# Introductory Remarks

## Best Practices Workshop, March 25-26 2019, Hampton VA

Carlos Cruz
Jules Kouatchou
Brent Smith

NASA GSFC Code 606/610 (ASTG/GMAO)
Greenbelt, Maryland 20771

# Who are we?

- Carlos Cruz (Computational Scientist - ASTG)
- Jules Kouatchou (Computational Scientist - ASTG)
- Brent Smith (Computational Scientist- GMAO)


# Objectives of this tutorial

**To provide an overview of useful software engineering practices to HPC code developers**


# Best Practices for Scientific Computing

OPEN ACCESS Freely available online

PLOS | BIOLOGY

Community Page

## Best Practices for Scientific Computing

**Greg Wilson[1]*, D. A. Aruliah[2], C. Titus Brown[3], Neil P. Chue Hong[4], Matt Davis[5], Richard T. Guy[6¤], Steven H. D. Haddock[7], Kathryn D. Huff[8], Ian M. Mitchell[9], Mark D. Plumbley[10], Ben Waugh[11], Ethan P. White[12], Paul Wilson[13]**

**1** Mozilla Foundation, Toronto, Ontario, Canada, **2** University of Ontario Institute of Technology, Oshawa, Ontario, Canada, **3** Michigan State University, East Lansing, Michigan, United States of America, **4** Software Sustainability Institute, Edinburgh, United Kingdom, **5** Space Telescope Science Institute, Baltimore, Maryland, United States of America, **6** University of Toronto, Toronto, Ontario, Canada, **7** Monterey Bay Aquarium Research Institute, Moss Landing, California, United States of America, **8** University of California Berkeley, Berkeley, California, United States of America, **9** University of British Columbia, Vancouver, British Columbia, Canada, **10** Queen Mary University of London, London, United Kingdom, **11** University College London, London, United Kingdom, **12** Utah State University, Logan, Utah, United States of America, **13** University of Wisconsin, Madison, Wisconsin, United States of America

NASA

SSAI

# Best Practices for Scientific Computing

1. Write programs for people, not computers Coding standards
2. Let the computers do the work make, cmake
3. Make incremental changes Git, testing
4. Don't repeat yourself (or others) Coding standards
5. Plan for mistakes Testing
6. Optimize software only after it works correctly Use profilers
7. Document design and purpose, not mechanics Documentation
8. Collaborate Agile methodology, GitHub


# Agenda

**Day 1**

- Introductory Remarks
- Version Control
- Documentation
- Coding Standards

**Day 2**

- Agile Development
- Unit Testing and TDD
- Continuous Integration
- Regression Testing
- Containers


# Get Lecture Materials from Github