---

## Subject: Velocity vectors on maps
## Posted by bowman on Wed, 06 Oct 1999 07:00:00 GMT
View Forum Message <> Reply to Message

---

Can anyone give me pointers to IDL procedures to making correct velocity
vector plots on maps, notably azimuthal polar projections?

This is one of those annoying inadequacies in IDL (see earlier discussion
on mathematical routines) that include niggling mapping and contouring
problems. (NCAR graphics did vector and streamline plots well years ago
-- if you could stand to use NCAR graphics.)

Thanks, Ken Bowman

--
Dr. Kenneth P. Bowman, Professor 409-862-4060
Department of Meteorology 409-862-4466 fax
Texas A&M University bowmanATcsrp.tamu.edu
College Station, TX 77843-3150 Replace AT with @

---

## Subject: Re: Velocity vectors on map
## Posted by m218003 on Mon, 25 Oct 1999 07:00:00 GMT
View Forum Message <> Reply to Message

---

In article <38074F24.15FB@csrp.tamu.edu>,
Kyong Hwan Seo <khseo@csrp.tamu.edu> writes:
> Hi folks,
>
> Does anyone have a code to plot wind vectors correctly on a map
> (azimuthal polar projection)?
> The vector length should not plotted with respect to a cell (In the
> procedure "VELOVECT", the vector length is with respect to a cell) ..
> If you have any idea about this, please reply for me.
> Thanks,

try out my modification of VELOVECT which allows for irregularily spaced
data and includes your desired modification for a proper LENGTH keyword
(by Franz Rohrer).

Please find msvelovect.pro attached.

Martin

--
[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[ [[[[[[[
[[ Dr. Martin Schultz Max-Planck-Institut fuer Meteorologie [[
[[ Bundesstr. 55, 20146 Hamburg [[

```
[[                    phone: +49 40 41173-308 or 416         [[
[[                    fax:   +49 40 41173-298                [[
[[ martin.schultz@dkrz.de                                    [[
 [[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[ [[[[[[[
```

```
; $Id: velovect.pro,v 1.15 1998/06/26 22:13:02 griz Exp $
;
; Copyright (c) 1983-1998, Research Systems, Inc.  All rights reserved.
; Unauthorized reproduction prohibited.

PRO MSVELOVECT,U,V,X,Y, Missing = Missing, Length = length, Dots = dots,  $
        Color=color, CLIP=clip, NOCLIP=noclip, OVERPLOT=overplot,  $
        NOZERO=nozero, _EXTRA=extra
;
;+
; NAME:
; MSVELOVECT
;
; PURPOSE:
; Produce a two-dimensional velocity field plot.
;
; A directed arrow is drawn at each point showing the direction and
; magnitude of the field.
;
;       This routine was developed from the original VELOVECT routine
;       and now allows two dimensional arrays for X and Y for irregularly
;       spaced data.
;
; CATEGORY:
; Plotting, two-dimensional.
;
; CALLING SEQUENCE:
; MSVELOVECT, U, V [, X, Y]
;
; INPUTS:
; U: The X component of the two-dimensional field.
;  U must be a two-dimensional array.
;
; V: The Y component of the two dimensional field.  Y must have
;  the same dimensions as X.  The vector at point [i,j] has a
;  magnitude of:
;
;   (U[i,j]^2 + V[i,j]^2)^0.5
;
;  and a direction of:
;
;   ATAN2(V[i,j],U[i,j]).
;
```

```
; OPTIONAL INPUT PARAMETERS:
;  X: Optional abcissae values.  X must be a vector with a length
;  equal to the first dimension of U and V *OR* a 2-dimensional
;               array with the same dimensions as U and V.
;
; Y: Optional ordinate values.  Y must be a vector with a length
;  equal to the first dimension of U and V *OR* a 2-dimensional
;               array with the same dimensions as U and V.
;
; KEYWORD INPUT PARAMETERS:
; COLOR: The color index used for the plot.
;
; DOTS: Set this keyword to 1 to place a dot at each missing point.
;  Set this keyword to 0 or omit it to draw nothing for missing
;  points.  Has effect only if MISSING is specified.
;
; LENGTH: Length factor.  The default of 1.0 makes the longest (U,V)
;  vector the length of a cell.
;              ### Franz Rohrer's modification of the LENGTH keyword:
;              LENGTH now applies a scale factor relative to the data values,
;              not just a relative scaling. If you don't specify LENGTH it acts
;              as before.
;
;       MISSING: Missing data value.  Vectors with a LENGTH greater
;  than MISSING are ignored.
;
;       OVERPLOT: Set this keyword to make VELOVECT "overplot".  That is, the
;               current graphics screen is not erased, no axes are drawn, and
;               the previously established scaling remains in effect.
;
;       NOZERO: Do not plot zero vectors as dots.
;
; Note:   All other keywords are passed directly to the PLOT procedure
;  and may be used to set option such as TITLE, POSITION,
;  NOERASE, etc.
; OUTPUTS:
; None.
;
; COMMON BLOCKS:
; None.
;
; SIDE EFFECTS:
; Plotting on the selected device is performed.  System
; variables concerning plotting are changed.
;
; RESTRICTIONS:
; None.
;
```

```
; PROCEDURE:
; Straightforward.  Unrecognized keywords are passed to the PLOT
; procedure.
;
; MODIFICATION HISTORY:
; DMS, RSI, Oct., 1983.
; For Sun, DMS, RSI, April, 1989.
; Added TITLE, Oct, 1990.
; Added POSITION, NOERASE, COLOR, Feb 91, RES.
; August, 1993.  Vince Patrick, Adv. Visualization Lab, U. of Maryland,
;  fixed errors in math.
; August, 1993. DMS, Added _EXTRA keyword inheritance.
; January, 1994, KDB. Fixed integer math which produced 0 and caused
;             divide by zero errors.
; December, 1994, MWR. Added _EXTRA inheritance for PLOTS and OPLOT.
; June, 1995, MWR. Removed _EXTRA inheritance for PLOTS and changed
;    OPLOT to PLOTS.
;       September, 1996, GGS. Changed denominator of x_step and y_step vars.
;       February, 1998, DLD.  Add support for CLIP and NO_CLIP keywords.
;       June, 1998, DLD.  Add support for OVERPLOT keyword.
;       16 Sep 1999: Martin Schultz added support for 2D U and V arrays
;             cleaned up the routine some and added the NOZERO keyword.
;             (renamed as msvelovect.pro on Sep 23)
;             Also included Franz Rohrer's modification of the LENGTH keyword:
;             LENGTH now applies a scale factor relative to the data values,
;             not just a relative scaling. If you don't specify LENGTH it acts
;             as before.
;-
;
       on_error,2                     ;Return to caller if an error occurs
       s = size(u)
       t = size(v)
       if s[0] ne 2 then begin
baduv:     message, 'U and V parameters must be 2D and same size.'
       endif
       if total(abs(s[0:2]-t[0:2])) ne 0 then goto,baduv
;
       if n_params() lt 3 then x = findgen(s[1]) $
       else begin
          sx = size(x)
          if (sx[0] eq 2) then begin
             if total(abs(sx[0:2]-s[0:2])) ne 0 then begin
badx:              message, 'X array has incorrect size.'
             endif
          endif else $
               if n_elements(x) ne s[1] then goto,badx
       endelse
;
```

```
        if n_params() lt 4 then y = findgen(s[2]) $
        else begin
          sy = size(y)
          if (sy[0] eq 2) then begin
            if (sx[0] ne 2) then goto,bady
            if total(abs(sy[0:2]-s[0:2])) ne 0 then begin
bady:               message, 'Y array has incorrect size.'
            endif
          endif else $
              if n_elements(y) ne s[1] then goto,bady
        endelse
;
        if n_elements(missing) le 0 then missing = 1.0e30
; ### FR: use LENGTH differently -- allows absolute scaling
;       if n_elements(length) le 0 then length = 1.0

        mag = sqrt(u^2.+v^2.)             ;magnitude.
                ;Subscripts of good elements
        nbad = 0                        ;# of missing points
; ## mgs: because of defaulting 5 lines above, missing always has a value!!
;       if n_elements(missing) gt 0 then begin
          good = where(mag lt missing)
          if keyword_set(dots) then bad = where(mag ge missing, nbad)
;       endif else begin
;               good = lindgen(n_elements(mag))
;       endelse

        ugood = u[good]
        vgood = v[good]
        x0 = min(x)                     ;get scaling
        x1 = max(x)
        y0 = min(y)
        y1 = max(y)
 x_step=(x1-x0)/(s[1]-1.0)   ; Convert to float. Integer math
 y_step=(y1-y0)/(s[2]-1.0)   ; could result in divide by 0

 maxmag=max([max(abs(ugood/x_step)),max(abs(vgood/y_step))])
; ### FR:
                if n_elements(length) gt 0 then maxmag=length/x_step
        sina = (ugood/maxmag)
        cosa = (vgood/maxmag)
; ### original:
;       sina = length * (ugood/maxmag)
;       cosa = length * (vgood/maxmag)
;
        if n_elements(title) le 0 then title = ''
        ;--------------  plot to get axes  ---------------
        if n_elements(color) eq 0 then color = !p.color
```

```
        if n_elements(noclip) eq 0 then noclip = 1
        x_b0=x0-x_step
 x_b1=x1+x_step
 y_b0=y0-y_step
 y_b1=y1+y_step
        if (not keyword_set(overplot)) then begin
;         if n_elements(position) eq 0 then begin
            plot,[x_b0,x_b1],[y_b1,y_b0],/nodata,/xst,/yst, $
              color=color, _EXTRA = extra
;         endif else begin
;           plot,[x_b0,x_b1],[y_b1,y_b0],/nodata,/xst,/yst, $
;             color=color, _EXTRA = extra
;         endelse
        endif
        if n_elements(clip) eq 0 then $
            clip = [!x.crange[0],!y.crange[0],!x.crange[1],!y.crange[1]]
;
        r = .3                          ;len of arrow head
        angle = 22.5 * !dtor            ;Angle of arrowhead
        st = r * sin(angle)             ;sin 22.5 degs * length of head
        ct = r * cos(angle)
;
        for i=0,n_elements(good)-1 do begin     ;Each point
             if (sx[0] eq 2) then begin
                x0 = x[good[i]]         ;get coords of start & end
                y0 = y[good[i]]
             endif else begin
                x0 = x[good[i] mod s[1]]        ;get coords of start & end
                y0 = y[good[i] / s[1]]
             endelse
                dx = sina[i]
                x1 = x0 + dx
                dy = cosa[i]
                y1 = y0 + dy
  xd=x_step
  yd=y_step
                ; plot zero vectors as dots
                if (mag[i] eq 0.) then begin
                  if  (not keyword_set(NOZERO)) then $
                    plots,x[i],y[i],psym=3,color=color,clip=clip,  $
                    noclip=noclip
                endif else $
                   plots,[x0,x1,x1-(ct*dx/xd-st*dy/yd)*xd, $
   x1,x1-(ct*dx/xd+st*dy/yd)*xd], $
                      [y0,y1,y1-(ct*dy/yd+st*dx/xd)*yd, $
   y1,y1-(ct*dy/yd-st*dx/xd)*yd], $
                      color=color,clip=clip,noclip=noclip
        endfor
```

```
    if nbad gt 0 then begin
      if (sx[0] eq 2) then begin          ;Dots for missing?
          PLOTS, x[bad], y[bad], psym=3, color=color, $
                clip=clip,noclip=noclip
      endif else begin
          PLOTS, x[bad mod s[1]], y[bad / s[1]], psym=3, color=color, $
                clip=clip,noclip=noclip
      endelse
    endif

end
```

## File Attachments

1) msvelovect.pro, downloaded 69 times