Subject: Re: looking for IDL streamlines help
Posted by agraps on Thu, 12 Dec 1996 08:00:00 GMT
View Forum Message <> Reply to Message

nolf@cscsun3.larc.nasa.gov (Scott Nolf) writes:

> I'm having trouble figuring out how to do streamlines in IDL. What I'm
> looking for is something analogous to the STRMLN routine in NCAR
> graphics. It seems that the IDL routines PLOT_FIELD or VEL would
> be applicable, but I guess I'm not getting the keywords set correctly.

> Any help would be greatly appreciated.

> Scott Nolf

> p.s. - the data I'd like to streamline is global u and v wind components
> at a particular pressure level (ECMWF data).

Several years ago I needed to plot global u and v wind components at a particular pressure level of ECMWF data too..

I discovered in the process (meteorology is not really my field) that meteorlogists often use 'streamlines' differently than how my vector mechanics texts defined it.

There seem to be two ways "streamlines" are used in meteorology:

-streamlines calculated by integrating tangents to the vector field.
-streamlines calculated by finding the level sets of stream functions.

The first was the standard way from vector mechanics texts, and the second way was new to me, but defined in fluid mechanics and meteorology texts. The first way is also very s l o w. Because you are integrating a 2D field point-by-point.

I wrote up some notes about how to do for both ways. The part that I'm not satisfied with though, is that I got streamlines that didn't look the same for both ways.

I'll append my notes about how to do this, but you can also retrieve them at:

http://www.amara.com/ftpstuff/streamlines1.txt
http://www.amara.com/ftpstuff/streamlines2.txt

And you can see output from the second way of my plotting u,v wind vectors of ECMWF data at:

http://www.amara.com/past/viz.html

If you are interested in my IDL code to do this, let me know. It's not in a very polished form and I don't give guarantees on it, especially since that project was something on the side I did a long time ago that I was just playing with. I believe that I took Dave Stern's VEL routine and modified it a little for my purpose. And like I said, I'm not satisfied that it's giving me the right answers.

Amara

Calculating Streamlines by Integrating Tangents to the Vector Field. (method 1)

There seem to be two ways "streamlines" are used in meteorology:

-streamlines calculated by integrating tangents to the vector field.
-streamlines calculated by finding the level sets of stream functions.

My method below describes the first calculation. It is computationally very intensive, even though I was using a simple Euler algorithm for integration.

The idea is to integrate (x,y) using (dx, dy) until I reach the edges of the grid (x_0, y_0) and (x_1, y_1) that I want to have streamlines for. To integrate, I use the nearest vectors in a weighted way- the weights for the current vector are inversely proportional to their distance from the current vector.

0. Given a vector (x,y) (dx, dy)

1. Find nearest "n" vectors to any 1 vector (to calculate derivative functions of the vector (x,y)).

For example if we use the 4 nearest vectors,
then let d1, d2, d3, d4 = nearby vector distances from current vector.

2. Calculate the derivative functions gx(x_i, y_i) and gy(x_i, y_i) of (x,y) that will be used in the Euler integration using nearest vectors.

I.e. our Euler integration scheme will be:
x_{i+1} = x_i + gx(x_i, y_i) * h
y_{i+1} = y_i + gy(x_i, y_i) * h

where h is a step size that is some fraction smaller than
your grid.

Let w1, w2, w3, w4 = weights for nearby vectors, and
k = a proportionality constant which we must calculate.

Equation being solved for the weights is:
k/d1 + k/d2 + k/d3 +k/d4 = 1

So we want: w1 = k/d1, w2 = k/d2, w3 = k/d3, w4 = k/d4

I.e.:
k = (d1*d2*d3*d4) / (d2*d3*d4 + d1*d3*d4 + d1*d2*d4 + d1*d2*d3)
w1 = k/float(d1) & w2 = k/float(d2) & w3 = k/float(d3) &
w4 = k/float(d4)

Then calculate the gx and gy:

```
;x-component
d1x = nearest_gx(1) &  d2x = nearest_gx(2)
d3x = nearest_gx(3) &  d4x = nearest_gx(4)
gx = (w1*d1x + w2*d2x + w3*d3x + w4*d4x)/4.0

;y-component
d1y = nearest_gy(1) &  d2y = nearest_gy(2)
d3y = nearest_gy(3) &  d4y = nearest_gy(4)
gy = (w1*d1y + w2*d2y + w3*d3y + w4*d4y)/4.0
```

where:
"nearest_gx( )" = nearby vector array of deriv function in x
direction

"nearest_gy( )" = nearby vector array of deriv function in y
direction

3. Apply Eulers algorithm to integrate to next (forward or backward)
step in the streamline. In IDL, this part might look like:

```
case j of
  0: begin
    ;forward
    xstream(i,istep, j) = curr_x + gx * h
    ystream(i,istep, j) = curr_y + gy * h
    end

  1: begin
```

```
      ;backward
      xstream(i,istep, j) = curr_x - gx * h
      ystream(i,istep, j) = curr_y - gy * h
      end
  endcase
  curr_x = xstream(i,istep, j)
  curr_y = ystream(i,istep, j)

  ;See if out of bounds to end integration
  case 1 of
    curr_x le xleft: out_of_bounds = 1  ;true
    curr_x ge xright: out_of_bounds = 1
    curr_y le ybottom: out_of_bounds = 1
    curr_y ge ytop: out_of_bounds = 1
    else: out_of_bounds = 0       ;false
  endcase
```

Then "xstream" and "ystream" are the x and y components of the streamline that you're looking for, given a wind vector and some nearby vectors.

I really need a way to draw a picture to show how this works, but I hope you can tell what's going on with the equations and text.

Calculating Streamlines by Finding the Level Sets of Stream Functions (method 2)

There seem to be two ways "streamlines" are used in meteorology:

-streamlines calculated by integrating tangents to the vector field.
-streamlines calculated by finding the level sets of stream functions.

My method below describes the second calculation. This method assumes mass conservation.

Say F is the stream function.

u = -dF/dy

v = dF/dx

(d's are partial derivatives)

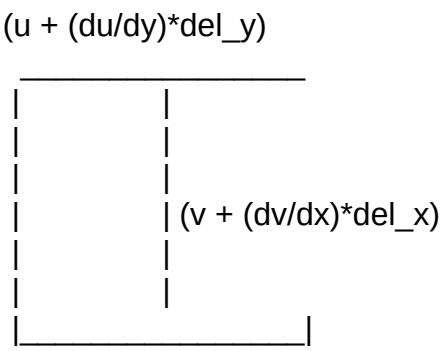
From these equations:

d^2 F/dx^2 + d^2F/dy^2 = d^2 F

= dv/dx - du/dy = psi.

where psi = the vertical component of vorticity (which equals the circulation about a loop in the limit where the area approaches 0)

The relationship between circulation and vorticity for an area element in the horizontal plane looks sort of like:



v velocity compoent up ^

u velocity component right ->

Stokes theorem states that the circulation about any closed loop is equal to the integral of the normal compoent of the vorticity over the area enclosed by the loop. Hence, for a finite area, the circulation divided by the area gives the average normal compoent of vorticity the region.

A streamline is a continuous line through the fluid such that it has the direction of the velocity at every point throughout its length.

(See Holton: _An Introduction to Dynamic Meteorology_ pg. 66., also Streeter: _Fluid Dynamics_)

I fill in a grid of streamfunctions by using the horizontal and vertical components of the (wind) velocity. Then I contour the stream function to get level sets -> which gives me my streamlines.

Step One. Fill in a grid of points with your stream function.

- Define min/max x and y for your grid, as well as delta_x, delta_y.

let N = length of square grid

- Initialize F(N,N)

- Set F(0,0) = 0 (a constant)

- Calculate "up" to fill in the left side of the grid

F(0,1) = C + Integral(v dx) - Integral(u dy) = -Integral(u dy)
(these integrals are closed integrals)

Using Trapezoidal Rule: Int_0^h f(x)dx = 1/2(f(0)+f(h))*h

-Integral(u dy) = -1/2(u(0,0) + u(0,1)) * delta_y for value
of stream function 1st step "up"

or more generally:

F(0,n+1) = -1/2 (u(0,n) + u(0,n+1)) *delta_y + F(0,n)

n are the individual boxes in your grid. You may want do interpolation,
say by using nearest neighbors, to fill in the grid more.

Repeat for position y_{n+1} to fill in the left side of the grid.

- Calculate "right" to fill in the bottom side of the grid

An identical calculation gives you:

F(n+1,0) = -1/2 (v(n,0) + v(n+1,0)) *delta_x + F(n,0)

Repeat for position y_{n+1} to fill in the left side of the grid.

Step 2. Now that we have F(*,0), do the same for F(*,1), F(*,2)
etc. and finally CONTOUR F(*,*) by using x,y as the grid.

---------- -

Amara

--

****
Amara Graps email: agraps@netcom.com
Computational Physics vita: finger agraps@best.com
Multiplex Answers URL: http://www.amara.com/
****