Faculty of Science

Computational Astrophysics

1a Introduction

Troels Haugbølle, Kai Hendriks

Niels Bohr Institute
University of Copenhagen

Main messages to take away this week

- How the course works
  - Tools, **assignments, grading**, …
- Topics overview
  - Topics covered in the course
  - Example astrophysics application
- A start on understanding fluid dynamics
  - How can we solve partial differential equations
  - A lot is about "**transport**" of properties
  - Care is needed to keep methods stable

Practicalities

- Tools and mechanisms
  - Absalon
  - ERDA
  - Laptop tools

- Exercises
  - How to acquire them with ERDA, and how to submit answers
    - Working at ERDA
    - Working on your laptops
  - Advection
    - Stepwise Python assignments

Weekly format

Sunday:

- Before 11:59; hand in exercises from last week

Tuesday 9-12:

- Walk-through of last week's exercise
- Introduction to this week's theme
- Introductory exercise

Thursday 9-12, 13-16:

- Morning: In-depth lecture, intro to exercise
- Afternoon: Main exercise time

Course Plan:

Week	Date	Chapter	Topic
1	22+24/11	1, 2, 6.3.2	Intro, PDEs, and advection
2	29/11+1/12	6	Fluid dynamics
3	6+8/12	8	Magnetohydrodynamics
4	13+15/12	7	Selfgravity
5	20/12+22/12	Absalon	Chemistry & thermodynamics
6	3+5/1	1.6 + 9	Radiation transfer
7	10+12/1	3 + Absalon	Collisionless dynamics
8	17-23/1		Project week
9	26/1		Exam

Course Plan:

It's a really good idea to prepare for each week by reading the book (and browsing the net!)

time saved > time used !

Textbook

P. Bodenheimer, G. P. Laughlin, M. Rozyczka, T. Plewa, H. W. Yorke: "Numerical Methods in Astrophysics: An Introduction", ...netohydrodynamics

Week	Date	Chapter	
1	22+24/11	1, 2, 6.3.2	
2	29/11+1/12	6	
4	13+15/12	7	Selfgravity
5	20/12+22/12	Absalon	Chemistry & thermodynamics
8	17-23/1		Project week
9	26/1		Exam

Exam: examination consists of a 72 hr written project, with a small 10 minutes oral presentation. You can choose to do the project either Wednesday-Saturday (18-21/1) or, in case this works better with your other exams, Friday-Monday (20-23/1). The project starts at 12 noon and ends at 12 noon. The oral presentation of the project is Thursday 26/1.

Course format – Lessons, Exercises & Projects

- Lessons: Tuesday and Thursday morning
- Exercises = **assignments**
  - Solutions give **points** (≈120 pt/week) **to accumulate in Absalon**
  - Exercises can be submitted by **up to three persons together**
- Hard deadlines -> rapid feedback
- **Learning is by example**, with pointers to reference material
  - this is much more efficient than 'reading from A-Z'
- The course ends with **projects**, chosen from a list of suggestions
  - the possible projects will be discussed before choices are made
  - projects is a 72hr assignment with 'oral defense' the following day
  - we are flexible with project placement to accommodate w.r.t. other courses.
- The final grade is a combination of exercises (50%) and project (50%). At least 60% of exercises have to be turned in. I will disregard one week exercise; we can all have a bad day.

Computational Astrophysics, Examples

Why Computational Modeling?

Basically, because it today ***defines the front line of science*** in many research fields, and this is particularly true in astrophysics:

- It's the ***only way to make experiments in astrophysics***
  - Set up simulations; vary the overall conditions ("surrounding conditions")
- Simulations are now so realistic that they can match and exceed observational resolution
  - In classical "***backward analysis***" of observations one *tries to* derive the physical properties (density, pressure, temperature, velocity and magnetic fields) directly from observations.
  - In "***forward analysis***" one computes and smears synthetic images to instrument resolution, and can then ***compare the results, one-on-one***
- Simulations provide ***direct access to 4-D (space-time) data***, allowing understanding of which mechanisms are the decisive ones
  - Turbulence or gravity?
  - Local or global dynamics?
  - ...

Why Computational Modeling?

From this follows that

- Truly ground-braking research takes place at supercomputer centers
  - Access to such facilities are as competitive as access to ALMA, JWST, …
  - Hence, they act as filters, engaging the very best scientists
  - The fields of **Galaxies and Star & Planet Formation are good examples**
    - Tremendous progress (in understanding!) due to numerical simulations
    - In combination with observations, which need increasingly complex models

- It's crucially important to stay at the **technical forefront**
  - Computing capacity keeps following Moore's law (x2 every 18 months)
  - But the technology changes repeatedly
    - Currently towards many more, instead of much faster, processors
    - A strong trend is also the introduction of accelerators (GPUs)
    - Individual computers are getting more and more CPU cores, GPUs, and memory. F.x. one node in the upcoming LUMI supercomputer will reach 200 Tflops or the same as 3,000 CPU cores.

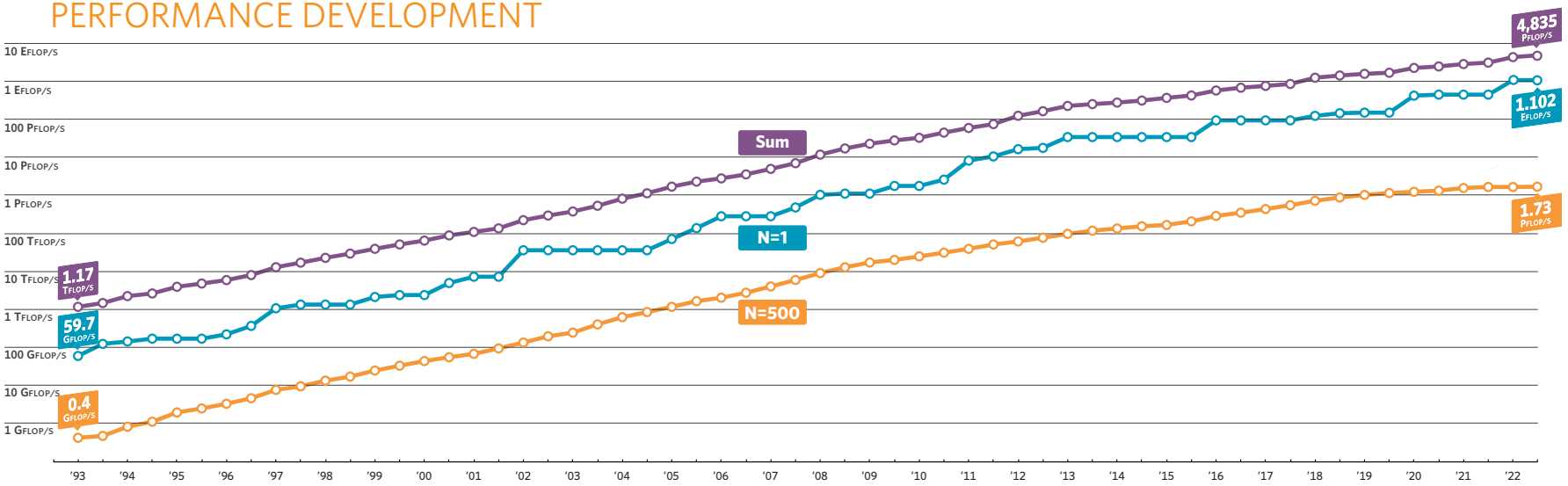
Why Computational Modeling?

TOP500 The List. FIND OUT MORE AT top500.org

NOVEMBER 2022

			SITE	COUNTRY	CORES	RMAX PFLOP/S	POWER MW
1	**Frontier**	HPE Cray EX235a, AMD Opt 3rd Gen EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-10	DOE/SC/ORNL	USA	8,730,112	1,102.0	21.1
2	**Fugaku**	Fujitsu A64FX (48C, 2.2GHz), Tofu Interconnect D	RIKEN R-CCS	Japan	7,630,848	442.0	29.9
3	**LUMI**	HPE Cray EX235a, AMD Opt 3rd Gen EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-10	EuroHPC/CSC	Finland	2,174,976	304.2	5.82
4	**Leonardo**	Atos Bullsequana intelXeon (32C, 2.6 GHz), NVIDIA A100 quad-rail NVIDIA HDR100 Infiniband	EuroHPC/CINEC	Italy	1,463,616	174.7	5.61
5	**Summit**	IBM POWER9 (22C, 3.07GHz), NVIDIA Volta GV100 (80C), Dual-Rail Mellanox EDR Infiniband	DOE/SC/ORNL	USA	2,414,592	148.6	10.1

PERFORMANCE DEVELOPMENT



GPU accelerated computers: ≈10^3 powerful nodes

Many-Core computers: ≈10^4-10^5 simple nodes

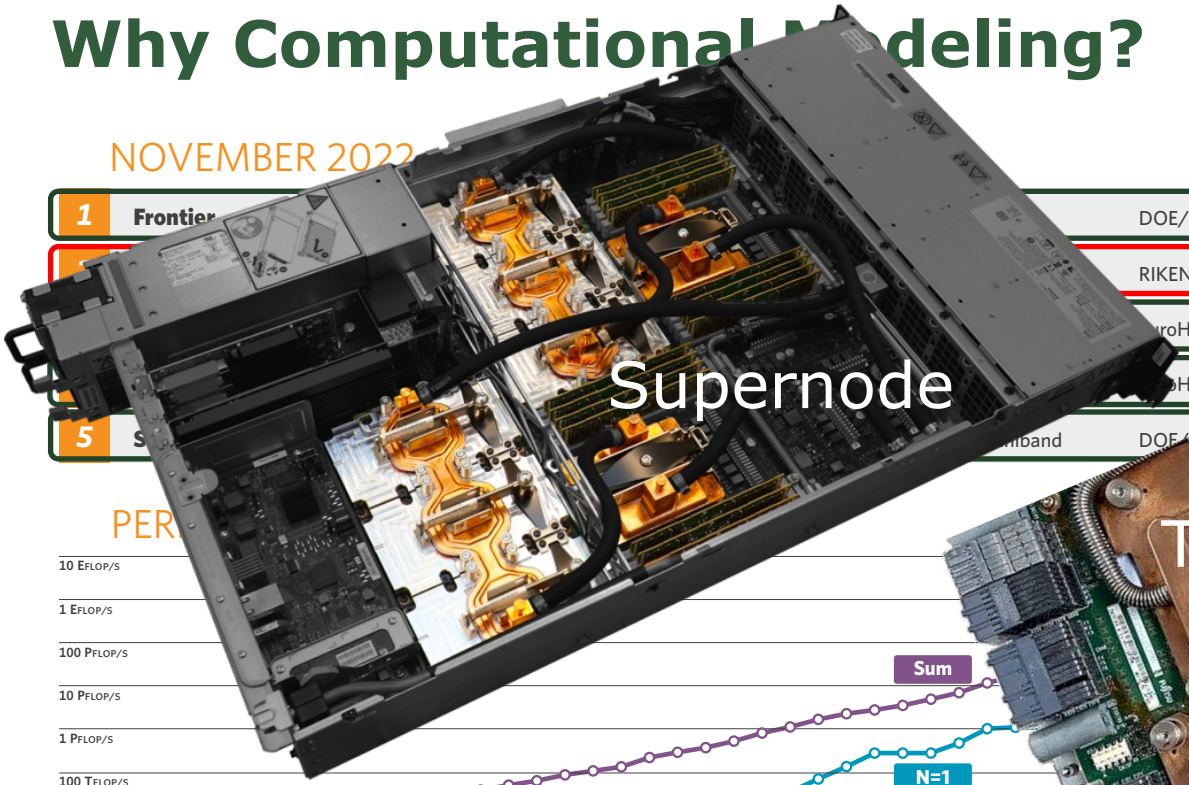
Why Computational Modeling?

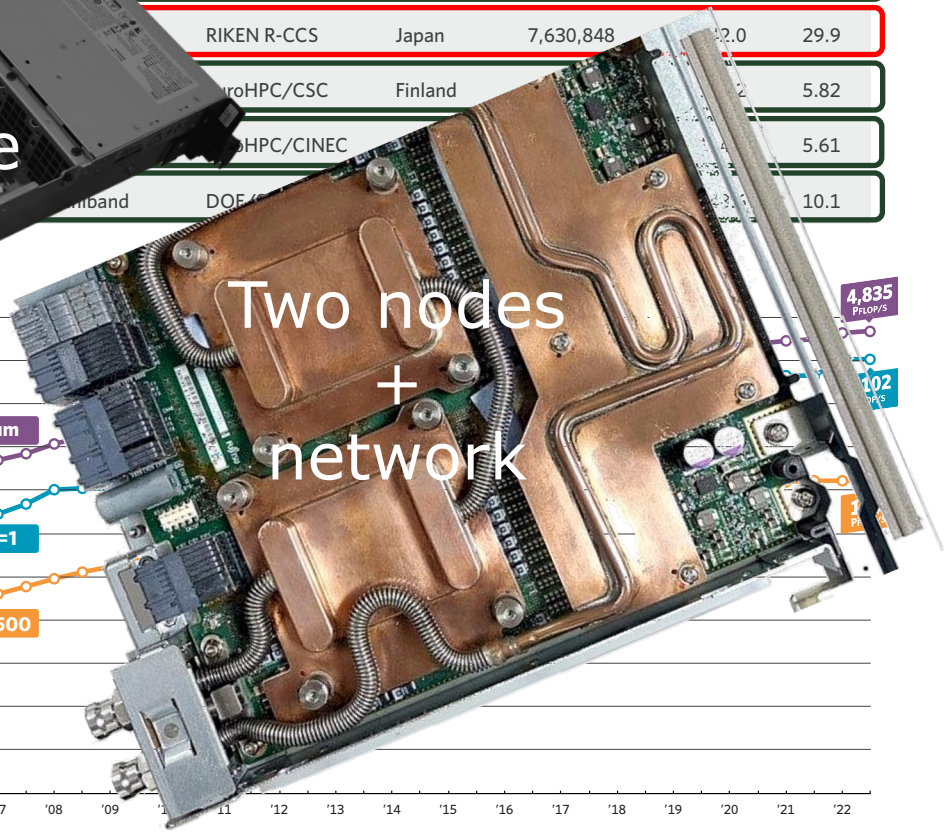
TOP500 The List. FIND OUT MORE AT top500.org

NOVEMBER 2022

		SITE	COUNTRY	CORES	RMAX PFLOP/S	POWER MW
1	Frontier	DOE/SC/ORNL	USA	8,730,112	1,102.0	21.1
		RIKEN R-CCS	Japan	7,630,848	2.0	29.9
		roHPC/CSC	Finland			5.82
		HPC/CINEC				5.61
5	S		nband	DOE/		10.1

PER





GPU accelerated computers: ≈10^3 powerful nodes

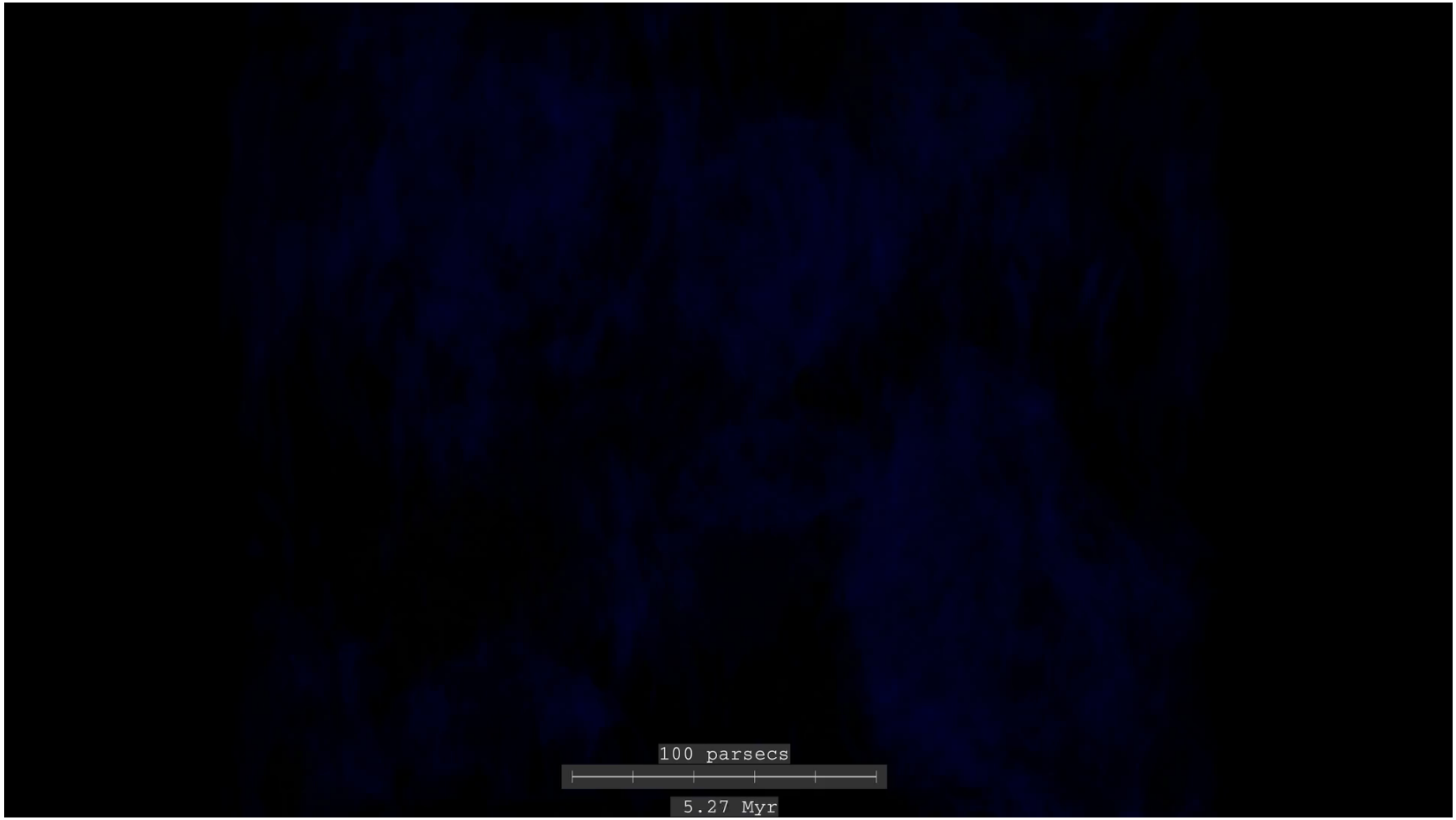
Many-Core computers: ≈10^4-10^5 simple nodes

Why Computational Modeling?

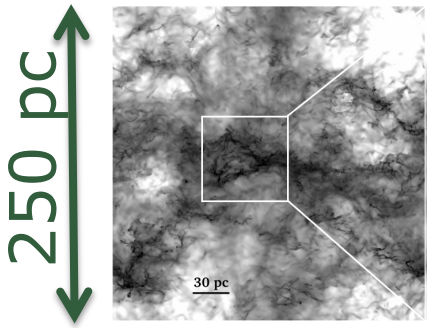
From this follows that

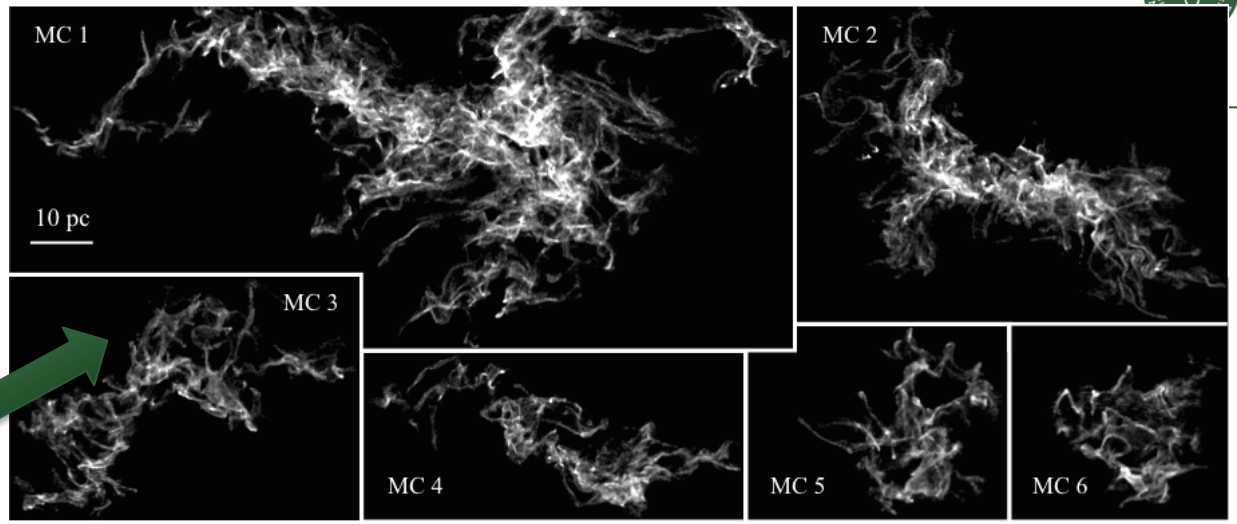
- Truly ground-braking research takes place at supercomputer centers
  - Access to such facilities are as competitive as access to ALMA, JWST, …
  - Hence, they act as filters, engaging the very best scientists
  - The field of **Star and Planet Formation is a case in point**
    - Tremendous progress (in understanding!) due to numerical simulations
    - In combination with observations, which need increasingly complex models
- It's crucially important to stay at the **technical forefront**
  - Computing capacity keeps following Moore's law (x2 every 18 months)
  - But the technology changes repeatedly
    - Currently towards many more, instead of much faster, processors
    - A strong trend is also the introduction of accelerators (GPUs)
- …but in this course we will
  - Not have emphasis on High Performance (Computing) tools → see HPPC in B3
  - Not learn how to program…though you may pick up good practices
  - **Focus on numerical methods using Python to *explore* results**

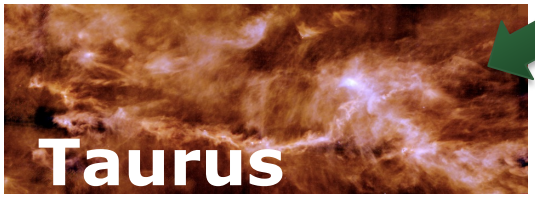
Example: the Galactic disk with supernova feedback

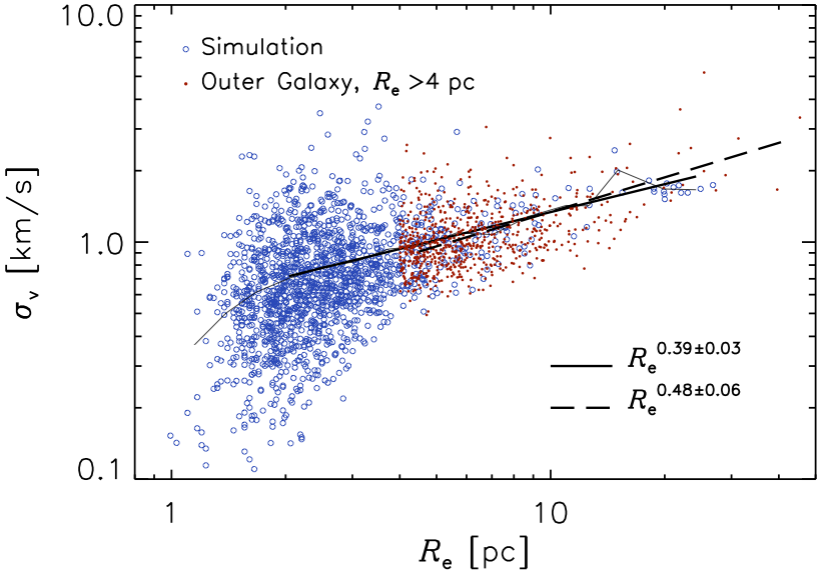


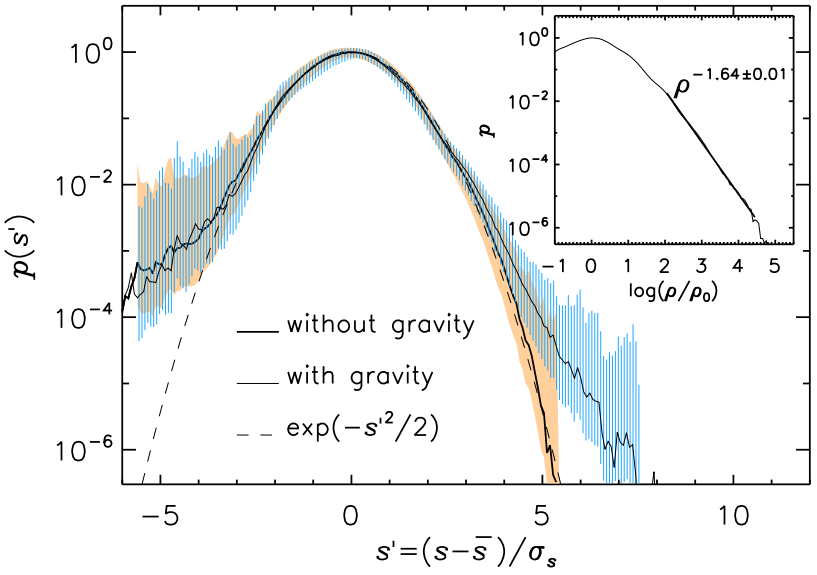
Supernovae & Turbulence in the ISM



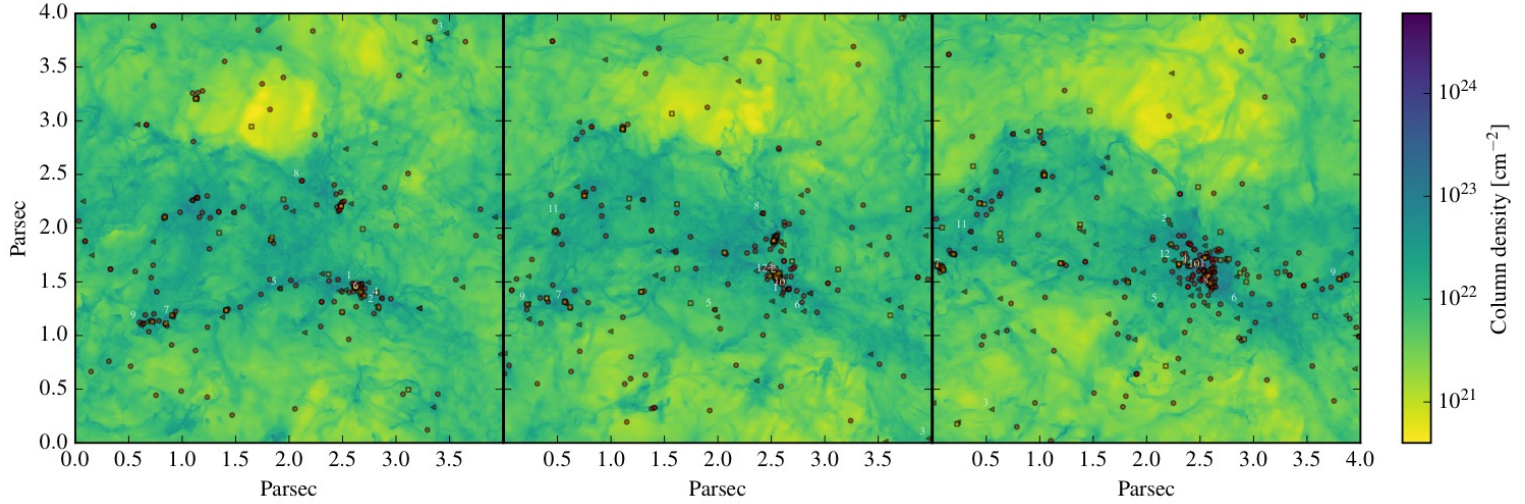






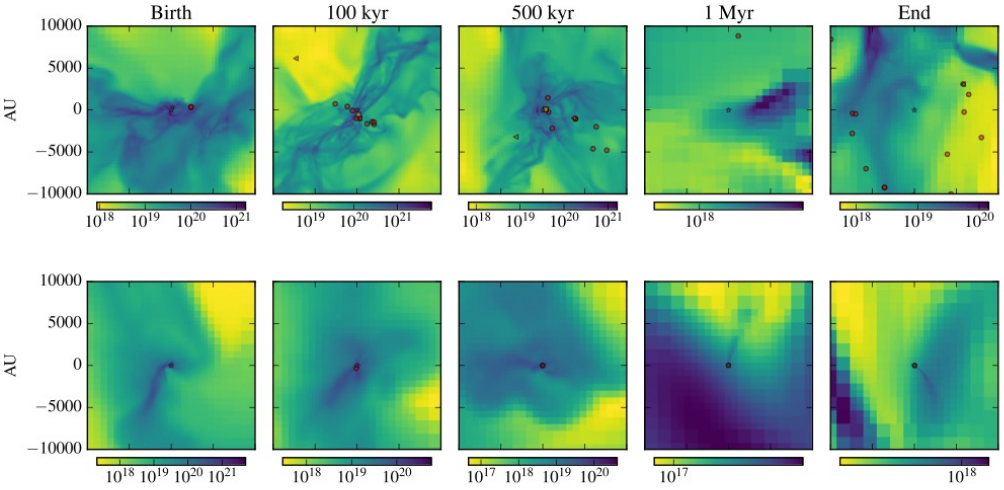


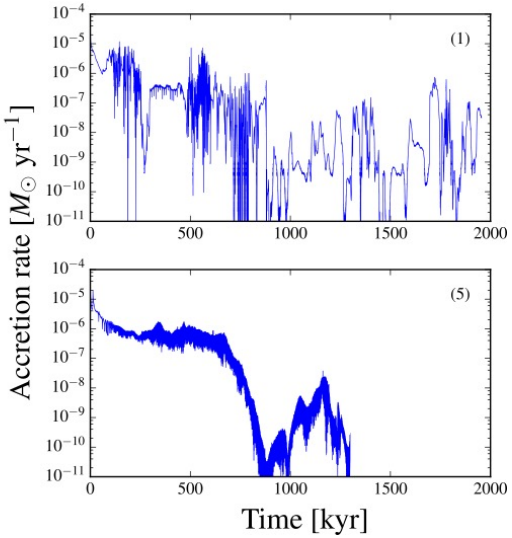
Smaller scales: Low Mass Star Formation



4 pc GMC fragment

3000 Msun, more than 500 stars



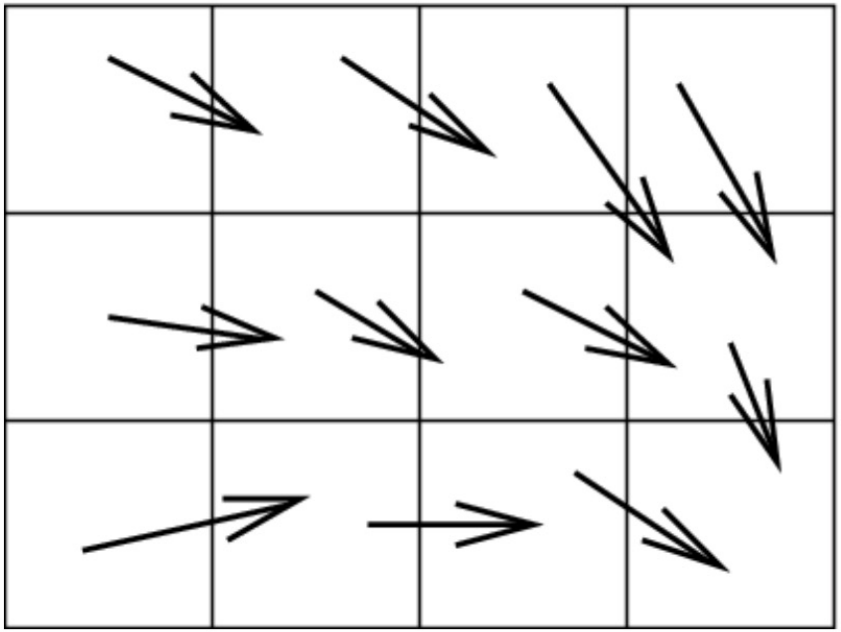


Realistic:

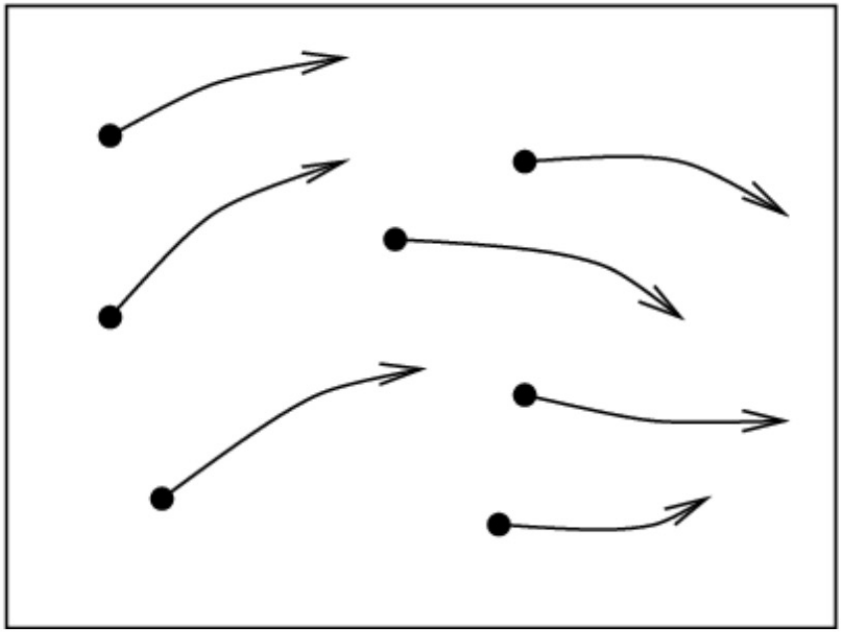
- accretion histories,
- luminosity distribution
- initial mass function

Mechanism needed for modeling astrophysics

How to make a numerical model of reality?

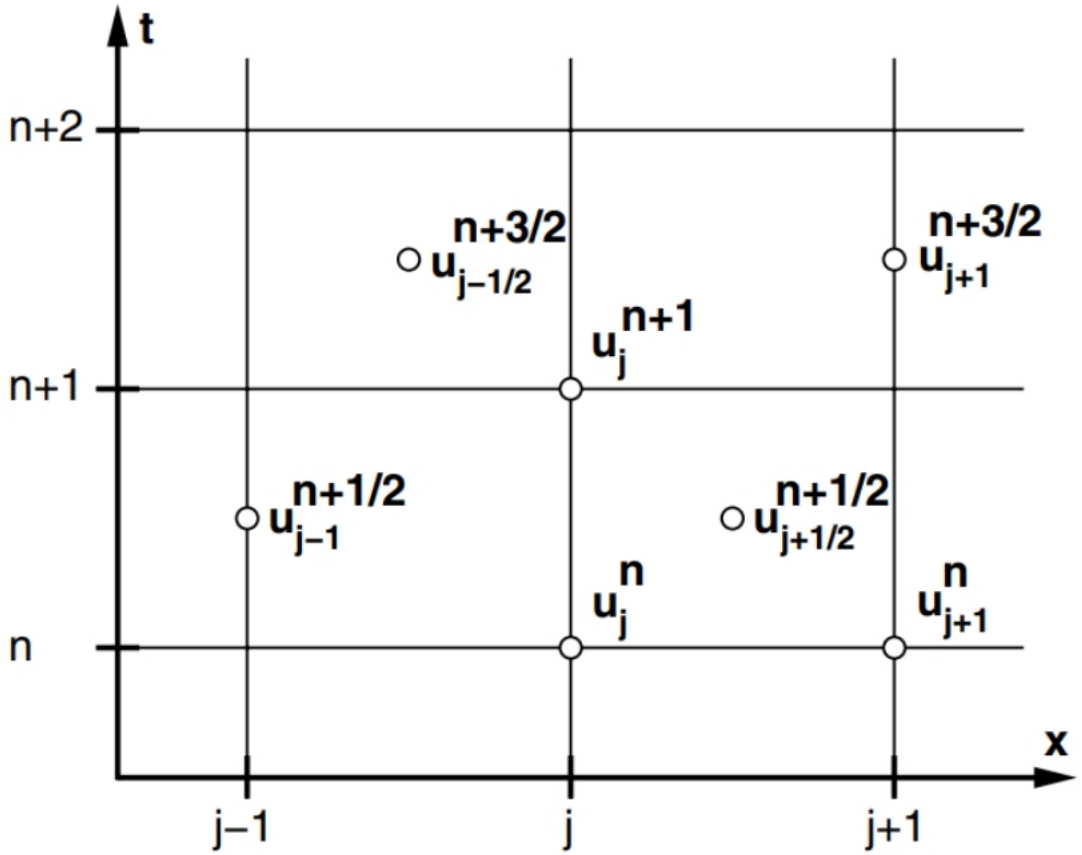


Eularian regular mesh



Lagranian unstructured mesh

Placing variables and good notation



[see chap 2.3]

Tools and Exercises with Assignments

Tools you can use (see Absalon for details)

1. **JupyterLab at ERDA**, to
   - ❑ develop a good understanding of astrophysical modeling and numerical methods
   - ❑ …and as an added bonus you may develop best practices in Python

2. **JupyterLab in Anaconda** on your laptop, to
   - ❑ avoid having to rely on a network connection
   - ❑ have a better graphics interface

3. **Spyder development environment in Anaconda**, to
   - ❑ have access to a good debugger, with a variable browser
   - ❑ have direct, interactive Python syntax control

Exercises with Assignments

Much of what would traditionally be lectures notes and view graphs will instead be **Jupyter Notebooks**, with

- A mix of text and executable Python
  - The text often contains equations (formatted with LaTeX commands)
  - The active Python cells illustrate methods, in preparation for …
- A mix of learning-by-example Python code, and turn-in-exercises
  - After working with the examples, and with the book as reference …
  - … you write your own code, to solve some problem / answer some question
- A corresponding Absalon assignment page, where
  - You answer the exercise questions by…
  - .. uploading a PDF report and/or the notebook with answers inline together with a PDF generated from the notebook

A note about "kernels"

When working with Jupyter Notebooks you may see the term ***kernel*** used.

A "kernel"

- is the Python process that *evaluates the cells* with executable code
- *remembers variable values* from previous cell executions
- only cares about *order of execution*, not order of cells in the text

Therefore, if you "go back up" to some previous cell, it might give a different result than last time (happens if some variable it depends upon now has a different value)

If results are strange try reseting your notebook and re-evaluate it

Time-outs and restart

When you close your laptop, your Jupyter server at ERDA keeps running for a number of hours (days?). When you open up (or before you close), you should

- Type CTRL-s (or hit the "save" icon) in the notebook
- close the JupyterLab tab in the web browser – you won't lose work

If you reconnect to ERDA within a few hours, you will see

.. and you should choose "My server" to reconnect. After more than a few hours, you will see

The difference with a new server is that the you will need to execute the cells in the notebook to get the values you had back into the kernel. As long as you "save" the notebook, only CPU time is lost.

This Week's Exercises

This week: Python primer + Advection exercises

- 1a Introduction to Python and Erda (20 pt)
  - hopefully you already have some knowledge of Python
- 1b Central Difference Derivatives (20p)
  - order and accuracy
- 1c Numerical Advection (20p)
  - direct implementation of the equations
  - order and phase errors
- 1d VanLeer's Advection method (40p)
  - "upstream" methods
  - "monotonic" property
  - face-centered "flux" of properties
- 1e von Neumann Analysis (20p)
  - Stability in time integration

Main theme is "Advection"

Quoting now directly from the JupyterLab Notebok:

Background

Advection is a central phenomenon in all kinds of fluid dynamics. To quote Wikipedia: "*In the field of physics, engineering, and earth sciences, advection is the transport of a substance by bulk motion*".

Any property of a gas or fluid, such as color, temperature, density, or even velocity or momentum, may be "advected" by the flow.

One way to think of and understand advection, and its central importance in fluid dynamics, is to consider the difference between studying the behavior of some system in a coordinate system moving with the flow (a *co-moving* coordinate system), and a stationary one (a *lab-frame* coordinate system).

Since we are looking at the same phenomenon, the terms that describe the physics (forces, heating / cooling, etc) are the same, and the only difference is apparent, and is due to the motion of the coordinate system. If what happens is described by $f(\mathbf{r}, t)$, then in a system moving with velocity $\mathbf{v}$, the same thing is described by $f(\mathbf{r} - \mathbf{v}t, t)$.

From the derivative chain-rule, the motion gives rise to a difference in partial time derivate $\partial f/\partial t$, which is

$$-v_x \partial f/\partial x - v_y \partial f/\partial y - v_z \partial f/\partial z = -\mathbf{v} \cdot \nabla f \qquad (1)$$

Main theme is "Advection"

Quoting now directly from the JupyterLab Notebok:

Background

Advection is a central phenomenon in all kinds of fluid dynamics. To quote Wikipedia: "*In the field of physics, engineering, and earth sciences, advection is the transport of a substance by bulk motion*".

Any property of a gas or fluid, such as color, temperature, density, or even velocity or momentum, may be "advected" by the flow.

One way to think of and understand advection, and its central importance in fluid dynamics, is to consider the difference between studying the behavior of some system in a coordinate system moving with the flow (a *co-moving* coordinate system), and a stationary one (a *lab-frame* coordinate system).

Since we are looking at the same phenomenon, the terms that describe the physics (forces, heating / cooling, etc) are the same, and the only difference is apparent, and is due to the motion of the coordinate system. If what happens is described by $f(\mathbf{r}, t)$, then in a system moving with velocity $\mathbf{v}$, the same thing is described by $f(\mathbf{r} - \mathbf{v}t, t)$.

From the derivative chain-rule, the motion gives rise to a difference in partial time derivate $\partial f/\partial t$, which is

$$-v_x \partial f/\partial x - v_y \partial f/\partial y - v_z \partial f/\partial z = -\mathbf{v} \cdot \nabla f \qquad (1)$$

Main theme is "Advection"

Quoting now directly from the JupyterLab Notebok:

Background

Advection is a central phenomenon in all kinds of fluid dynamics. To quote Wikipedia: "*In the field of physics, engineering, and earth sciences, advection is the transport of a substance by bulk motion*".

Any property of a gas or fluid, such as color, temperature, density, or even velocity or momentum, may be "advected" by the flow.

One way to think of and understand advection, and its central importance in fluid dynamics, is to consider the difference between studying the behavior of some system in a coordinate system moving with the flow (a *co-moving* coordinate system), and a stationary one (a *lab-frame* coordinate system).

Since we are looking at the same phenomenon, the terms that describe the physics (forces, heating / cooling, etc) are the same, and the only difference is apparent, and is due to the motion of the coordinate system. If what happens is described by $f(\mathbf{r}, t)$, then in a system moving with velocity $\mathbf{v}$, the same thing is described by $f(\mathbf{r} - \mathbf{v}t, t)$.

From the derivative chain-rule, the motion gives rise to a difference in partial time derivate $\partial f/\partial t$, which is

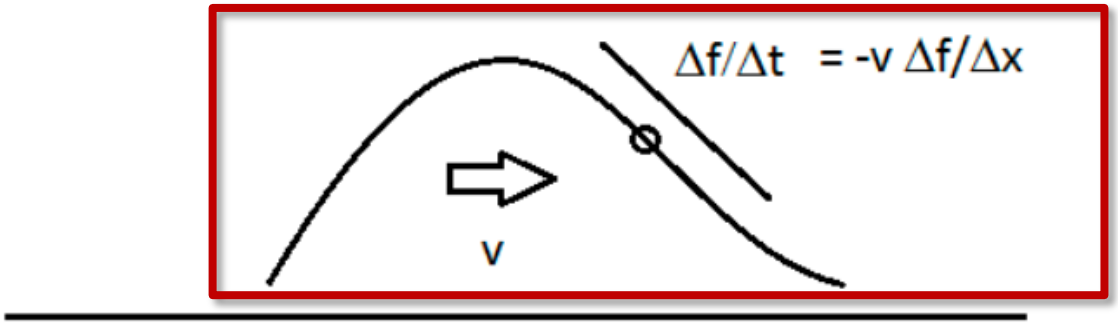
$$-v_x \partial f/\partial x - v_y \partial f/\partial y - v_z \partial f/\partial z = -\mathbf{v} \cdot \nabla f \qquad (1)$$

Main theme is "Advection"

From the derivative chain-rule, the motion gives rise to a difference in partial time derivate $\partial f/\partial t$, which is

$$-v_x \partial f/\partial x - v_y \partial f/\partial y - v_z \partial f/\partial z = -\mathbf{v} \cdot \nabla f \qquad (1)$$

If, for example, the shape of a function is stationary in the co-moving coordinate system, then in the lab-frame one sees the function shape passing by, with the local value changing slowly where the function is smooth, and rapidly where the funcion is steep.

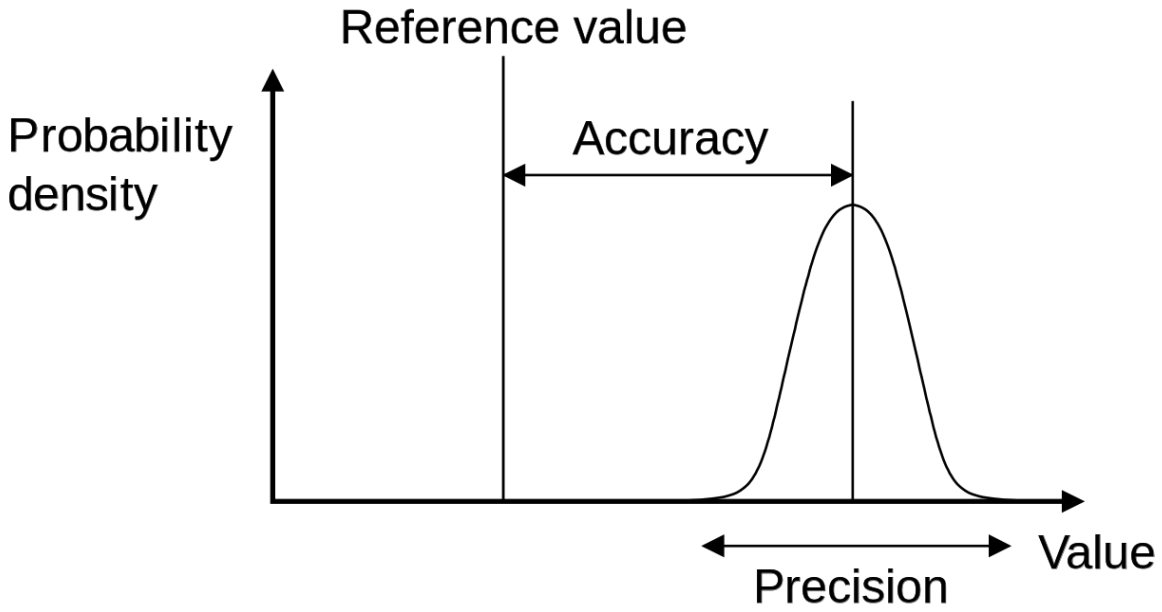


It is customary to write the partial time derivative in the lab-frame $\partial/\partial t$, while the one in the co-moving frame is written $\mathrm{D}/\mathrm{D}t$, so

$$\mathrm{D}f/\mathrm{D}t = \partial f/\partial t + \mathbf{v} \cdot \nabla f \qquad (2)$$

Precision and Accuracy in Numerical Methods

- A numerical model first and foremost has to be accurate (a reasonable representation of reality / the mathematical model)
- Precision in the model is also important, but secondary



Precision and Accuracy in Numerical Methods

- A numerical model first and foremost has to be accurate (a reasonable representation of reality / the mathematical model)
- Precision in the model is also important, but secondary
- We often talk about *numerical convergence*, and *convergence rate*

  → How fast the error decreases as a function of resolution (here as global norm measure or as a max norm)

$$Error = \frac{1}{N}\sum_{i=1}^{N}|f(x_i) - f_{solution}(x_i)|\,, \qquad Error = \max_{i=1-N}(|f(x_i) - f_{solution}(x_i)|$$

- Convergence rate is measured by how the error changes as the size of the mesh cells decrease
- A method is *stable* if the correct answer is recovered in the limit of infinite resolution (e.g. infinitesimal small mesh cells)