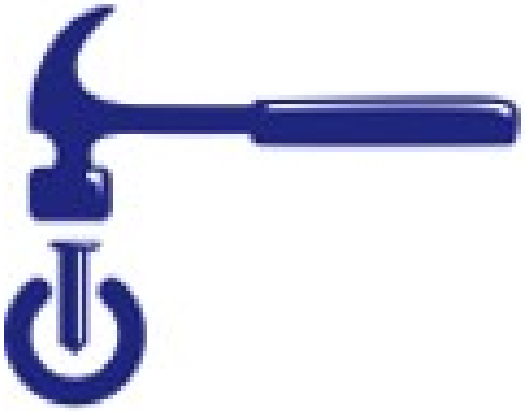
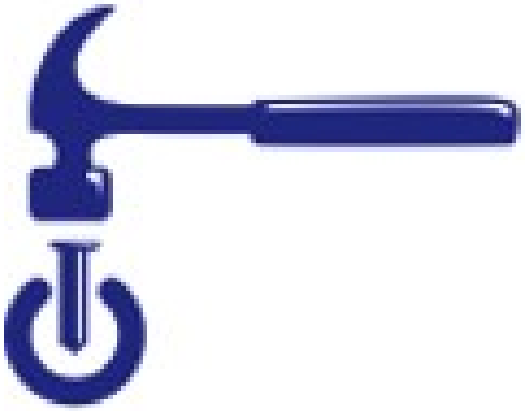


# Welcome to Software Carpentry!
# January 13-16, 2015


Instructors:

Matt Gidden
Steve Goldstein
Lauren Michael
Danielle Nielsen
Cliff Rodgers
Paul Wilson


ACI
Advanced Computing Infrastructure



# If You Can't Reproduce It, Is It Still Science?

## And how long will it take?

Paul Wilson

Inspired by Greg Wilson
Software Carpentry

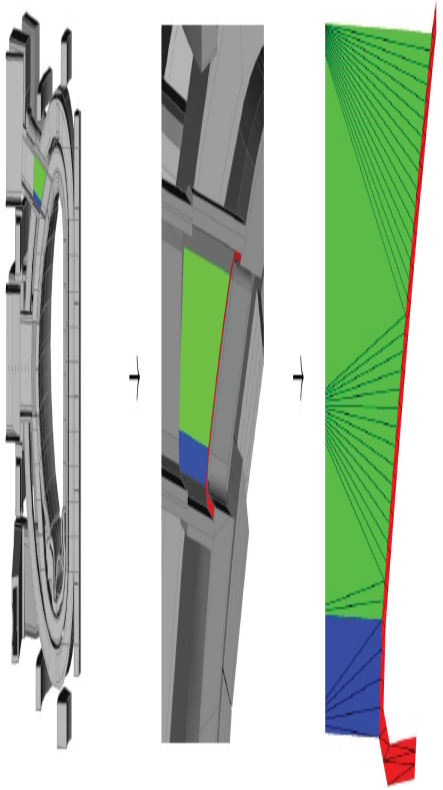
# Reality of Research Computing

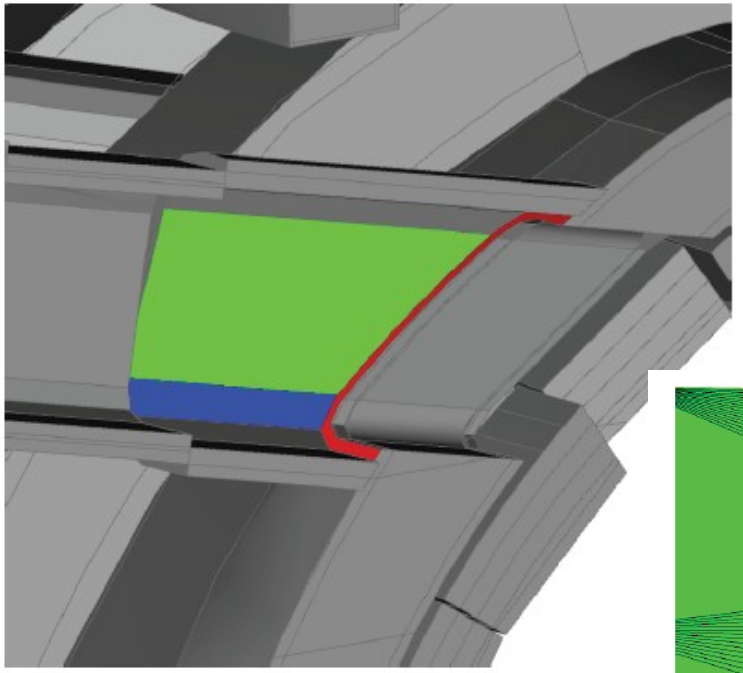
- Many scientists spend most of their time developing, maintaining, or running software
  - Most don’t consider themselves software engineers
  - Few have ever been taught how
    - Learned on-the-job
    - Tribal knowledge

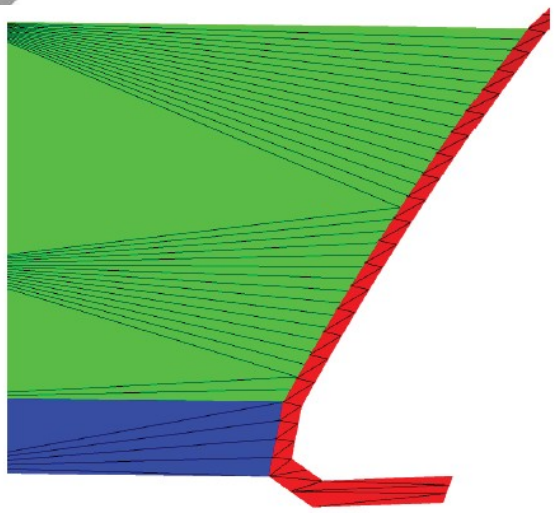
# So What…

- Most results take longer to produce than they need to
  - Not because of a lack of computers
- Difficult to assess quality
  - Often measured by reproducibility
  - “System” doesn’t care

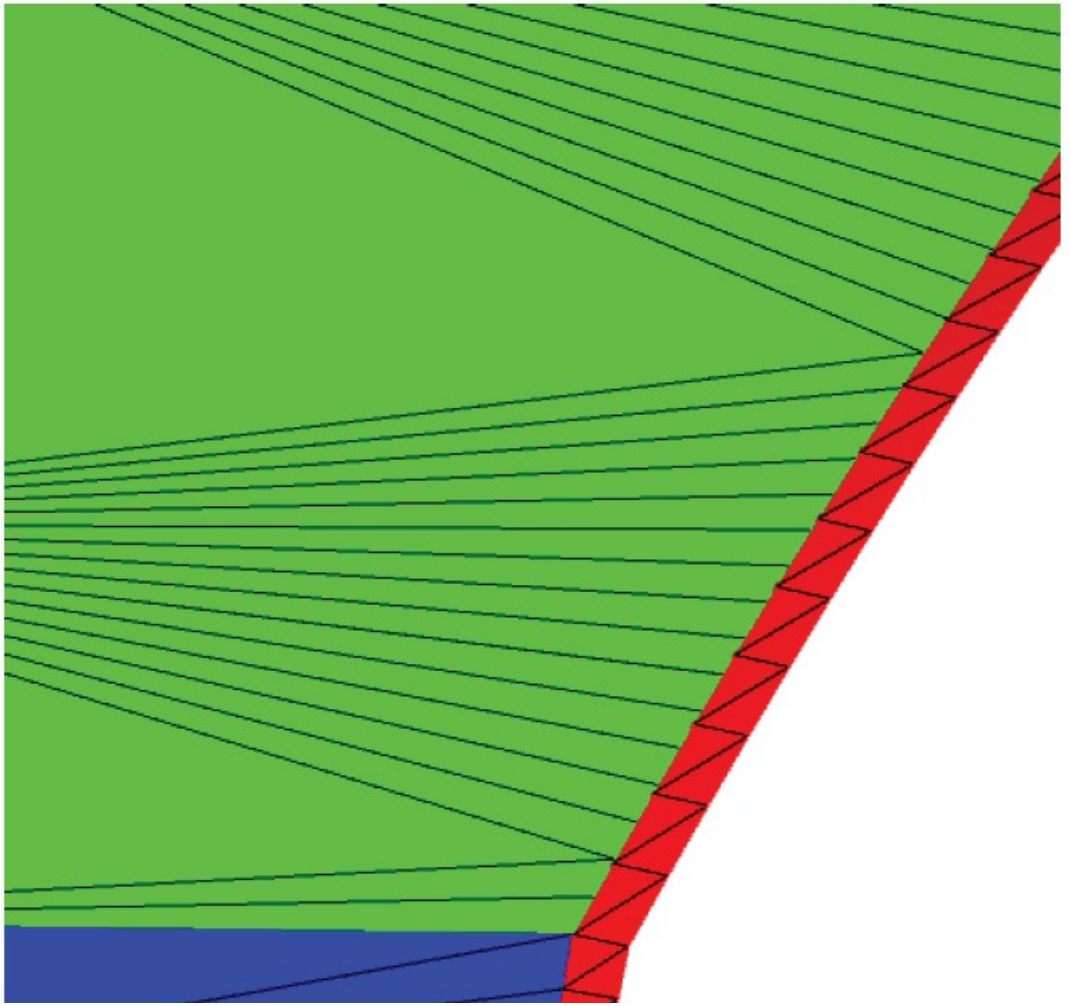
# A Recent Story: Sealing a Faceted Geometry





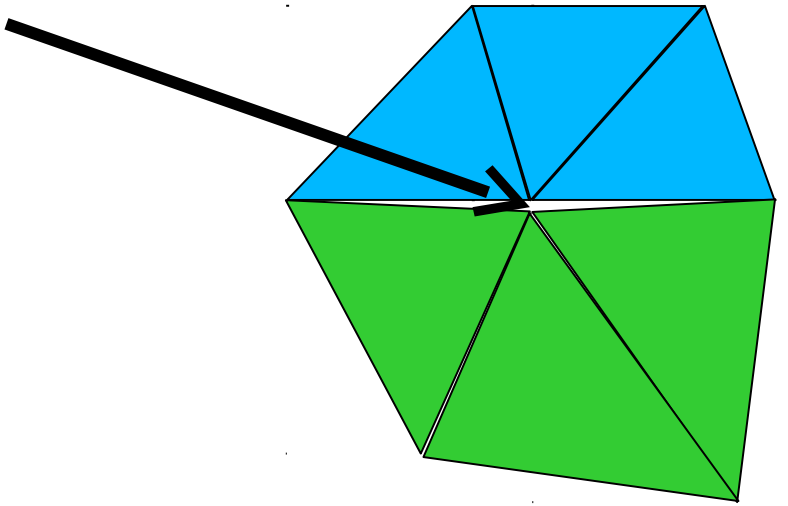


# A Recent Story: Sealing a Faceted Geometry



# A Recent Story: Sealing a Faceted Geometry

- Lost particles through “leaks”



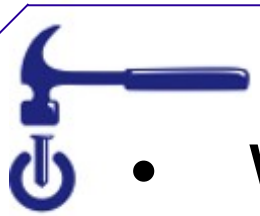
- Reduce confidence in solution

# A Recent Story: Sealing a Faceted Geometry

| Model | Particles Simulated [millions] | Lost Particles | |
|---|---|---|---|
| | | Original | Robust |
| UW Nuclear Reactor | 41 | 5649 ± 178 | 0 |
| Advanced Test Reactor | 74 | 141 ± 32 | 0 |
| 40° ITER Benchmark | 225 | 67 ± 39 | 0 |
| ITER TBM | 205 | 665 ± 184 | 0 |
| ITER Module 4 | 59 | 59 ± 19 | 0 |
| ITER Module 13 | 79 | 450 ± 60 | 0 |
| FNG Benchmark | 1310 | 31273 ± 989 | 0 |
| ARIES First Wall | 4070 | 25 ± 18 | 0 |
| HAPL IFE | 286 | 65 ± 19 | 0 |
| Z-Pinch Fusion | 409 | 2454 ± 317 | 0 |

# Software Carpentry to the Rescue

- Best practices used by the best software engineers whose business is development of quality software
  - They don’t always have formal training
  - They don’t always follow all the practices
  - Growing evidence supported by empirical studies



- Write software for people, not computers
- Automate repetitive tasks
- Use the Computer to Record History
- Make Incremental Changes
- Use Version Control
- Don’t Repeat Yourself
- Plan for Mistakes
- First make it correct, then make it fast
- Document Design & Purpose
- Conduct Code Reviews

# Two Days ≠ Ten Practices

- Automate repetitive tasks
- Write software for people, not computers
- Don’t Repeat Yourself (or Others)
- Make Incremental Changes/Use Version Control
- Plan for Mistakes
- Conduct Code Reviews

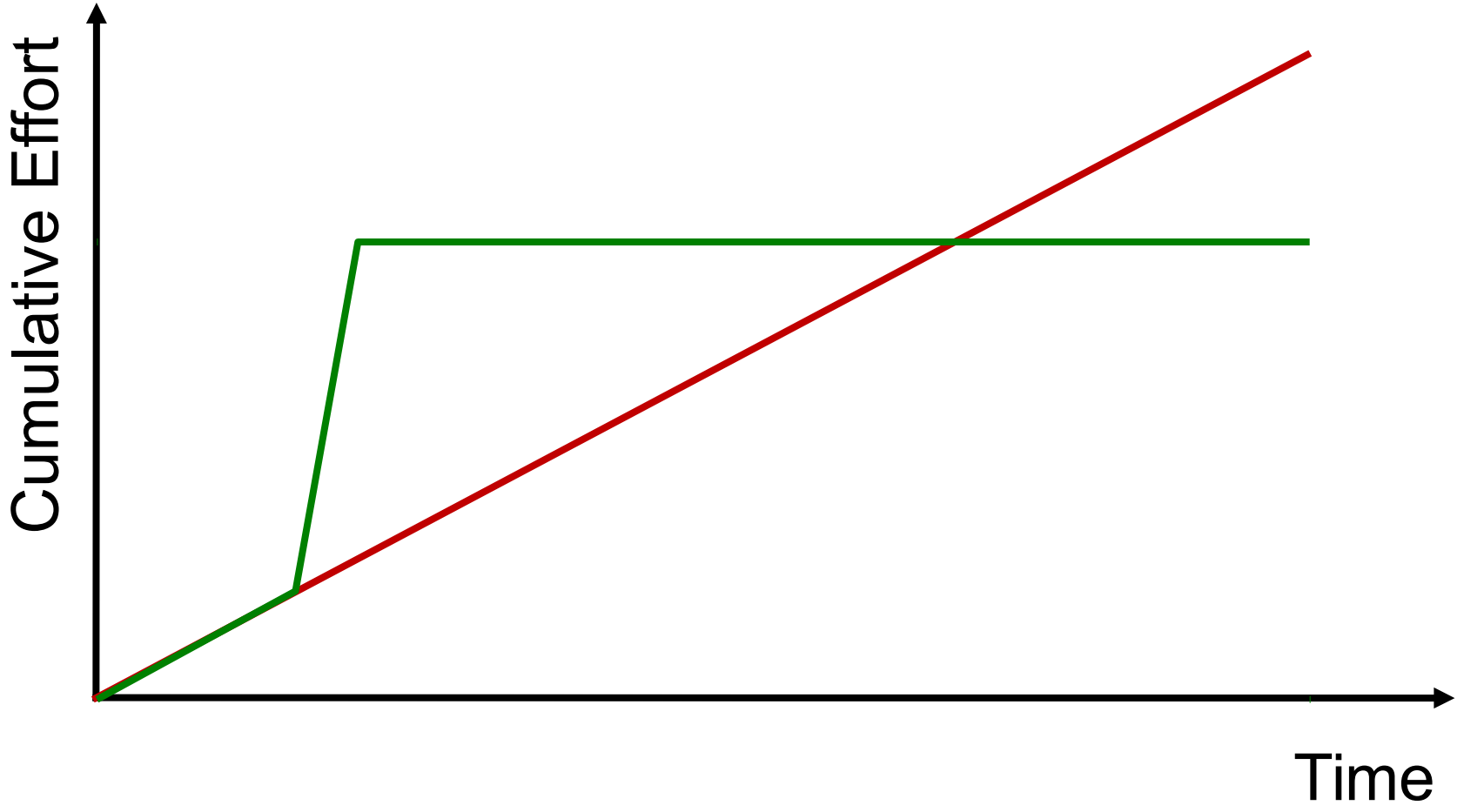
# Make Incremental Changes Redux

- This applies to HOW you work
- Choose one practice
  - Implement it in your work
  - Share it with your lab group
  - Allow it to sink in
- Repeat

# How to Choose Where to Start?

- It will depend on the nature of your work
- Consider the purpose:
  - Improve productivity
  - Improve quality

# Thoughts on Productivity and Automation



# Thoughts on Productivity and Automation

HOW LONG CAN YOU WORK ON MAKING A ROUTINE TASK MORE EFFICIENT BEFORE YOU'RE SPENDING MORE TIME THAN YOU SAVE?
(ACROSS FIVE YEARS)