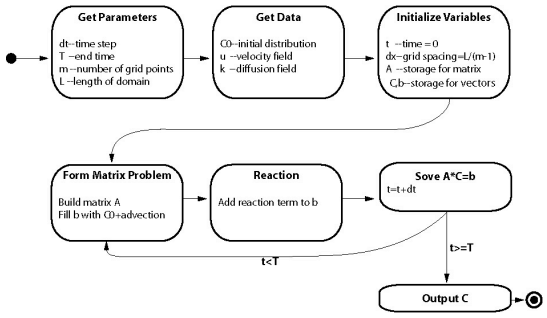
Outline

- Announcements:
  - HW I key online this afternoon
  - HW II due Friday
- Implementing model problem
- Validation

Implementing Model Problem



Implementing Model Problem

- Specification of command file
  1. m--(integer)--number of grid points
  2. L-- (real)--lengthof domain
  3. dt--(real)--time step
  4. T--(real)--max time
  5. kcase--(integer)--0=>case a, 1=>case b, 2=> case c
  6. real (case a) or file name (case b,c)
  7. Etc. for u, g, K

Implementing Model Problem

- Time-dependent input file:
  - N--(integer)--number of time samples
  - t1--(real)--time offirst obs.
  - V1:m-(real)--values at m points at time t1
  - t2--(real)--time offirst obs.
  - V1:m-(real)--values at m points at time t2
  - :
  - tN--(real)--time offirst obs.
  - V1:m-(real)--values at m points at time tN

Validation

- Assume your code will compile and run
- THIS DOESN"T MEAN YOU"RE DONE
- You need to verify that your code is solving the the right problem

Formal Specification

- $I \wedge P \Rightarrow O$
- This says what a program should do, but says nothing about how it will get done
  - No details of *P*
- We now have P (we think)
  - We can test that I^P=>O for some inputs and outputs

Test Cases

- Test cases are an important part of scientific computing
- Typically, these are simple problems for which the answer is known
  - Ideally, an analytic solution
- Examine output--systems like MATLAB are useful

Test Cases

- Test cases are also useful for sharing and extending code
  - Provide a suite of test cases to new users
    - Tests that code runs on their system
    - Helps them learn to run your code
  - Make sure your extensions didn't break anything!

Test Cases for Model Problem

- Advection only:
  - If u is constant, then regain initial conditions at t= L/u
- Diffusion only, constant:
  - Complicated analytical solution, but know that solution should tend to mean(C)
- Reaction only
  - Solution should agree with ODE: dC/dt=reaction (each grid point represents a unique initial condition)

Flaw of Test Cases

- Test cases are simple and usually won't test all parts of the code
- Pick small, but hard cases to test logic of program
  - Small enough that you can reason them through yourself
- Test subroutines individually, then test program
- Key of testing:
  - Look at your output!

Next 2 weeks

- Debugging tools
- Version control & working in groups
- Profiling and tuning (lecture and lab)
- Picking a system
- Other ideas