... All our PRNGs pass **rigorous statistical tests** and produce at least  **$2^{64}$  unique parallel streams of random numbers, each with period  $2^{128}$**  or more. In addition to essentially unlimited parallel scalability, our PRNGs offer excellent single-chip performance: **Philox is faster than the CURAND library on a single NVIDIA GPU**.

# Counter-Based RNGs

## Overview

counter-based RNGs are stateless functions whose arguments are a counter, and a key and whose return value is the same type as the counter.

**value = CBRNGname(counter, key)**

The returned value is a *deterministic function of the key and counter*, i.e. a unique (counter, key) tuple will always produce the same result. The result is highly sensitive to small changes in the inputs, so that the sequence of values produced by simply incrementing the counter (or key) is effectively indistinguishable from a sequence of samples of a uniformly distributed random variable.

Exactamente lo que necesitamos!

$$F_p[u(X, t), X] = f_p[U, X] + (f_p[U + 1, X] - f_p[U, X])(u - U)$$

CBRNG(U,X)                      CBRNG(U,X)

Counter = U = (int)u  
Key = X = thread/particle ID

# Counter-Based RNGs

Exactly what we need

- Good quality, state-less
- **Easy to use**
- **Can recover RNs**

DESORDEN

$$F_p[u(X, t), X] = f_p[U, X] + (f_p[U + 1, X] - f_p[U, X])(u - U)$$

CBRNG(U,X)

CBRNG(U+1,X)

Counter = U = (int)u  
Key = X = thread/particle ID

TEMPERATURA

$$R(X, t) = \text{CBRNG}'(t, X)$$

Counter = n (tiempo)  
Key = X = thread/particle ID

Force update and Euler step on each particle

$$F_{tot}(X, n) = C[u(X - 1, n) + u(X + 1, n) - 2u(X, n)] + F_p[u(X, n), X] + f + \sqrt{\frac{2T}{\Delta t}} R(X, n)$$

In parallel  
X=0,1,...,L-1

► Synchronize threads

$$u(X, n + 1) = u(X, n) + F_{tot}(X, n)\Delta t$$

In parallel  
X=0,1,...,L-1



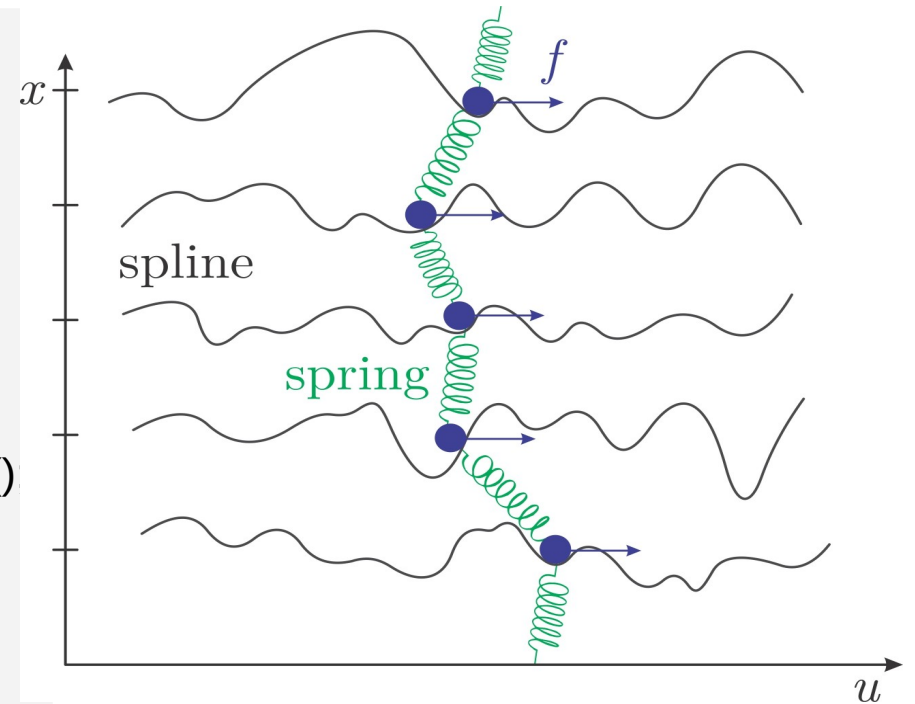
# GPU fast implementation

```
// #include<...>
using namespace thrust;
int main()
{

    device_vector<REAL> u(L);
    device_vector<REAL> Ftot(L);

    device_vector<REAL>::iterator u_begin = u.begin();
    device_vector<REAL>::iterator Ftot_begin = Ftot.begin()

    for(long n=0;n<Trun;n++)
    {
        ....
        // (1) parallel stencil → Ftot(X,n) en la GPU
        ....
        // (2) parallel map → u(X,n+1) en la GPU
        ....
        // (3) parallel reductions → properties (GPU)
        ....
        // GPU -> CPU copies (reduced to a minimum)
    }
    ....
}
```



# GPU and multicore CPU parallel implementation

```
// #include<...>
using namespace thrust;
int main()
{

    device_vector<REAL> u(L);
    device_vector<REAL> Ftot(L);

    device_vector<REAL>::iterator u_begin = u.begin();
    device_vector<REAL>::iterator Ftot_begin =
Ftot.begin();

    for(long n=0;n<Trun;n++)
    {
        ....
        // (1) calculo Ftot(X,n) en la GPU
        ....
        // (2) calculo u(X,n+1) en la GPU
        .....
        // (3) calculo propiedades (GPU)
        .....
        // GPU -> CPU copies (reduced to a minimum)
    }
    ....
}
```

## Same code!!

- **In GPU**
  - ✓ `nvcc -o stringCUDA string.cu`
- **CPU-MULTICORE**
  - ✓ `cp string.cu string.cpp`
  - ✓ `g++ -o stringOMP string.cpp -O2 -fopenmp -DTHRUST_DEVICE_BACKEND=THRUST_DEVICE_BACKEND_OMP -lgomp`

Thrust Device backends  
<http://thrust.github.io/>

***PORTABILITY***  
***any GPU, CPU***  
***Portable parallel performance***

# Real Time Simulation on this laptop

CUDA Device Query (Runtime API) version (CUDART static linking)

Detected 1 CUDA Capable device(s)

Device 0: "GeForce GT 620M"

CUDA Driver Version / Runtime Version          6.0 / 5.5

CUDA Capability Major/Minor version number:    2.1

Total amount of global memory:                  1024 MBytes (1073479680 bytes)

( 2) Multiprocessors, ( 48) CUDA Cores/MP:    96 CUDA Cores

GPU Clock rate:                                    1250 MHz (1.25 GHz)

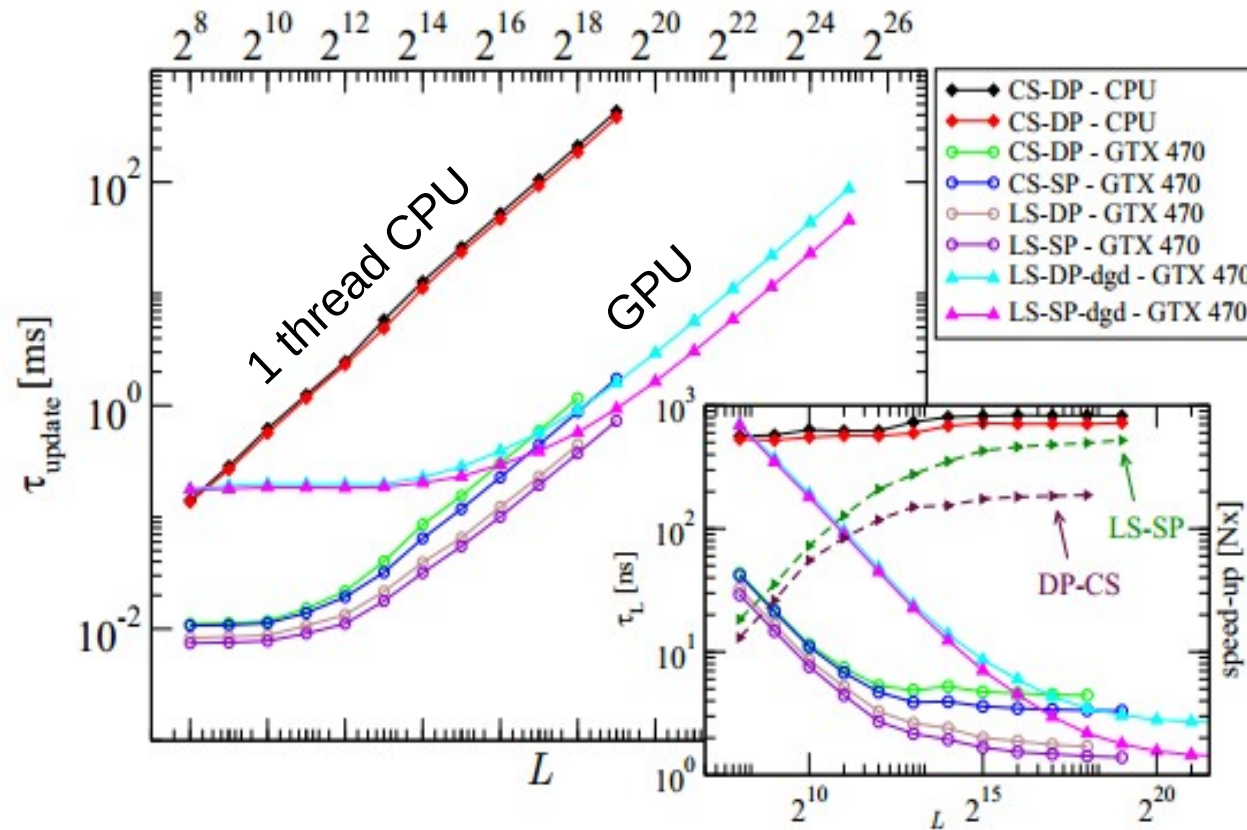
Memory Clock rate:                                900 Mhz

Memory Bus Width:                                64-bit

ASUS Zenbook, ubuntu 14.04

Intel(R) Core(TM) i5-3317U CPU @ 1.70GHz

# Performance !!!



- **33554432** particles: largest model system to date.

- **Theoretical predictions:** Scaling corrections unveiled → Most Accurate critical exponents for the depinning transition to date.

- **Low cost !!** (100-900 Euros) Same GPUs used for computer games.

- **CPU:** AMD Phenom(tm) II X4 955 Processor @3.2GHz
- **GPU:** NVIDIA GTX 470.

# Theoretical Results

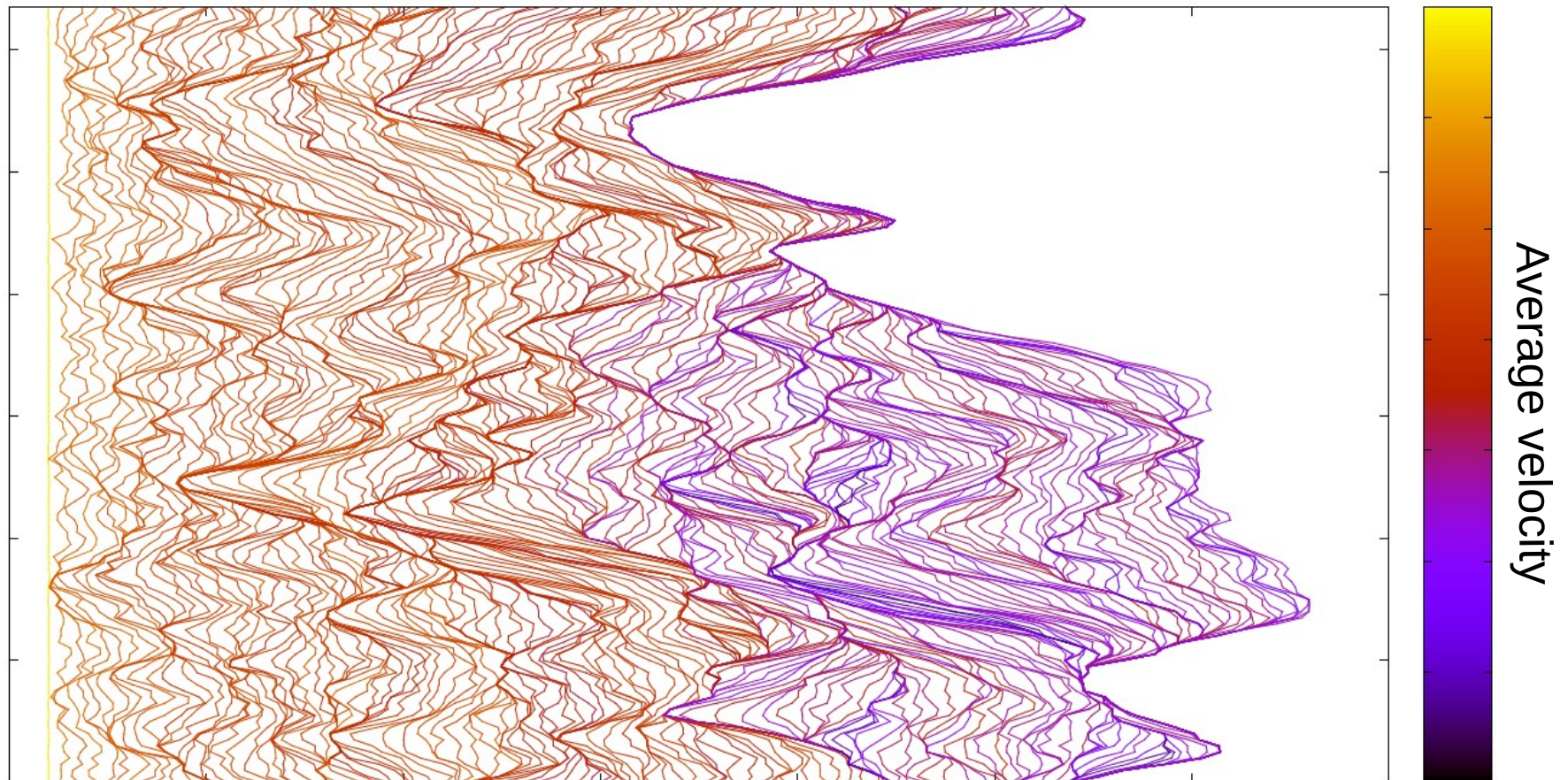
PHYSICAL REVIEW E **87**, 032122 (2013)

## Nonsteady relaxation and critical exponents at the depinning transition

E. E. Ferrero, S. Bustingorry, and A. B. Kolton

*CONICET, Centro Atómico Bariloche, 8400 San Carlos de Bariloche, Río Negro, Argentina*

(Received 28 November 2012; revised manuscript received 4 February 2013; published 11 March 2013)



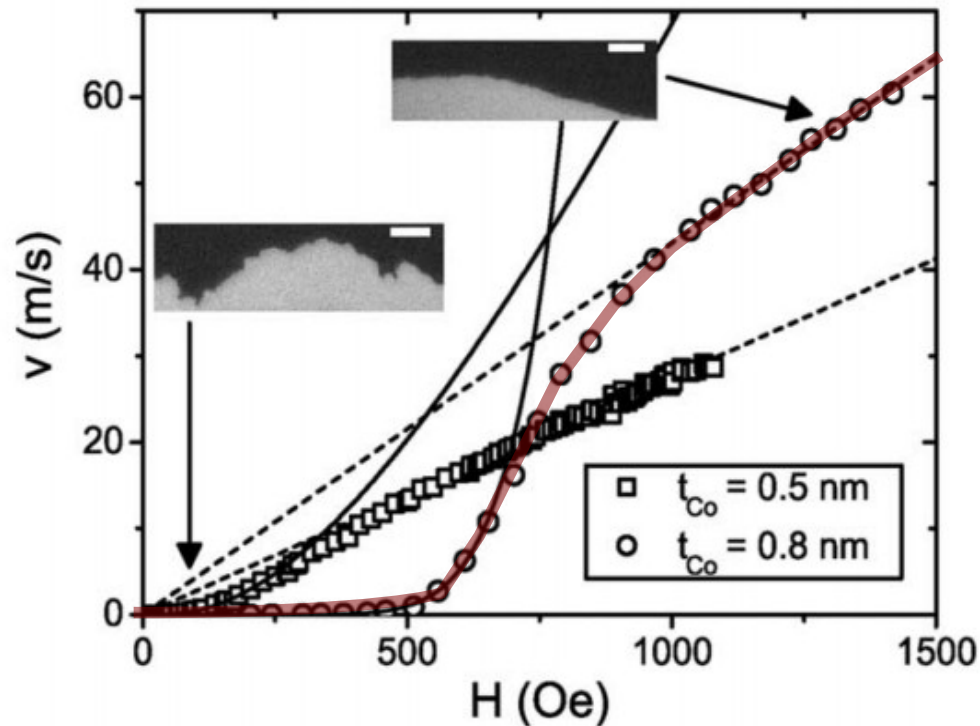
But... do magnetic domain walls in thin films  
behave as simple elastic strings in  
disordered media?

# Qualitatively OK!

Experiment

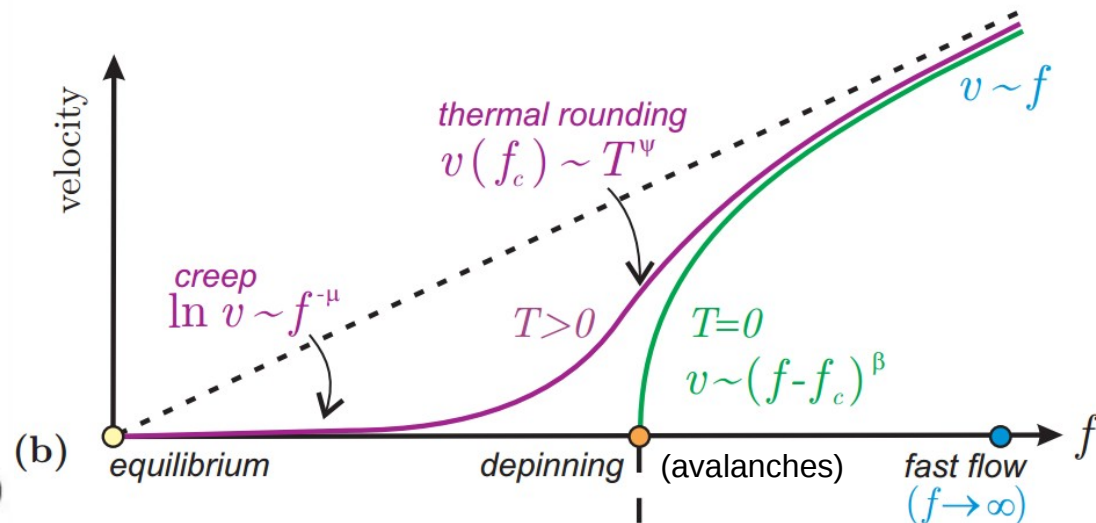
$$H \leftrightarrow f$$

Theory



Metaxas et al, PRL 2007  
Pt/Co/Pt films

- Scaling Arguments
- Functional renormalization group
- Numerical Simulations

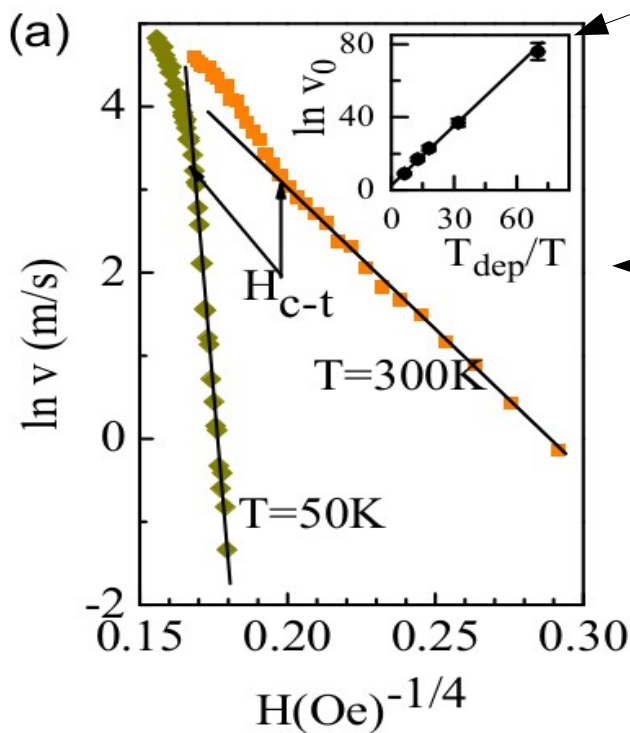
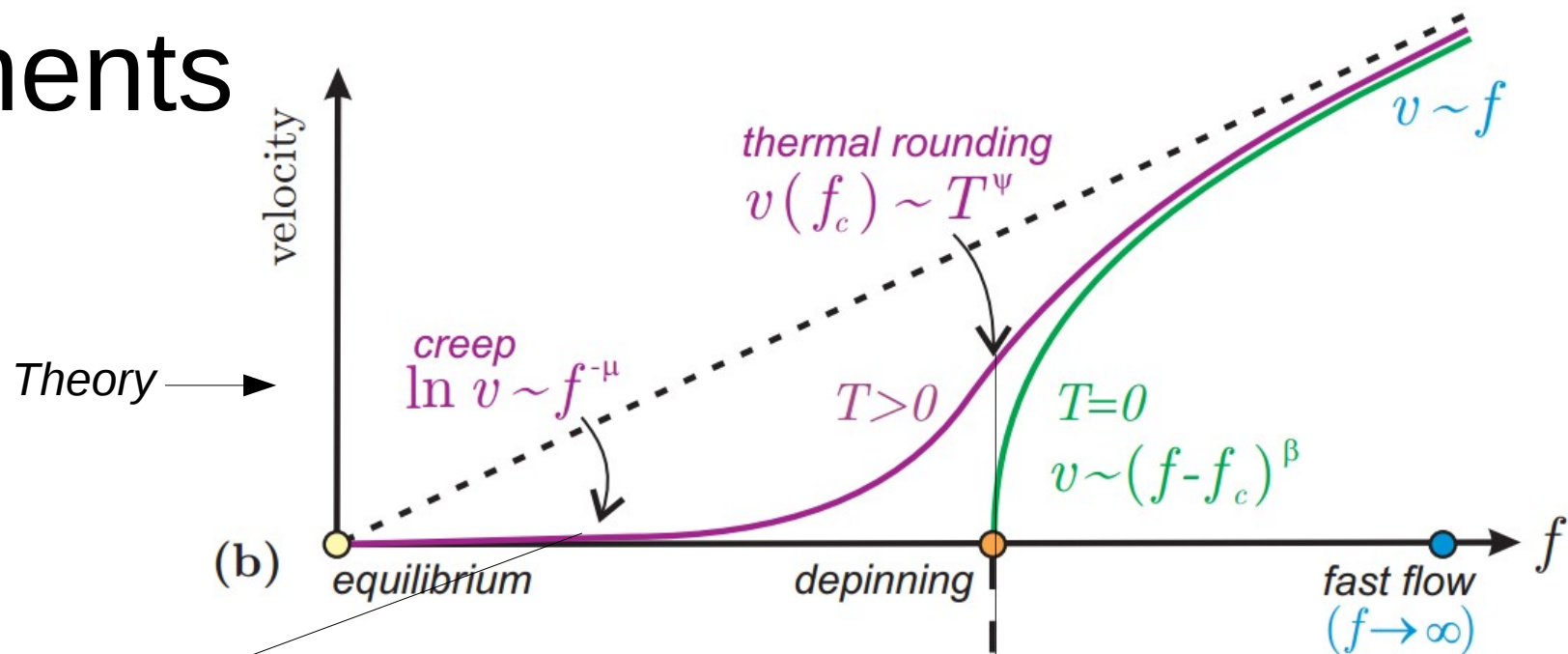


Three-reference nonequilibrium steady states

... and quantitatively?

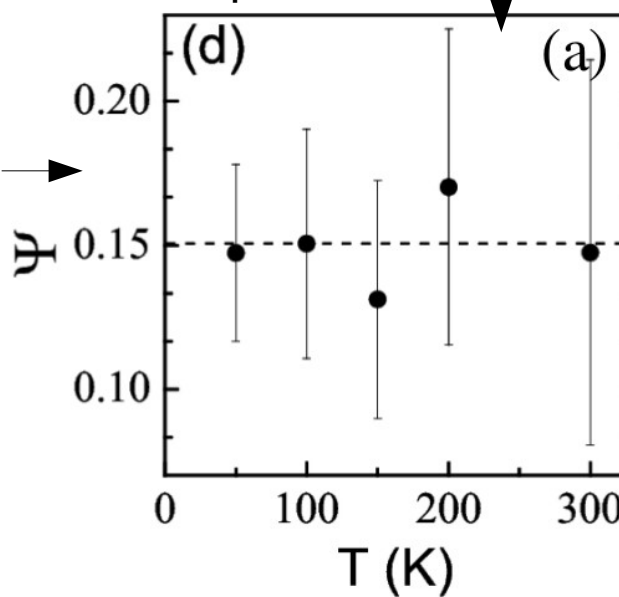
# Theory vs Experiments

$$\psi = 0.15, \beta = 0.245, \mu = 1/4$$



Experiments

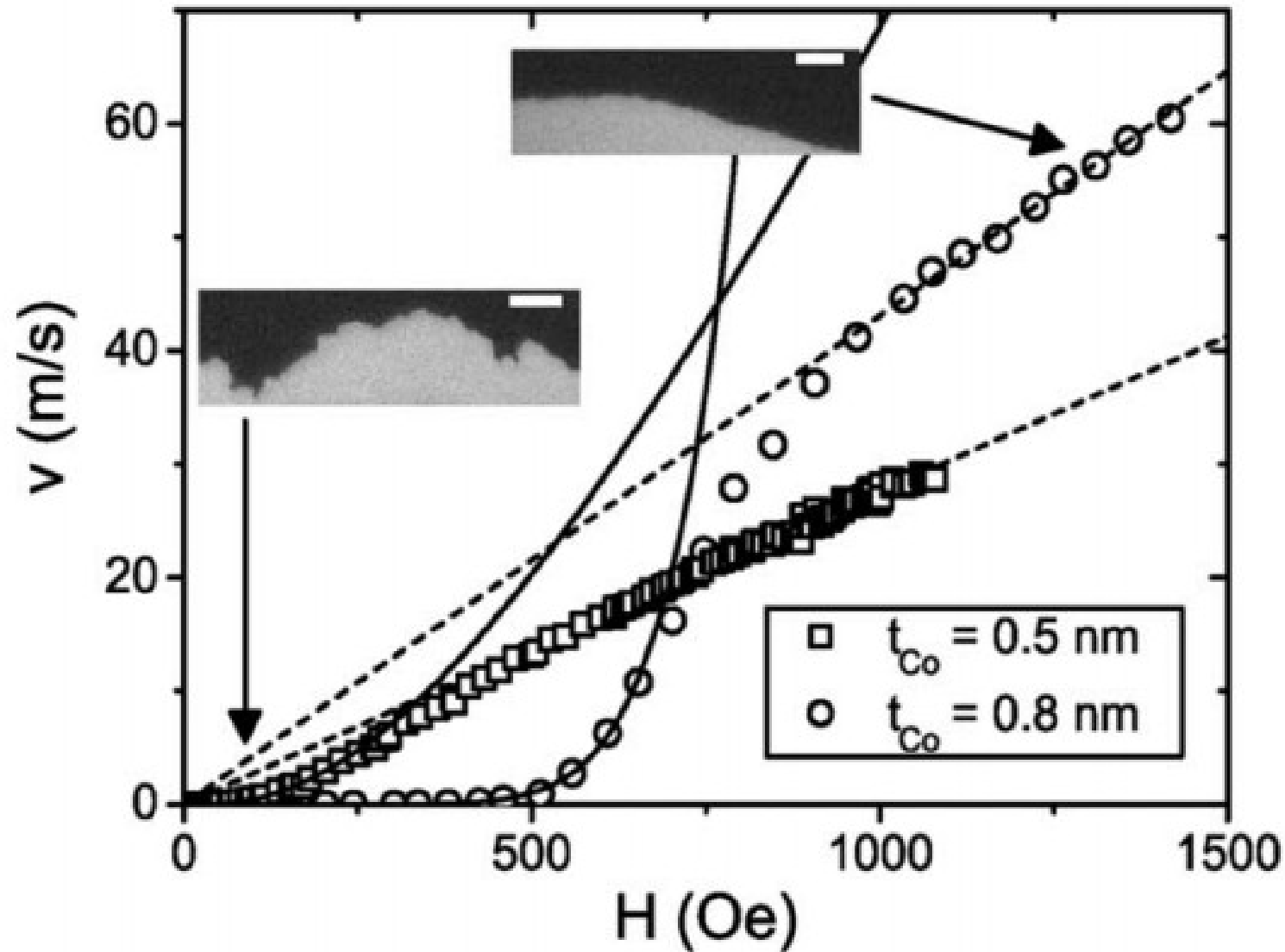
Experiment



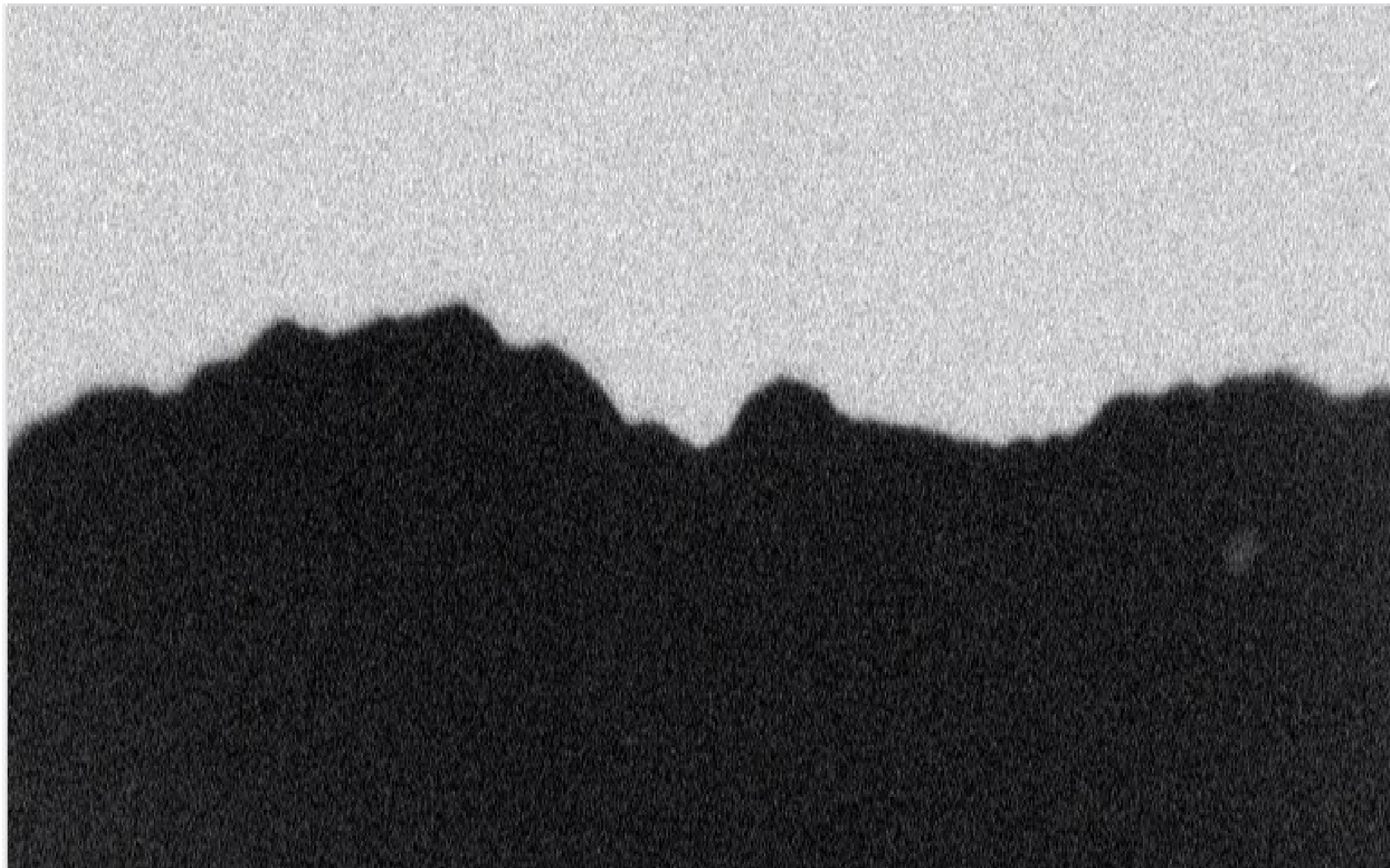
[J. Gorchon et al,  
PRL July 2014]



# Transport and Rough geometry



Is it moving?

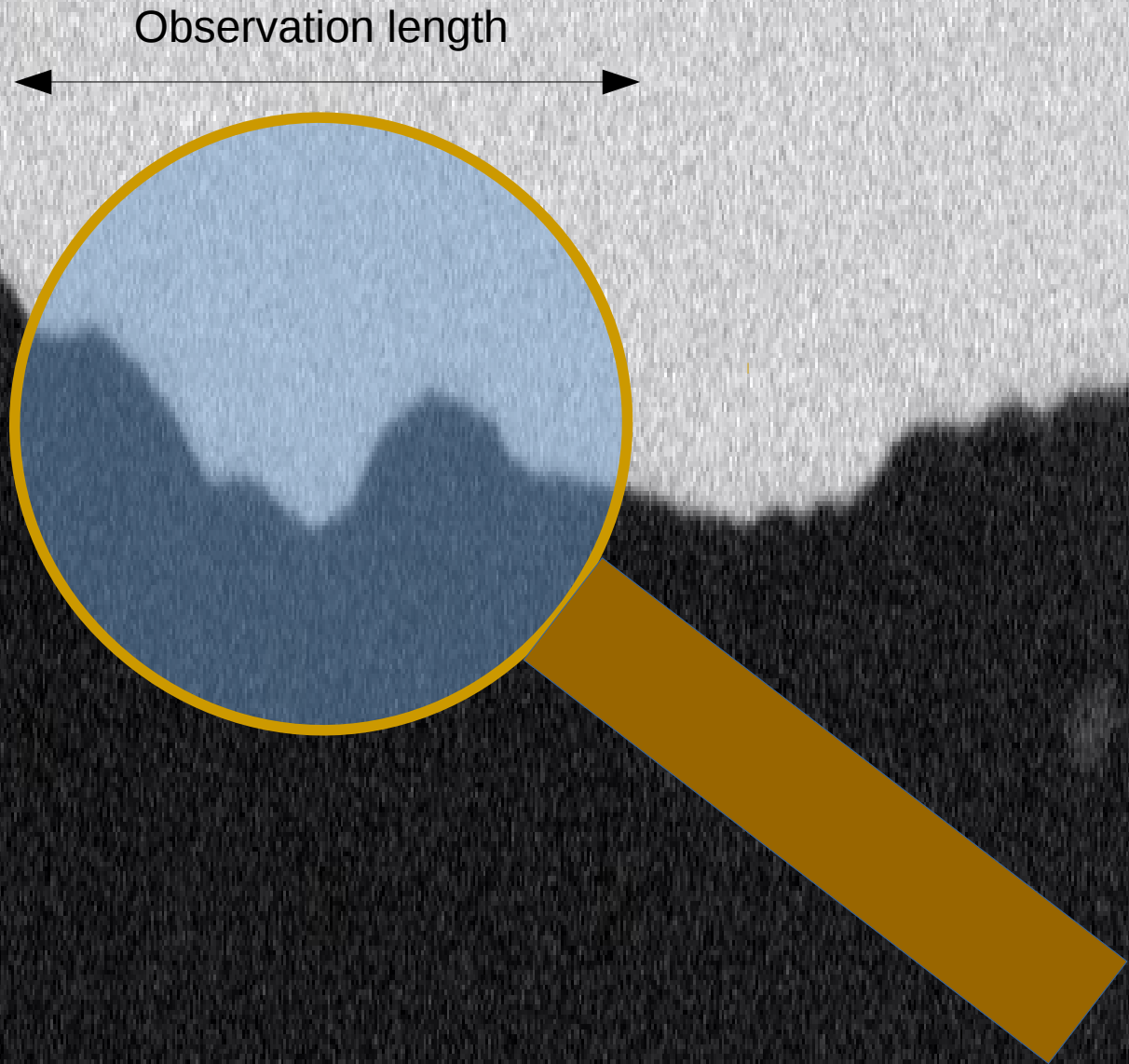


# Is it moving?



Top Secret! (1984)

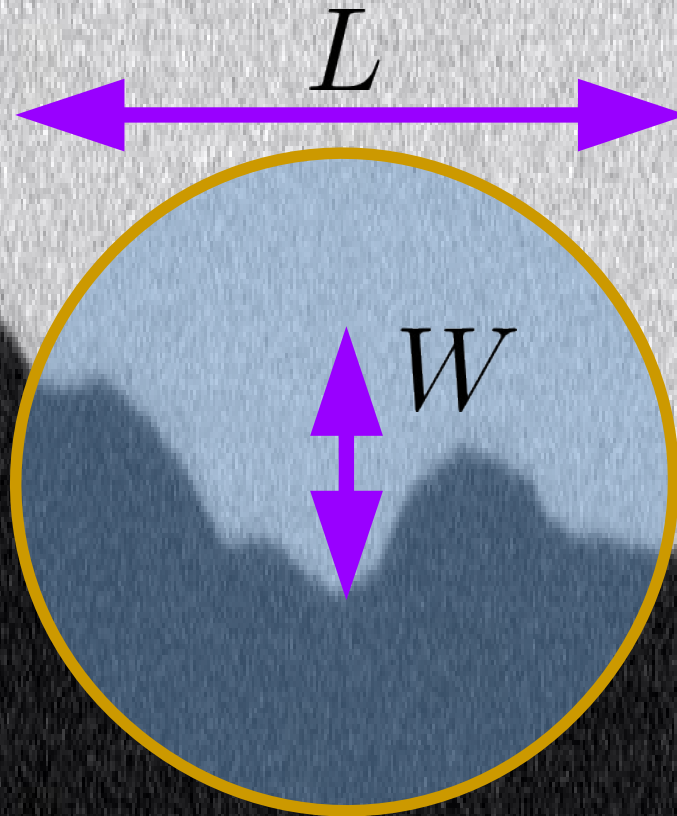
*Looking for a “hair-like” clue....*



*Rough Geometry*

$$W \sim L^\zeta$$

*One roughness exponent  
only for a self-affine  
Interface [cf diffusion]*

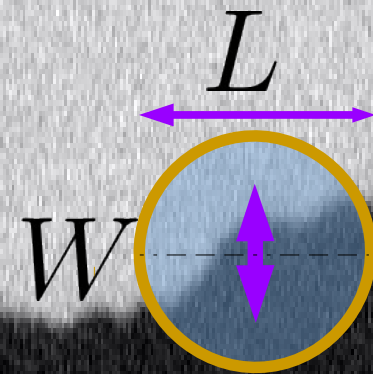
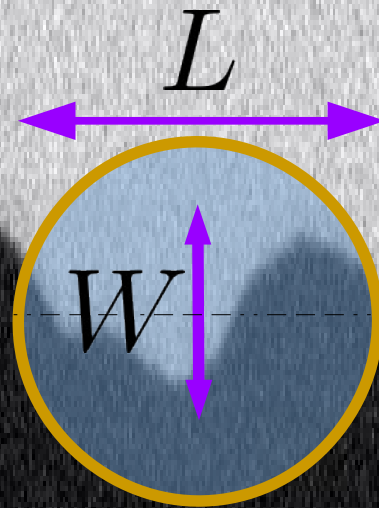
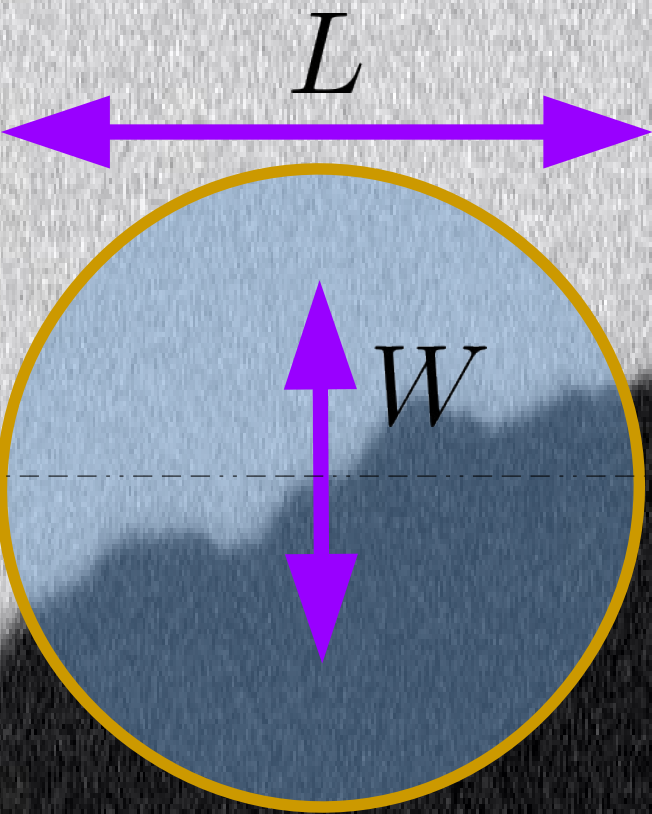


**UNIVERSAL ROUGHNESS EXPONENTS**

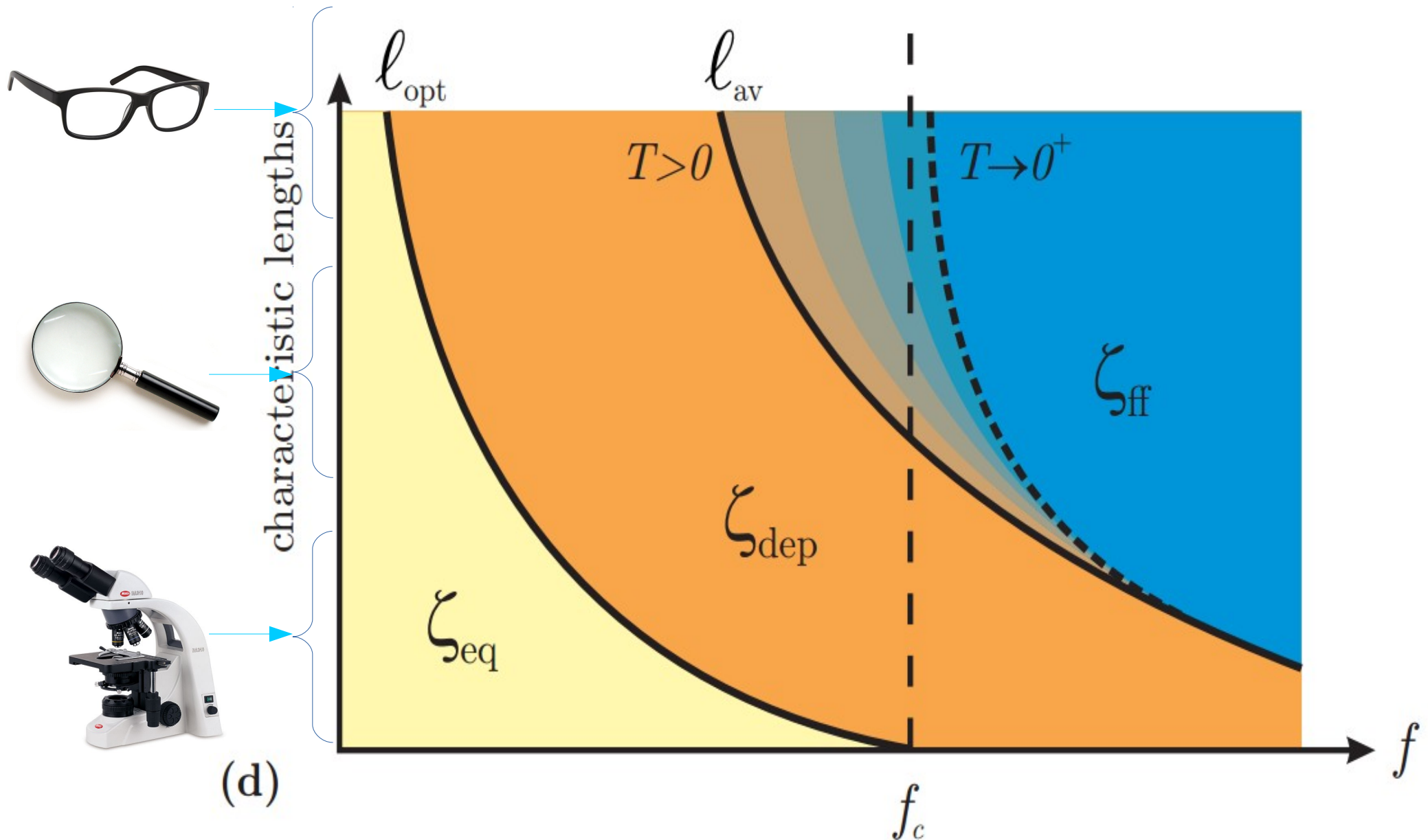
Does  $\zeta$  depend on  $L$ ?

*Rough Geometry*

$$W_L \sim L^\zeta$$



# The “hair” of the moving elastic string



ABK, A. Rosso, W. Krauth, T. Giamarchi, PRL (2006), PRB (2009)  
E. Ferrero, S. Bustingorry, A. B. Kolton, A. Rosso, Comptes Rendus (2013)

# Is it moving?

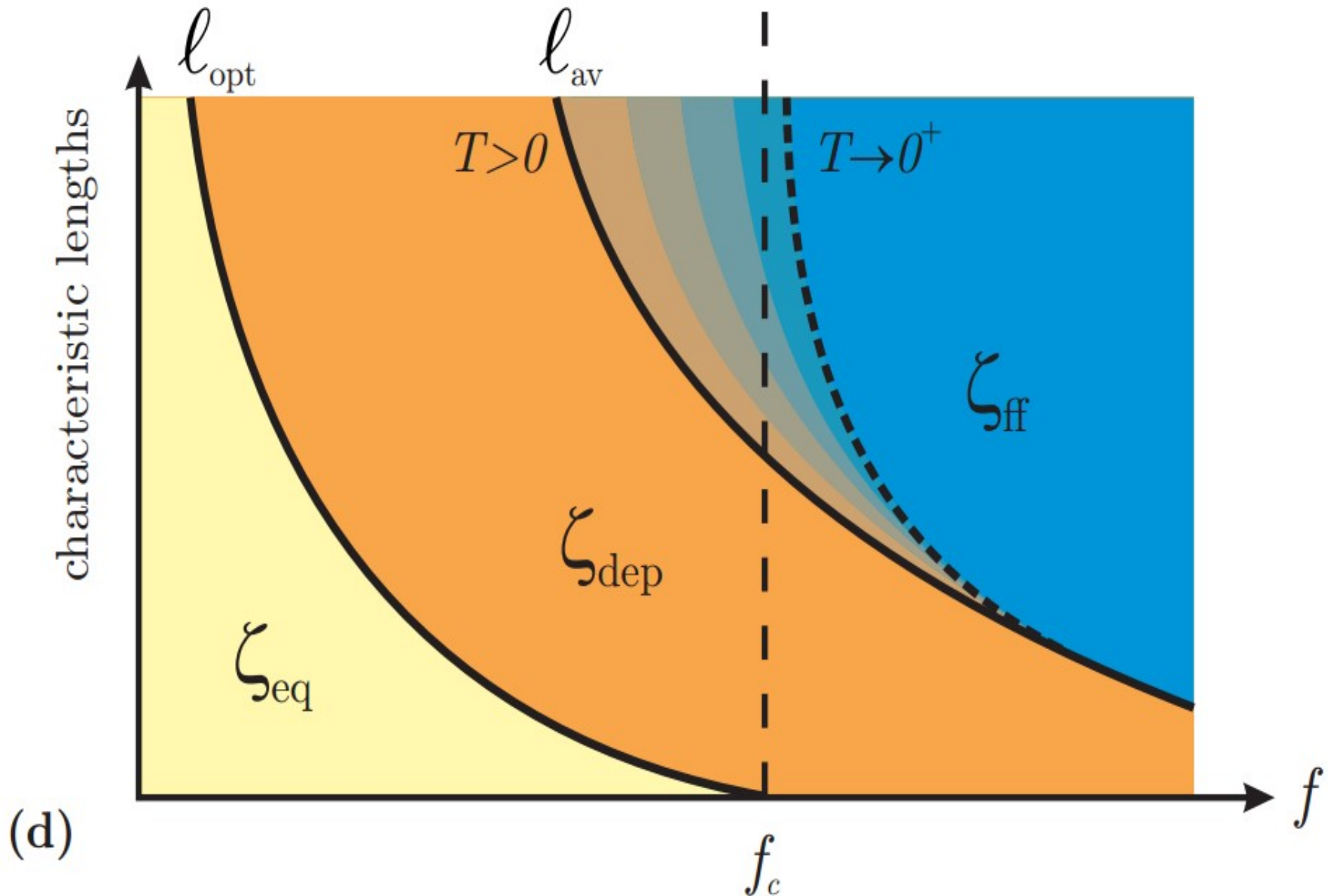


Top Secret! (1984)

We forgot to eliminate memory effects!



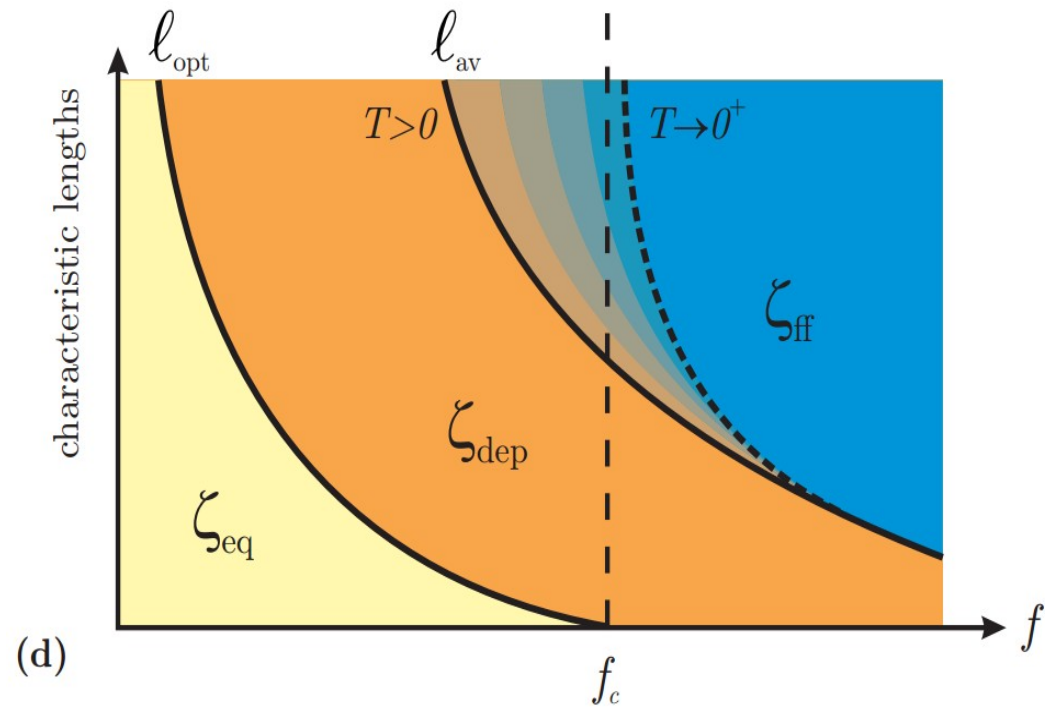
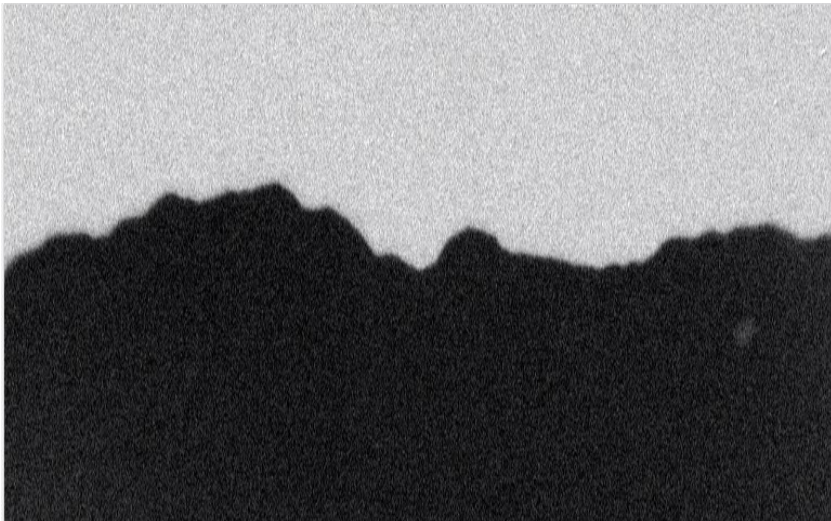
# ***Steady-State*** dynamical diagram



# Is it moving?



## Theory: Observe its rough geometry



*Roughness is the clue for understanding the motion !!*

# Conclusions

- **Motivation:**
  - Elasticity vs Disorder: many disordered systems display a similar effective physics → minimal models allow us to describe them all.
- **Concepts:**
  - Universality, collective transport, roughness.
- **Playing the game:**
  - Parallel finite difference scheme
  - Why in GPU?
  - Quenched disorder tricks and other details.
- **Comparison with Experiments:**
  - We can interpret measurements...
  - we can make quantitative predictions!

Thank you for the attention!

Questions?