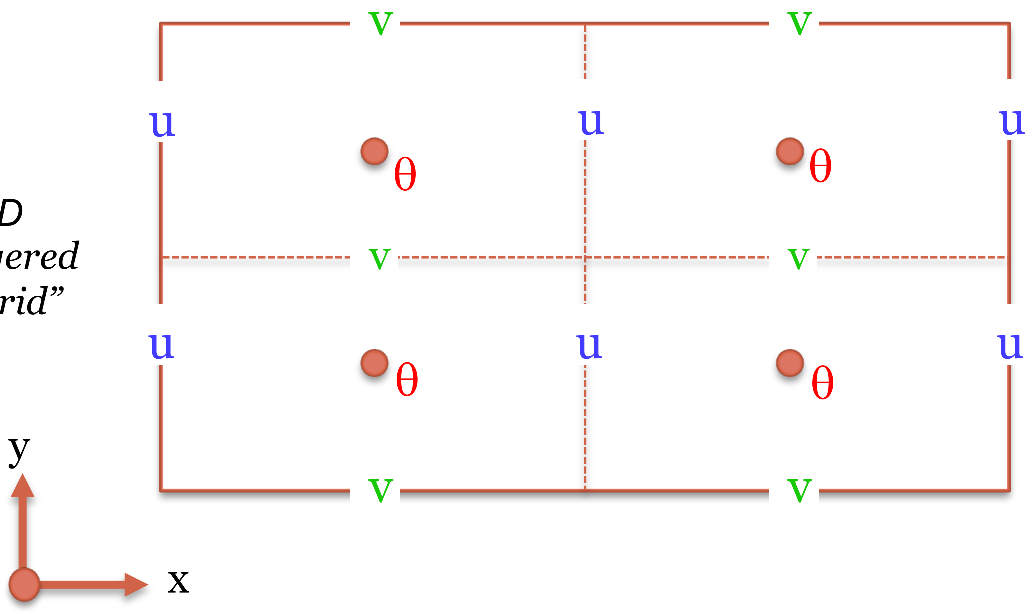
Operator splitting:
Computer Program #2

TWO-DIMENSIONAL ADVECTION

2-D Staggered ”C-grid”



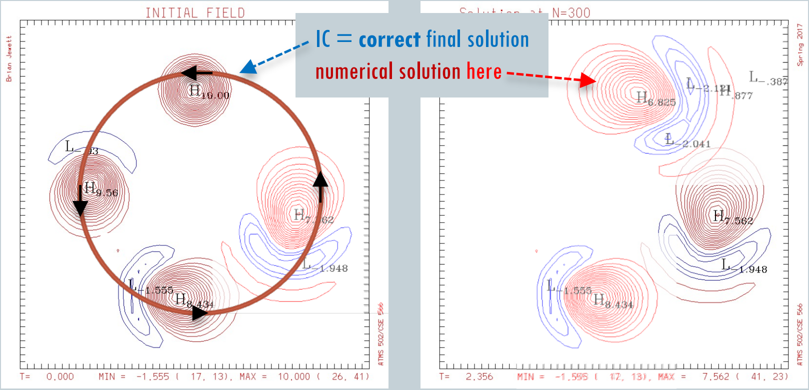
To start program #2:

1. Login
2. cp -R Pgm1 Pgm2

   ... copies the entire Pgm1 folder into new "Pgm2" .
3. Keep your old programs!
4. Work in the new folder.

Program 2 - what we're doing (and why)

- In program 1, we used periodic boundary conditions so *the initial condition was the (perfect) final solution*
  - This made for an easy visual assessment of the solution



IC = **correct** final solution

numerical solution here

- In pgm #2, circular flow (counter-clockwise) brings the "cone" one full rotation
  - *The initial condition is again the* (correct) *final solution*
  - This permits an easy visual assessment of the solution, *and* -
  - We use Takacs' error measures to quantify the behavior, with a breakdown into *dissipation* /*dispersion* / *total* error.

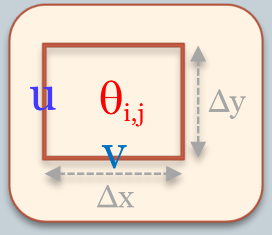
Program 2 changes - arrays, parameters

- Revise parameters and convert arrays to 2-D
  - s1 array becomes 2-D, with 3 ghost points (for 3rd numerical method)
  - old 1-D flow speed "u" replaced with two 2-D arrays *u*, *v*
    - u and v are 2-D but have *no* ghost points: set in IC, constant w/time!
    - dimensions of u, v vary due to staggering: u(nx+1, ny) • v(nx, ny+1)
  - need parameters "nx" as well as "ny" (later, nx will *not* equal ny!)
    - in C, need J1, J2 and NY; BCWIDTH is now 3
  - strue() array: easiest to dimension same as s1()
  - smin, smax arrays: 1-D • dimension: max-number-of-time steps
    - This replaces the old "strace" array in program 1
  - routine *plot1d* no longer needed • you'll be changing *Makefile !*

Program 2 - initial conditions (IC)

- Get IC working <u>before</u> doing: *bc, advection, advect1d*
  - Remember the main three array sizes <u>differ</u>:
    - s1(-2:nx+3, -2:ny+3) C: s1[NXDIM][NYDIM] ... with BCWIDTH=3
    - u(nx+1 , ny) C: u[NX+1][NY]
    - v(nx , ny+1) C: v[NX][NY+1]
  - Because of the different array sizes, and *different physical locations* of s1, u, and v on the staggered grid, we use <u>three</u> separate double-loops to initialize arrays: s1, u, v.
    - one loop is for the second (Y) dimension of the arrays;
    - the other loop is for the first (X) dimension of the arrays.

C-GRID



  - Physical locations of variables:
    - s1 physical domain extends from -0.5 to +0.5; s1(1,1) at (-0.5, -0.5)
    - u is Δx/2 to *left (west)* of s1; v is Δy/2 *below (to south of)* s1.