Debugging

It Doesn't Work!

- How to fix it:
  - Develop a hypothesis
  - Test the hypothesis
  - If correct, fix bug. If incorrect,
    - Develop new hypothesis with the additional information
    - Repeat until code works
- Making random changes is rarely much help. (*Shotgun debugging*.) Try to understand what could be going wrong.

Common Errors

- Compiled languages are particularly prone to *segmentation faults*
  - This occurs when you go outside your memory allotment
- Most popular ways to do it in Fortran:
  - Array bounds errors (you try to access an index not in your range)
  - Number or type mismatch in parameter lists
    - Use interfaces/modules!
  - In older code: errors in `common` declarations

Heisenbugs

- A bug that alters its behavior or, worse, disappears when you try to observe it.
- Typically due to memory stomping that may or may not result in a segfault (at least with a segfault you know there's a problem).
- Uninitialized variables can cause Heisenbugs, mainly in C/C++
  - Beware: Fortran compilers often set uninitialized variables to zero, which may result in nonsense but not a crash. Usually a compiler flag will change this behavior.

Interpreters (Python Etc.)

- Since your code isn't an executable you can hope that you won't segfault, merely fail in some less spectacular manner
  - A segfaulting interpreter would be quite bad
- But you can still get mysterious errors
- Python does array bounds checking at runtime if it can (i.e. you have declared it to be a fixed size in advance).

Old-Style Debugging

- Print, print, print, print.
- Narrow down the point of failure to a segment of code.
- Print some more.

Debuggers

- Free
  - gdb: Fortran/C/C++ (command line)
  - pdb: Python (built-in)
  - DDD: GUI for gdb and pydb (add-on Python debugger)—Linux mainly
- Proprietary
  - Totalview: C/C++/Fortran
    - GUI
    - Can debug MPI and OpenMP codes

Fortran/C/C++

- Compile for the debugger:
- ifort –g mycode.f90
- For Fortran also add bounds checking (performed by Fortran runtime library):
- ifort –g –CB mycode.f90
  - Other compilers: use –g -C

Using gdb

- Other command-line debuggers are similar

```
gdb mybinary
<stuff>
(gdb) run opt1 opt2
<crash>
(gdb) bt
```

Python Debugger

- Standard module is pdb

```
from pdb import *
```

- From 2.5 (newer than default on clusters!) and up, can step through code as for compiled-language debuggers
- Can run non-interactively

```
python –m pdb myscript.py
```

- http://docs.python.org/library/pdb.html

Spyder for Python

- Spyder gives you direct access to Python debugging.
- It provides a window analogous to the stack window of graphical debuggers like Totalview.