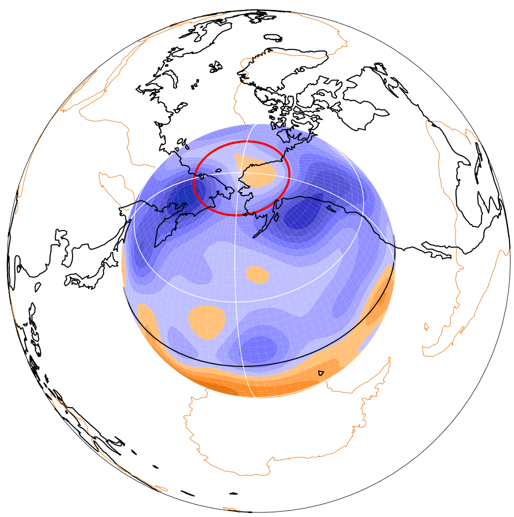
LEOPACK



arrows_const_r3

Constant radius plot with the ability to display flow arrows.

Steven J. Gibbons, Oslo

Original document: November 21^{st}, 2001. Updated: October 29^{th}, 2022.

1 arrows_const_r3

Source code is in

`LEOPACK_DIR/GPROGRAMS/arrows_const_r3.f`

although subprograms from `LEOPACK_DIR/GSUBS`, `LEOPACK_DIR/SUBS` and `LEOPACK_DIR/LINALG` are also required.

`arrows_const_r3` plots - from a solution in standard format (i.e. `.ints`, `.vecs` and `.xarr` files) - a rectangular diagram of a function at a constant spherical radius. Arrows indicating the horizontal flow at the specified radius may be added to the contours. A typical input file is

---

```
* input file for arrows_const_r3
example_fOUTPUT                           : Filename stem
../../EXAMPLES/FUNDAMENTALS/case1.ints    : integers
../../EXAMPLES/FUNDAMENTALS/case1.vecs    : vectorfile
../../EXAMPLES/FUNDAMENTALS/case1.xarr    : radialfile
   0.0  360.0    -88.0    88.0   0.96     : LONG1, LONG2, LAT1, LAT2, RAD
  60  140   3                             : NTHE  NPHI   NNDS
 0.05  0.95   0.05   0.95   6.0   0.8     : xleft xright ybot ytop rwidth rratio
 16   6    4    4   0                     : nlev idev  icont  icomp   ias
 120.0   0.0  1.0   0.85   1              : huepos, hueneg, csat, scal, iw
 120.0   0.0   0.00  2    1               : rphue rpsat rplight ipw ips
   0.0   0.0   0.00  2    3               : rnhue rnsat rnlight inw ins
 200  0.10   0.9                          : npba  rlong  rhead
   1        0.0       0.0                 : icontour  valmin   valmax
```

Any line in the input file beginning with an asterisk, `*`, is ignored by the program and can thus be used to enter comments and notes.

Any line in the input file beginning with an asterisk, `*`, is ignored by the program and can thus be used to enter comments and notes.

The inputs in the above file are as follows

- `Filename stem`. First characters in output files to be generated by current run. Running `arrows_const_r3` with the above input file will create either the file `example_fOUTPUT.ps` or `example_fOUTPUT.gif`: depending upon the value of the integer flag `idev`.

- `integers`: name of already existing indices file describing solution.

- `vectorfile`: name of already existing vector file describing solution. Must contain the same number of radial functions as indicated in the `.ints` file.

- `radialfile`: name of already existing radial spacings file describing solution. Must contain the same number of radial grid nodes as indicated in the `.vecs` file.

- `LONG1`: The longitude (in degrees) which is to be at the left hand side of the plot.
- `LONG2`: The longitude (in degrees) which is to be at the right hand side of the plot.
- `LAT1`: The latitude (in degrees) which is to be at the bottom of the plot.
- `LAT2`: The latitude (in degrees) which is to be at the top of the plot.
- `RAD`: Proportion of distance between inner and outer boundaries. Must be in the range $[0, 1]$. If x represents the number `RAD` then the actual radius to be looked at is $r = r_\mathrm{i} + x(r_\mathrm{o} - r_\mathrm{i})$.
- `NTHE`: The number of equally spaced grid points for the resolution of the plot in latitude. (Note that this is entirely independent of the numerical resolution of the original solution.)
- `NPHI`: The number of equally spaced grid points for the resolution of the plot in longitude. (Note that this is entirely independent of the numerical resolution of the original solution.)
- `NNDS`. Number of nodes for interpolating radial functions. `3` is a suggested value since no great accuracy is required here.
- `xleft`. Position within the output device of the left border of figure. (i.e. defines size of the left hand side margin.) Must be in the range $[0, 1]$.
- `xright`. Position within the output device of the right border of figure. (i.e. defines size of the right hand side margin.) Must be in the range $[0, 1]$.
- `ybot`. Position within the output device of the lower border of figure. (i.e. defines size of the bottom margin.) Must be in the range $[0, 1]$.
- `ytop`. Position within the output device of the upper border of figure. (i.e. defines size of the top margin.) Must be in the range $[0, 1]$.
- `rwidth`. Width in inches of output device (i.e. horizontal dimension of postscript of gif file.)
- `rratio`. Ratio of height to width for the output device.
- `nlev`. The number of contour levels required. There is a special value `nlev = -1` which applies a 16 contour level Red/Green/Blue coefficient set provided by Andy Jackson.

- `idev`. Device number. Can take the following values:-
  `idev = 1` $\rightarrow$ landscape gif file.
  `idev = 2` $\rightarrow$ portrait gif file.
  `idev = 5` $\rightarrow$ landscape colour postscript file.
  `idev = 6` $\rightarrow$ portrait colour postscript file.

- `icont`. Specification of what to display.
  `icont = 1` $\rightarrow$ coloured contours (shading) without contour lines or arrows.
  `icont = 2` $\rightarrow$ arrows of horizontal flow without contours.
  `icont = 3` $\rightarrow$ arrows of horizontal flow superimposed upon coloured contours (shading).
  `icont = 4` $\rightarrow$ arrows of horizontal flow superimposed upon contour lines.

- `icomp`. Field component to be displayed in the contour plot. Can take the following values:-
  `icomp = 1` $\rightarrow$ radial velocity, v_r.
  `icomp = 2` $\rightarrow$ theta velocity, v_θ.
  `icomp = 3` $\rightarrow$ phi velocity, v_ϕ.
  `icomp = 4` $\rightarrow$ radial magnetic field, B_r.
  `icomp = 5` $\rightarrow$ theta magnetic field, B_θ.
  `icomp = 6` $\rightarrow$ phi magnetic field, B_ϕ.
  `icomp = 7` $\rightarrow$ temperature, T.
  `icomp = 8` $\rightarrow$ heat-flux, $-(dT/dr)$.
  `icomp = 9` $\rightarrow$ upwelling, $-(dv_r/dr)$.

- `ias`. Axisymmetric only flag.
  `ias = 0` $\rightarrow$ full 3D solution is used.
  `ias = 1` $\rightarrow$ only the axisymmetric part is used.

- `huepos`. Hue value for numbers greater than zero for functions which are to be contoured using fill (i.e for options `icont = 1` and `icont = 3`, but otherwise not referred to). Number between `0` and `360`. See Appendix (2) and Figure (7) for details.

- `hueneg`. Hue value for numbers less than zero for functions which are to be contoured using fill (i.e for options `icont = 1` and `icont = 3`, but otherwise not referred to). Number between `0` and `360`. See Appendix (2) and Figure (7) for details.

- `csat`. Saturation value for shaded contours (i.e for options `icont = 1` and `icont = 3`, but otherwise not referred to). Number between `0.0` and `1.0` `csat = 1.0` implies full colour. `csat = 0.0` means monochrome and, in this case, the values `huepos` and `hueneg` become irrelevant and a grey-shade plot results with the most negative value as white and the most positive value as black. See Appendix (2) for details.

- `scal`. A very crude means of lightening a dark plot. Normal value is `scal = 1`, but reducing this (e.g. `scal = 0.7`) may give a better picture.

- `iw`. Width of lines used to draw arrows. Integer, with `1` being the thinnest available.

- `rphue`. Hue value for numbers greater than zero for functions which are to be contoured using lines (i.e for option `icont = 4` but otherwise not referred to). Number between `0` and `360`. See Appendix (2) and Figure (7) for details.

- `rpsat`. Saturation value for contour lines with positive values. Only referred to for option `icont = 4`. Number between `0.0` and `1.0` `rpsat = 1.0` implies full colour. `rpsat = 0.0` means monochrome and, in this case, the value `rphue` becomes irrelevant.

- `rplight`. Lightness value for contour lines with positive values. Only referred to for option `icont = 4`. Number between `0.0` and `1.0` with black at lightness 0.0 and white at lightness 1.0.

- `ipw`. Width value for contour lines with positive values. Only referred to for option `icont = 4`. Integer, with `1` being the thinnest available.

- `ips`. Style value for contour lines with positive values. Only referred to for option `icont = 4`. The following options are available:
  `ips = 1` → full line.
  `ips = 2` → long dashes.
  `ips = 3` → dash-dot-dash-dot.
  `ips = 4` → dotted.
  `ips = 5` → dash-dot-dot-dot.

- `rnhue`. Hue value for numbers less than zero for functions which are to be contoured using lines (i.e for option `icont = 4` but otherwise not referred to). Number between `0` and `360`. See Appendix (2) and Figure (7) for details.

- `rnsat`. Saturation value for contour lines with negative values. Only referred to for option `icont = 4`. Number between `0.0` and `1.0` `rnsat = 1.0` implies full colour. `rnsat = 0.0` means monochrome and, in this case, the value `rphue` becomes irrelevant.

- `rnlight`. Lightness value for contour lines with negative values. Only referred to for option `icont = 4`. Number between `0.0` and `1.0` with black at lightness 0.0 and white at lightness 1.0.

- `inw`. Width value for contour lines with negative values. Only referred to for option `icont = 4`. Integer, with `1` being the thinnest available.

- `ins`. Style value for contour lines with negative values. Only referred to for option `icont = 4`. The following options are available:
  `ins = 1` → full line.
  `ins = 2` → long dashes.
  `ins = 3` → dash-dot-dash-dot.
  `ins = 4` → dotted.
  `ins = 5` → dash-dot-dot-dot.

- `npba`. Number of points between arrows. The starting point of an arrow is drawn every `npba` points in both latitude and longitude. The lower the value for `npba`, the more arrows there will be. If arrows are too concentrated in both directions, increase `npba`. If arrows are too concentrated in latitude only, then decrease `NTHE` with the same `npba` value, or increase `npba` and increase `NPHI` proportionately. If arrows are too concentrated in longitude only, then decrease `NPHI` with the same `npba` value, or increase `npba` and increase `NTHE` proportionately. A certain amount of trial and error is required in selecting verb+npba+, `NPHI` and `NTHE` such that a good spread of arrows is achieved.

- `rlong`. Length of longest arrow (which automatically corresponds to the greatest flow). All other arrows are scaled relative to this.

- `rhead`. Size of the biggest arrowhead.

- `icontour`. Chooses automatic or manual scaling of contours.
  `icontour = 1` → contours are scaled automatically and the values `valmin` and `valmax` become irrelevant.
  `icontour = 2` → contours are scaled between the following values, `valmin` and `valmax`.

- `valmin`. User-imposed minimum value for contour function. Only referred to if `icontour = 2`.

- `valmax`. User-imposed maximum value for contour function. Only referred to if `icontour = 2`.

1.1 Run-time limitations

Several parameters are set at the outset which limit the physical size of the problem.

```
       INTEGER NRMAX, NTHMAX, NLEVM, LHMAX, NHMAX, ISVMAX, NNDM,
     1         NPMAX, NPHMAX
       PARAMETER ( NRMAX = 250, NTHMAX = 250, NPHMAX = 250, NLEVM = 20,
     1             LHMAX = 124, NHMAX = 3000, ISVMAX = NRMAX*NHMAX,
     2             NNDM = 6, NPMAX = (LHMAX+1)*(LHMAX+2)/2 )
```

If the values are insufficient, then change them and recompile.

- `NRMAX` is the maximum permitted number of radial grid nodes.
- `NTHMAX` is the maximum permitted number of grid nodes in latitude.
- `NPHMAX` is the maximum permitted number of grid nodes in longitude.
- `NLEVM` is the maximum permitted number of contour levels.
- `LHMAX` is the highest permitted spherical harmonic degree, l.
- `NHMAX` is the highest permitted number of spherical harmonic radial functions.
- `NNDM` is the highest permitted value of `nnds`.

1.2 Sample runs of arrows_const_r3

The directory

`$LEOPACK_DIR/SAMPLERUNS/ARROWS_CONST_R3`

contains example input files only. Do not under any circumstances edit these files. They refer to other (solution vector) files which are elsewhere in the distribution and provide a relative path to avoid unnecessary duplication of files. The outputs from the different files are displayed here rather than left in the directory.

1.2.1 Example a

```
* input file for arrows_const_r3
example_aOUTPUT                           : Filename stem
../../EXAMPLES/FUNDAMENTALS/case1.ints    : integers
../../EXAMPLES/FUNDAMENTALS/case1.vecs    : vectorfile
../../EXAMPLES/FUNDAMENTALS/case1.xarr    : radialfile
   0.0  360.0    -88.0    88.0   0.5      : LONG1, LONG2, LAT1, LAT2, RAD
  60  140   3                             : NTHE  NPHI   NNDS
 0.05  0.95   0.05   0.95   6.0   0.8     : xleft xright ybot ytop rwidth rratio
 -1   6    1    1   0                     : nlev idev  icont  icomp   ias
 120.0   0.0  1.0   0.85   1              : huepos, hueneg, csat, scal, iw
 120.0   0.0   0.75  3    1               : rphue rpsat rplight ipw ips
   0.0   0.0   0.75  3    4               : rnhue rnsat rnlight inw ins
  3   0.16   0.9                          : npba  rlong  rhead
   1       0.0       0.0                  : icontour  valmin   valmax
```

In the first example, we contour v_r at a depth of 0.5 from the outer surface for the Case 1 benchmark dynamo ([CAC+01]). This function is displayed in Figure (1) of that paper. Note the difference in the definition of `RAD` between the programs `arrows_const_r3` and `full_sphere_plot`. In the former, `RAD` is simply a proportion of the depth of the spherical shell. In the latter, it is the actual radius.

In the first input file here, `nlev = -1` and so we use Andy Jackson's RGB colour scheme, over-riding the `huepos`, `hueneg` etc. inputs on the next line. The values `rphue` etc. on the next two lines are all ignored given that `icont = 1`.

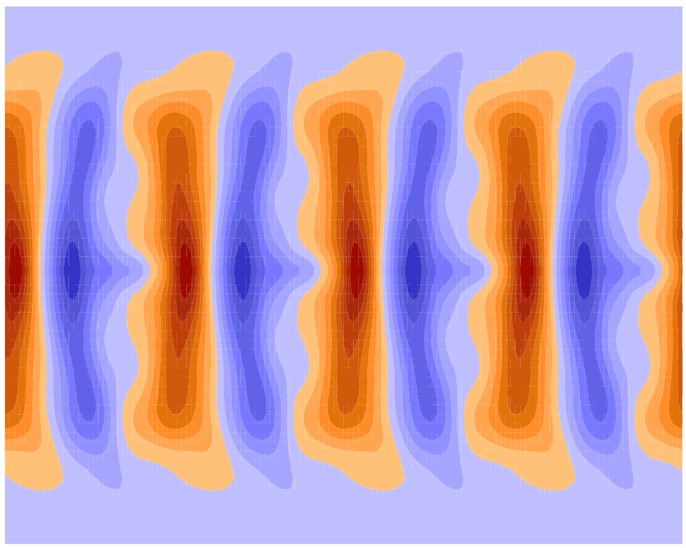


Figure 1: Output from arrows_const_r3 with example_a.input (Section 1.2.1).

1.2.2 Example b

```
* input file for arrows_const_r3
example_bOUTPUT                                : Filename stem
../../EXAMPLES/FUNDAMENTALS/case1.ints         : integers
../../EXAMPLES/FUNDAMENTALS/case1.vecs         : vectorfile
../../EXAMPLES/FUNDAMENTALS/case1.xarr         : radialfile
   0.0  360.0     -88.0     88.0    0.96       : LONG1, LONG2, LAT1, LAT2, RAD
  60   140    3                                : NTHE   NPHI   NNDS
 0.05   0.95    0.05    0.95    6.0    0.8     : xleft xright ybot ytop rwidth rratio
 14    6     3     2    0                      : nlev idev  icont  icomp   ias
 120.0    0.0   1.0    0.85    1               : huepos, hueneg, csat, scal, iw
 120.0    0.0    0.75   3     1                : rphue rpsat rplight ipw ips
   0.0    0.0    0.75   3     4                : rnhue rnsat rnlight inw ins
  3    0.16    0.9                             : npba  rlong  rhead
   1         0.0        0.0                    : icontour  valmin   valmax
```

In this second example, we consider v_θ (`icomp = 2`) much closer to the surface. With `icont = 3`, we want both arrows of flow and a contoured function: 14 contour levels with red for positive (`huepos = 120`) and blue for negative. We are encouraged to see Southbound arrows for red contour regions (positive v_θ) and Northbound arrows for blue regions! We draw an arrow at every third point in both x (`PHI`) and y (`THETA`) directions.

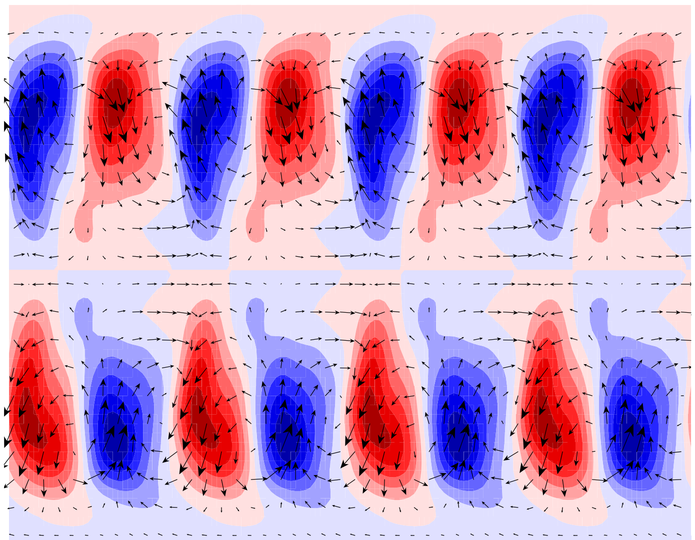


Figure 2: Output from arrows_const_r3 with example_b.input (Section 1.2.2).

1.2.3 Example c

```
* input file for arrows_const_r3
example_cOUTPUT                             : Filename stem
../../EXAMPLES/FUNDAMENTALS/case1.ints      : integers
../../EXAMPLES/FUNDAMENTALS/case1.vecs      : vectorfile
../../EXAMPLES/FUNDAMENTALS/case1.xarr      : radialfile
   0.0  360.0    -88.0    88.0    0.96      : LONG1, LONG2, LAT1, LAT2, RAD
  60  140   3                               : NTHE  NPHI   NNDS
 0.05  0.95   0.05   0.95   6.0   0.8       : xleft xright ybot ytop rwidth rratio
 14   6    4    4   0                       : nlev idev  icont  icomp   ias
 120.0   0.0  1.0   0.85   1                : huepos, hueneg, csat, scal, iw
 120.0   0.0   0.75  3    1                 : rphue rpsat rplight ipw ips
   0.0   0.0   0.75  3    4                 : rnhue rnsat rnlight inw ins
  3   0.16   0.9                            : npba  rlong  rhead
   1        0.0       0.0                   : icontour  valmin   valmax
```

For output in journals, there is often a need to produce quality monochrome diagrams showing the relationship between arrows of flow and another function contoured. Black arrows upon black contour lines can make the image very confusing and so the `icomp = 4` option can be used to subtly change the properties of the underlying contour lines to make them different from the arrows whilst still being monochrome. We impose monochromatic contours by setting `rpsat` and `rnsat` to zero. Setting `rplight` and `rnlight` to a value between zero and one makes them somewhat paler (greyer) which enhances the contrast between contours and arrows. So as not to be hidden, we make the thickness of contours `ipw = inw = 3` and finally positive and negative values are distinguished by setting the style for positive contours to solid line (`ips = 1`) and the style for negative contours to dotted lines (`ins = 4`).

The contours are of B_r close to the surface and ought therefore to be comparable with the spherical plots made by `full_sphere_plot`.

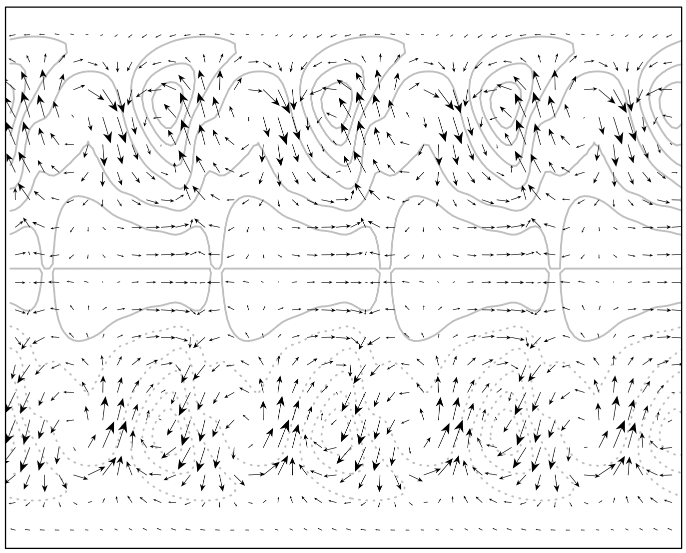


Figure 3: Output from arrows_const_r3 with example_c.input (Section 1.2.3).

1.2.4 Example d

```
* input file for arrows_const_r3
example_dOUTPUT                           : Filename stem
../../EXAMPLES/FUNDAMENTALS/case1.ints    : integers
../../EXAMPLES/FUNDAMENTALS/case1.vecs    : vectorfile
../../EXAMPLES/FUNDAMENTALS/case1.xarr    : radialfile
   0.0  360.0    -88.0    88.0   0.96     : LONG1, LONG2, LAT1, LAT2, RAD
  60  140   3                             : NTHE  NPHI   NNDS
 0.05  0.95   0.05   0.95  12.0   0.6     : xleft xright ybot ytop rwidth rratio
 14   1     4     4    0                  : nlev idev  icont  icomp   ias
 120.0   0.0  1.0   0.85  2               : huepos, hueneg, csat, scal, iw
  70.0   1.0   0.60  3    1               : rphue rpsat rplight ipw ips
```

```
 275.0   1.0   0.60  4    1              : rnhue rnsat rnlight inw ins
  3   0.16   0.9                         : npba  rlong  rhead
   1        0.0       0.0                : icontour  valmin   valmax
```

This is a gif file of essentially the same picture as before. As we are now in colour, we can have both sets of contour lines solid `ips = ins = 1` and chose different, grotesque colours for them. Note that, unlike postscript which can scale rather arbitrarily without loss of quality, gif files are pixel based and therefore, if the `rwidth` specification is too low, much quality can be lost.

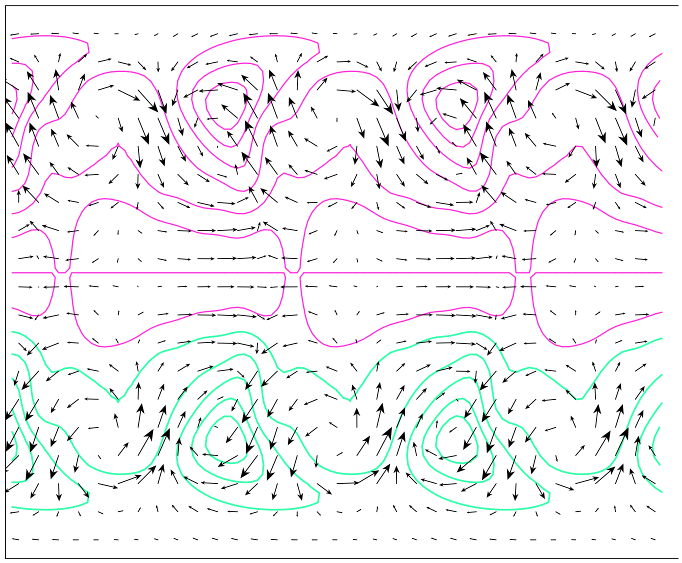


Figure 4: Output from arrows_const_r3 with example_d.input (Section 1.2.4).

1.2.5 Example e

```
* input file for arrows_const_r3
example_eOUTPUT                             : Filename stem
../../EXAMPLES/FUNDAMENTALS/case1.ints      : integers
../../EXAMPLES/FUNDAMENTALS/case1.vecs      : vectorfile
../../EXAMPLES/FUNDAMENTALS/case1.xarr      : radialfile
   0.0   90.0     20.0    70.0   0.96      : LONG1, LONG2, LAT1, LAT2, RAD
  50   90   3                               : NTHE  NPHI   NNDS
 0.05  0.95   0.05   0.95   6.0   0.75      : xleft xright ybot ytop rwidth rratio
 16   6    3    4   0                       : nlev idev  icont  icomp   ias
 240.0   0.0  1.0   0.85  4                 : huepos, hueneg, csat, scal, iw
  70.0   1.0   0.60  3    1                 : rphue rpsat rplight ipw ips
 275.0   1.0   0.60  4    1                 : rnhue rnsat rnlight inw ins
  4   0.08   1.2                            : npba  rlong  rhead
   1        0.0       0.0                   : icontour  valmin   valmax
```

Figure (5) does little more than demonstrate a close-up of a given region by changing our latitude and longitude bounds.

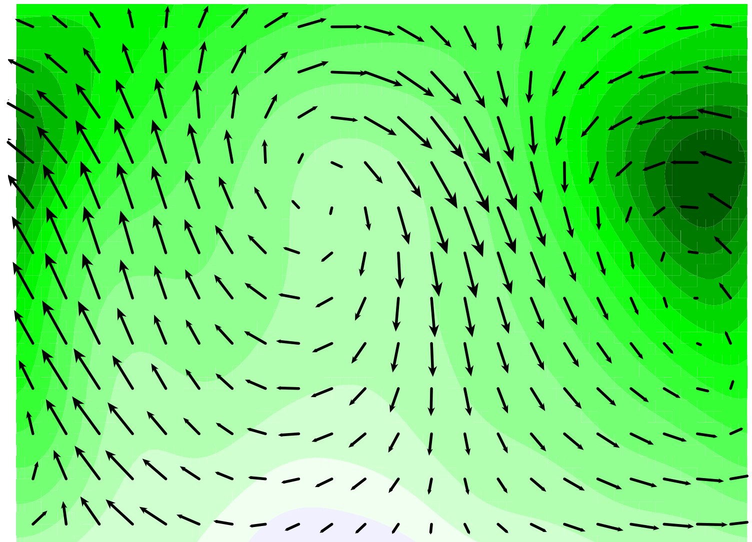


Figure 5: Output from arrows_const_r3 with example_e.input (Section 1.2.5).

1.2.6 Example f

```
* input file for arrows_const_r3
example_fOUTPUT                          : Filename stem
../../EXAMPLES/FUNDAMENTALS/case1.ints   : integers
../../EXAMPLES/FUNDAMENTALS/case1.vecs   : vectorfile
../../EXAMPLES/FUNDAMENTALS/case1.xarr   : radialfile
   0.0  360.0    -88.0    88.0   0.96    : LONG1, LONG2, LAT1, LAT2, RAD
  60  140   3                            : NTHE  NPHI   NNDS
 0.05  0.95   0.05   0.95   6.0   0.8    : xleft xright ybot ytop rwidth rratio
 16   6    4    4   0                    : nlev idev  icont  icomp   ias
 120.0   0.0  1.0   0.85   1             : huepos, hueneg, csat, scal, iw
 120.0   0.0   0.00  2    1              : rphue rpsat rplight ipw ips
   0.0   0.0   0.00  2    3              : rnhue rnsat rnlight inw ins
 200  0.10   0.9                         : npba  rlong  rhead
   1        0.0       0.0                : icontour  valmin   valmax
```

In Figure (6), we ask for an arrow every 200 grid points, and given that there are only 60 and 140 in the x and y directions respectively, we never actually see one. This is one way of doing a contour-only plot.

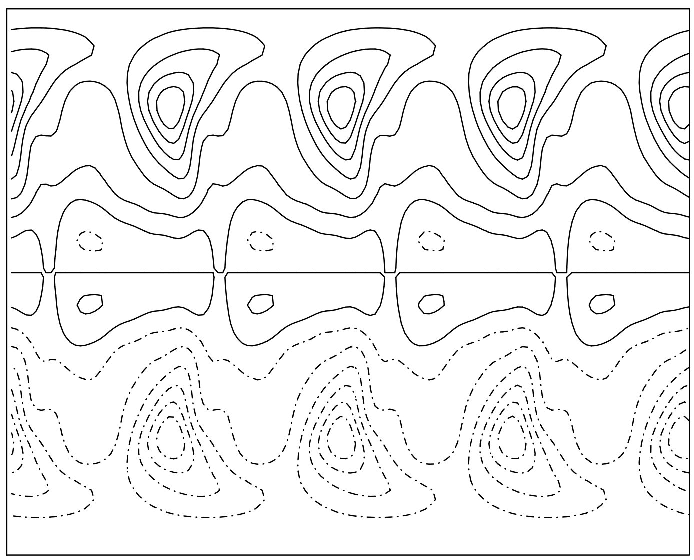


Figure 6: Output from arrows_const_r3 with example_e.input (Section 1.2.6).

2 The HLS colour scheme

When plotting using the PGPLOT software, a colour is specified by either one of the two calls

```
CALL PGSHLS( IND, CH, CL, CS )
```

or

```
CALL PGSCR( IND, CR, CG, CB )
```

The integer `IND` is the index of the colour being applied. `CR`, `CG` and `CB` are respectively the red, green and blue values in the ranges $[0, 1]$.

Figure 7: Colours as described by the integer HUE in the HLS (Hue, Light and Saturation) colour scheme.

The alternative HLS (Hue, Light and Saturation) system takes three real values

- `CH`. Hue. This is an angle between zero and 360 degrees which specifies the colour. Red is 120, Green is 240 and Blue is 0 (or 360). The full spectrum, in intervals of 5 degrees, is displayed in Figure (7).

- `CL`. Light. Ranges from 0.0 to 1.0 with black at lightness 0.0 and white at lightness 1.0.

- `CS`. Saturation. Ranges from 0.0 (grey) to 1.0 (pure colour). Hue is irrelevant when saturation is 0.0

I opted for the HLS system for the general graphics system - not because I thought the results were better - but because it is simply much easier to apply. I generally set one hue value for positive values and one for negative values and then vary the lightness as a function of the numbers being plotted.

Other users may find this colour scheme unappealing and so are welcome to devise a better way of assigning colours to contour levels! I did a job for Andy Jackson last year, for which he gave me a set of 16 red, green and blue (RGB) coefficients. This scheme is very nice and so I have implemented it in the majority of the codes as a special value of `NLEV` (the number of contour levels). Setting `NLEV = -1` should implement this colour scheme, resulting in 16 contour levels. I never got round to implementing any more general RGB scheme.

References

[CAC+01] U. R. Christensen, J. Aubert, P. Cardin, E. Dormy, S. Gibbons, G. A. Glatzmaier, E. Grote, Y. Honkura, C. Jones, M. Kono, M. Matsushima, A. Sakuraba, F. Takahashi, A. Tilgner, J. Wicht, and K. Zhang. A numerical dynamo benchmark. *Phys. Earth Planet. Inter.*, 128:25–34, 2001.