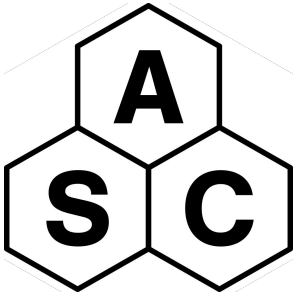
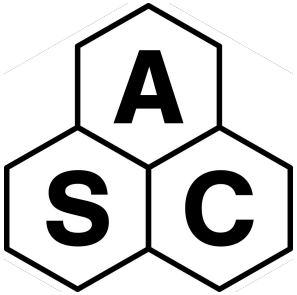


Teaching by Aalto Scientific Computing



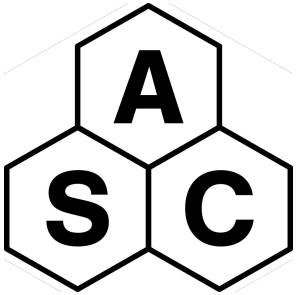
What we teach

Major yearly courses:

- Intro to Scientific Computing / HPC kickstart
- Python for Scientific Computing

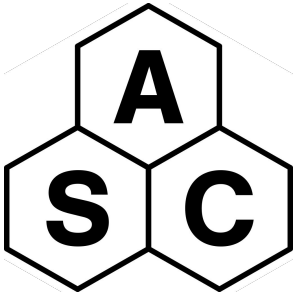
Special courses:

- Linux Shell Scripting
- Tuesday tools and techniques for HPC / Workflows course
- Deep learning in practice (on planning stage)



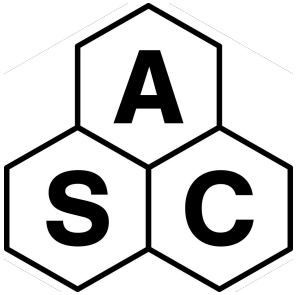
How we teach

- Minimal barrier of entry: anybody can join
- Everything done openly in the internet
- Collaboration with CodeRefinery and other friends
- Streaming via Twitch
- Use community notes for lively discussion during the course



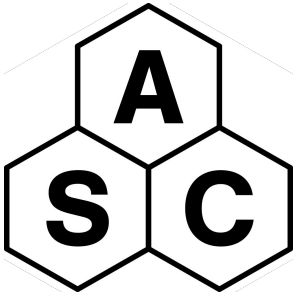
How we teach

- Emphasis on teaching the useful way instead of the perfect way
  - We want learners to feel like they can do stuff
- Courses should be fun and relatable
  - Stories, icebreakers etc. keeps the audience engaged, which gives the course more interactivity
- Co-teaching is very important
  - It reduces listening fatigue
  - Teaching flows more naturally when it is designed around discussion



Key learnings from our courses

- Getting into the learners' perspective is very important
  - Big picture explanation on **why** we're doing what we're doing can make them more motivated
- Every course becomes a basics course
  - Learners can reach complex topics as long as they can follow the talk
  - For advanced users we try to keep them motivated by going through reasons **why** the things they do are best practices



Key learnings from our courses

- People do not come to courses that are labeled “advanced”
  - Courses sell on solutions
  - First iteration of a course is always hard, but doing multiple iterations simplifies the material
  - If first iteration goes badly, teacher might not do a second run
- Solving technical problems does not usually go well in a course setting
  - We try to do tech setup sessions before courses and solve problems in our daily garage