



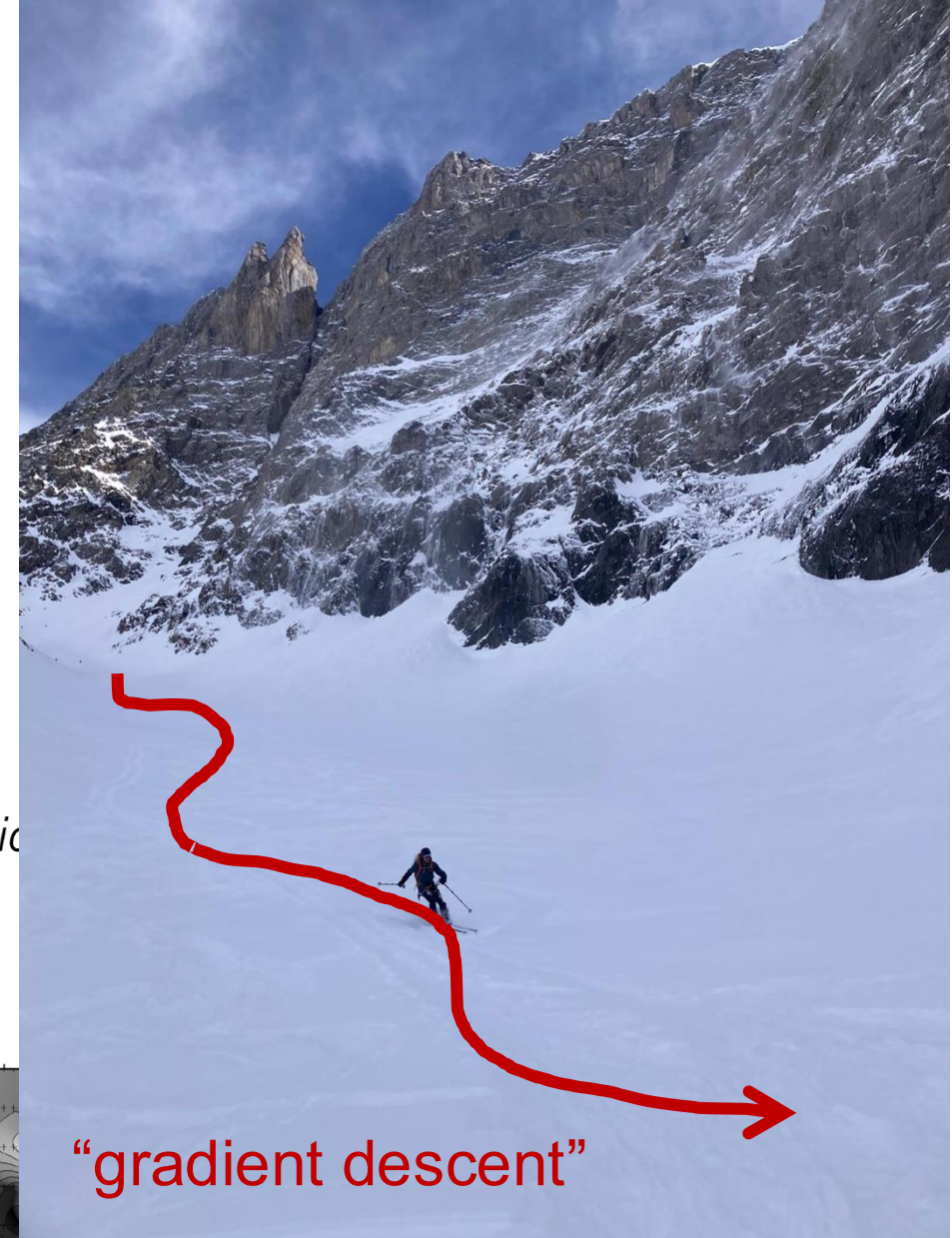
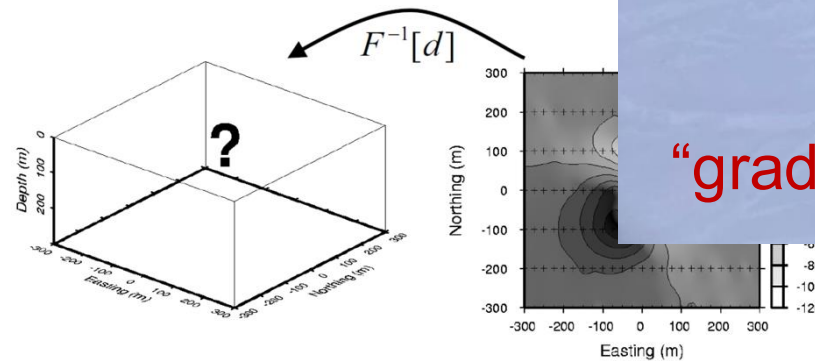
Building Sustainable Scientific Software:

Lessons learned from an industry-
academia collaboration

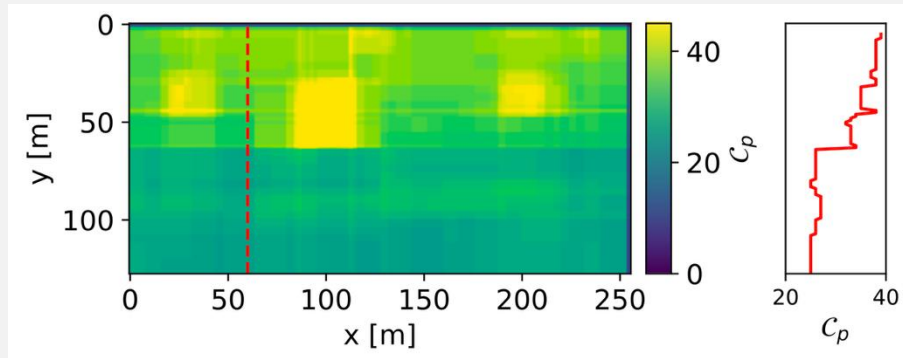
Arnaud Mercier
RSE Annual Day
31.08.2026



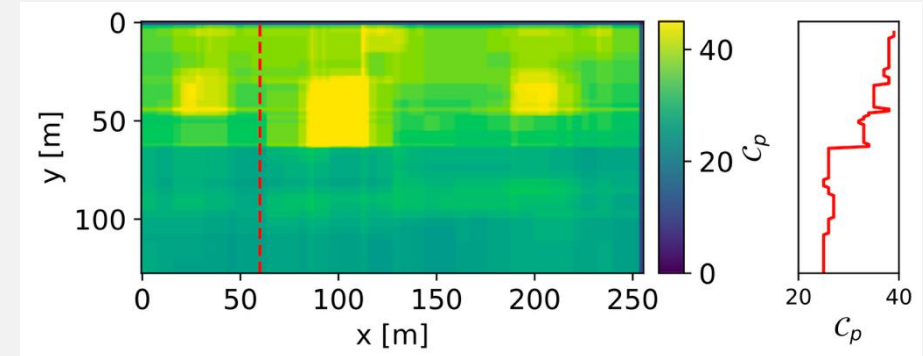
- Last year PhD student in Mathematical Geophysics **ETH** zürich
- Optimal experimental design for Full waveform inversion
- Differentiable physics



A scientific result is not a reusable workflow



$\neq$



Examples

Tests

Documented code

Inputs

Including the software development in the grant

Scientific objectives

Original scientific idea

Software work packages

Software requirements

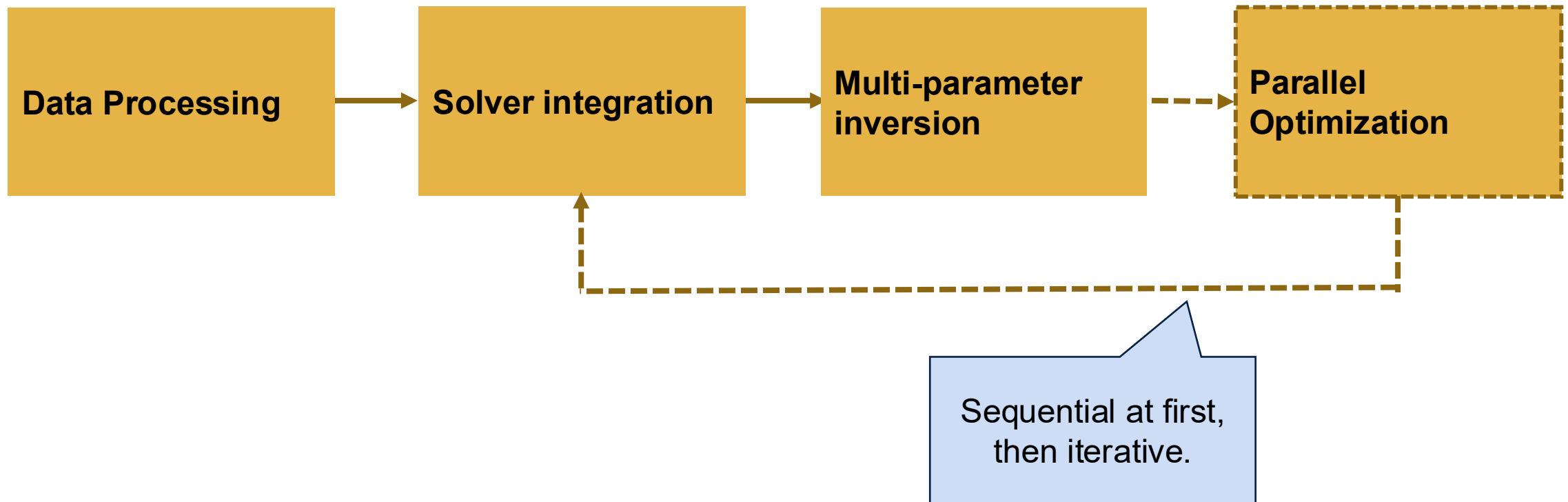


Durable research output

Reproducible and shareable workflows



Software work packages gave us a roadmap: A shared starting point, not a fixed specification



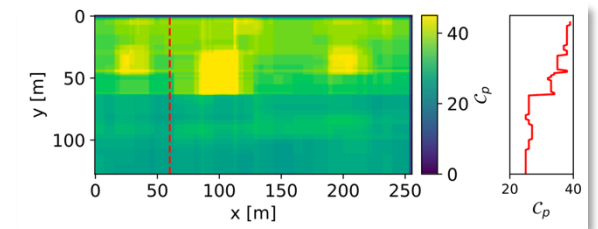
Leveraging software development tools

The repository became a collaboration space



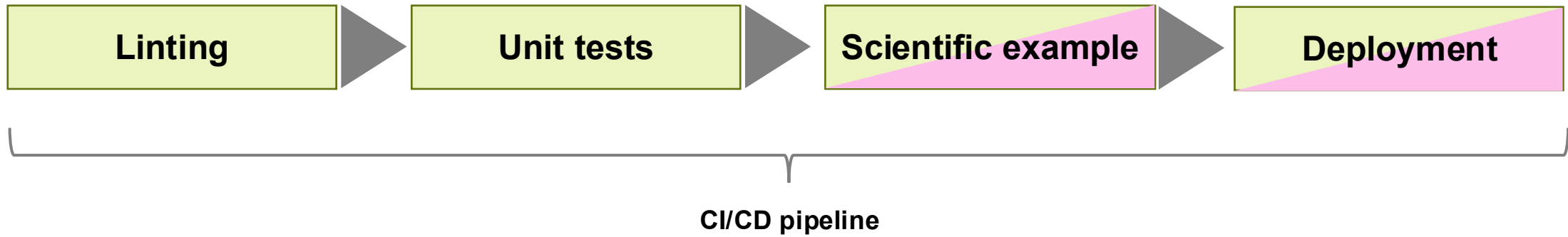
Key takeaways

- The repository gave the collaboration a shared, visible place;
- Code questions and scientific choices could be discussed in the same workflow.



	Academic environment (ETH)
	Software environment (Mondaic)

End-to-end testing as a gold standard



Key takeaways

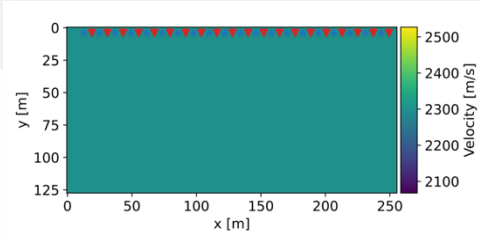
- Complete typical CI/CD pipeline from Mondaic
- Add scientific examples as notebook
- Deployment on GitLab pages as living documentation

	Academic environment (ETH)
	Software environment (Mondaic)

What we built during my PhD in collaboration with Mondaic AG

Inputs

- Prior knowledge/models
- Waveform data
- Inversion parameters
- Etc.



Elastic FWI Modular open-source Python framework

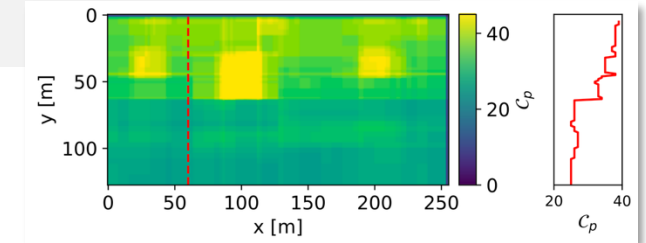
Simulation backends

Salvus

Other
solvers

Outputs

- Reusable workflows
- Simple examples & documentation
- Project hash



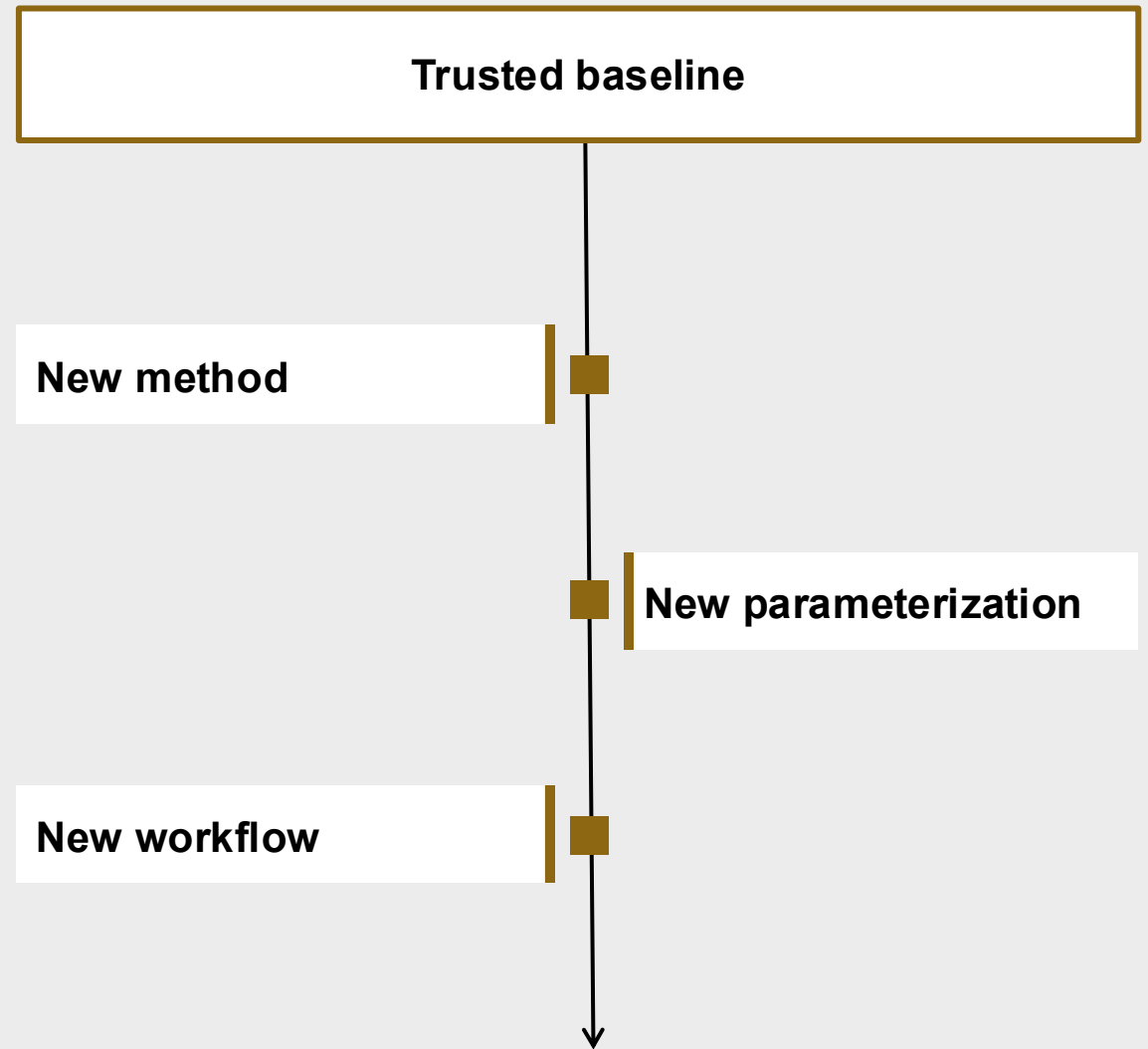
Benefits and drawbacks of the collaboration

Complementary expertise accelerated the project



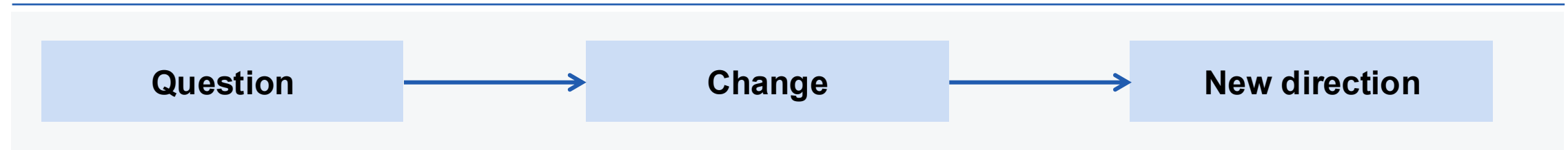
	Academic environment (ETH)
	Software environment (Mondaic)

Predictable software
made it safer to
explore new
directions.



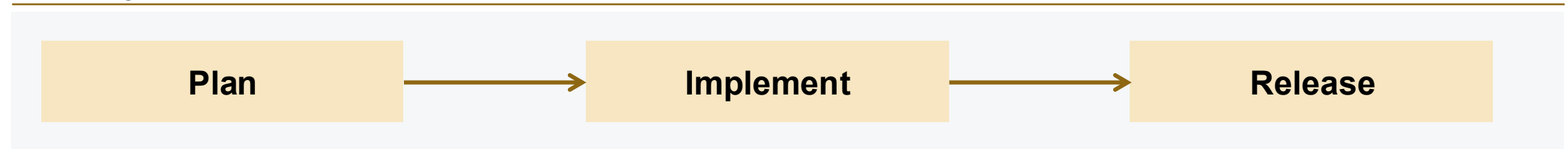
Agility of research vs. robustness of industry

Research



VS.

Industry



Key takeaways

- Different tempos need explicit synchronization points
- Proprietary core
- Clear responsibility
- Integration dependency

Personal lessons and conclusion

01

Make software integral
part of research planning

02

Start with a plan, iterate
to improve

03

Use integration tests as
scientific safeguards



The goal is research software that can **support and accelerate scientific research**