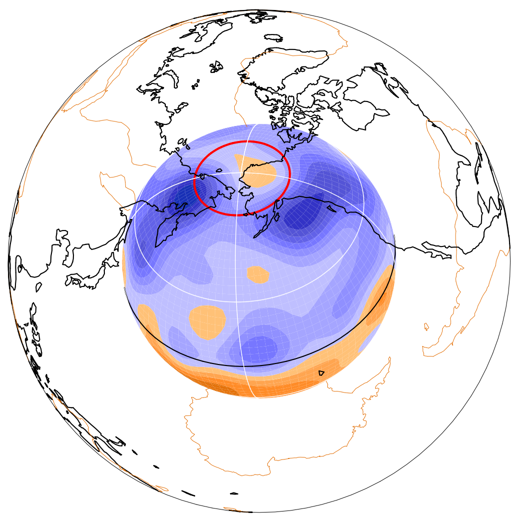
LEOPACK



continent_arrows_const_r3

Continents and arrows at constant radius

Steven J. Gibbons, Oslo

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

1 continent_arrows_const_r3

Source code is in

`LEOPACK_DIR/GPROGRAMS/continent_arrows_const_r3.f`

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

This program is not given a full section as it is in principle identical to the program `arrows_const_r3`: differing only in that it adds the outline of the continents on the surface of the sphere. For this to work, you need to include the file `coast.dat` in the directory of execution.

`continent_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 continent_arrows_const_r3
*
example_aOUTPUT_continent_arrows_const_r3  :  Filename stem
case1.ints                                : Name of integers file
case1.vecs                                : Name of vectors file
case1.xarr                                : Name of xarr file
-180.0  180.0   -88.0    88.0   0.98        : LONG1, LONG2, LAT1, LAT2, RAD
  80  160   3                               : NTHE  NPHI   NNDS
 0.10  0.90   0.20   1.00   8.0   0.6    : xleft xright ybot ytop rwidth rratio
 20   6    3    7   0                    : nlev idev  icont  icomp   ias
 120.0   0.0  1.0   0.85   1             : huepos, hueneg, csat, scal, iwcont
 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   0.15   0.9     2                   : npba  rlong  rhead  iwarrow
   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.

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` → landscape gif file.
  `idev = 2` → portrait gif file.
  `idev = 5` → landscape colour postscript file.
  `idev = 6` → portrait colour postscript file.

- `icont`. Specification of what to display.
  `icont = 1` → coloured contours (shading) without contour lines or arrows.
  `icont = 2` → arrows of horizontal flow without contours.
  `icont = 3` → arrows of horizontal flow superimposed upon coloured contours (shading).
  `icont = 4` → 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` → radial velocity, v_r.
  `icomp = 2` → theta velocity, v_θ.
  `icomp = 3` → phi velocity, v_ϕ.
  `icomp = 4` → radial magnetic field, B_r.
  `icomp = 5` → theta magnetic field, B_θ.
  `icomp = 6` → phi magnetic field, B_ϕ.
  `icomp = 7` → temperature, T.
  `icomp = 8` → heat-flux, $-(dT/dr)$.
  `icomp = 9` → upwelling, $-(dv_r/dr)$.

- `ias`. Axisymmetric only flag.
  `ias = 0` → full 3D solution is used.
  `ias = 1` → 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 Section (2) and Figure (2) 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 Section (2) and Figure (2) 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 Section (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 Section (2) and Figure (2) 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 Section (2) and Figure (2) 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 continent_arrows_const_r3

1.2.1 Example a

```
*
* input file for continent_arrows_const_r3
*
example_aOUTPUT_continent_arrows_const_r3  :  Filename stem
case1.ints                                : Name of integers file
case1.vecs                                : Name of vectors file
case1.xarr                                : Name of xarr file
-180.0  180.0   -88.0    88.0   0.98          : LONG1, LONG2, LAT1, LAT2, RAD
  80  160   3                                 : NTHE  NPHI   NNDS
 0.10  0.90   0.20   1.00   8.0   0.6    : xleft xright ybot ytop rwidth rratio
 20   6    3    7   0                    : nlev idev  icont  icomp   ias
 120.0   0.0  1.0   0.85   1             : huepos, hueneg, csat, scal, iwcont
 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   0.15   0.9     2                   : npba  rlong  rhead  iwarrow
   1       0.0       0.0                 : icontour  valmin   valmax
```

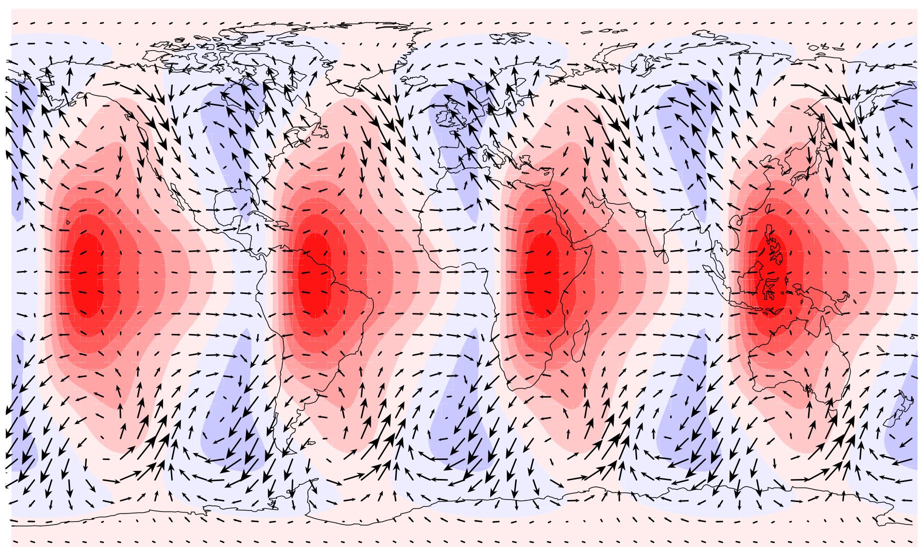


Figure 1: Output from continent_arrows_const_r3 with example_a.input (Section 1.2.1).

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 2: 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 (2).
- `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.