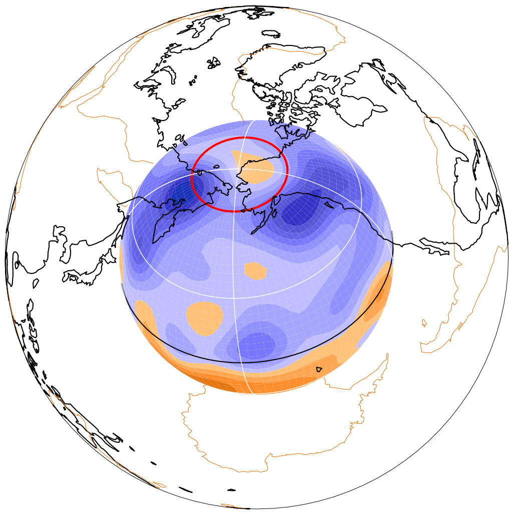
LEOPACK



krcmrnif

Kumar **R**oberts velocity **C**ritical **M**agnetic **R**eynolds **N**umber **I**terative **F**ind

Steven J. Gibbons, Oslo

Original document: November 21st, 2001. Updated: October 30th, 2022.

1 krcmrnif

Kumar **R**oberts velocity **C**ritical **M**agnetic **R**eynolds **N**umber **I**terative **F**ind

The majority of these notes are directly plagerised from [Sar94].

The equation describing the evolution of a magnetic field, $\boldsymbol{B}$, in a conducting fluid with velocity $\boldsymbol{u}$ is derived from the pre-Maxwell equations

$$\nabla \times \boldsymbol{E} = -\frac{\partial \boldsymbol{B}}{\partial t} \ , \quad \nabla \times \boldsymbol{B} = \mu \boldsymbol{J} \quad \text{and} \quad \nabla . \boldsymbol{B} = 0 \ , \tag{1}$$

and Ohm's law

$$\boldsymbol{J} = \sigma(\boldsymbol{E} + \boldsymbol{u} \times \boldsymbol{B}), \tag{2}$$

where $\boldsymbol{E}$ is the electric field and σ the electrical conductivity. The pre-Maxwell forms are used since the displacement current, $\partial \boldsymbol{E}/\partial t$, will be negligible for the relatively slow variations appropriate for the Earth. Assuming the electrical conductivity to be a constant, taking the curl of Equation (2) and applying the relations of (1) together with the vector identity

$$\nabla \times (\nabla \times \boldsymbol{V}) = \nabla(\nabla . \boldsymbol{V}) - \nabla^2 \boldsymbol{V},$$

gives the induction equation

$$\frac{\partial \boldsymbol{B}}{\partial t} = \nabla \times (\boldsymbol{u} \times \boldsymbol{B}) + \frac{1}{\mu_0 \sigma} \nabla^2 \boldsymbol{B}. \tag{3}$$

The generalised form of the induction equation, as used by the programs, is

$$c_k \frac{\partial \boldsymbol{B}}{\partial t} = c_m \nabla \times (\boldsymbol{u} \times \boldsymbol{B}) + c_l \nabla^2 \boldsymbol{B}. \tag{4}$$

We can write Equation (3) in the form

$$\frac{\partial \boldsymbol{B}}{\partial t} = \nabla \times (\boldsymbol{u} \times \boldsymbol{B}) + \eta \nabla^2 \boldsymbol{B}, \tag{5}$$

where $\eta = 1/(\mu_0 \sigma)$ is the magnetic diffusivity. If L, U, L^2/η and η^2/L^3 are respectively scales for length, velocity, time and the magnetic field strength (we are employing the diffusive time-scale) then we can write Equation (5) in the non-dimensional form

$$\frac{\partial \boldsymbol{B}}{\partial t} = R_m \nabla \times (\boldsymbol{u} \times \boldsymbol{B}) + \nabla^2 \boldsymbol{B}. \tag{6}$$

The magnetic Reynolds number, $R_m = (UL/\eta) = \mu_0 UL\sigma$, gives the ratio of the magnitude of the advective term, $\nabla \times (\boldsymbol{u} \times \boldsymbol{B})$, to the diffusive term, $\nabla^2 \boldsymbol{B}$.

We give the magnetic field, $\boldsymbol{B}$, an exponential time-dependence such that

$$\frac{\partial \boldsymbol{B}}{\partial t} = \sigma \boldsymbol{B}. \tag{7}$$

The growth rate, $\sigma = \sigma_{\mathrm{r}} + i\sigma_{\mathrm{i}}$, is in general complex. At $R_m = R_m^{\mathrm{c}}$, the real part of the growth rate, σ_{r}, is zero. σ_{i} is the oscillation frequency; the solution is steady if $\sigma_{\mathrm{i}} = 0$.

Substituting (7) into the induction equation (6) gives an eigenvalue problem of the form

$$\boldsymbol{A}\boldsymbol{x} = \sigma \boldsymbol{x}. \tag{8}$$

The vector $\boldsymbol{x}$ is a solution vector containing both the poloidal and toroidal magnetic field radial functions. The matrix $\boldsymbol{A}$ is the discretisation of the operator, $\nabla \times (\boldsymbol{u} \times \cdot) + \nabla^2 \cdot$. The eigenvalue problem (8) is solved using the Implicitly Restarted Arnoldi Method, or IRAM, ([Arn51],[LSY98], [Sor92]). The eigenvalues we are interested in are those with the largest real parts as these will control the stability. We apply a real shift, λ^{s}, to the system; subtracting $\lambda^{\mathrm{s}}\boldsymbol{x}$ from both sides of Equation (8) gives

$$\left(\boldsymbol{A} - \lambda^{\mathrm{s}}\boldsymbol{I}\right)\boldsymbol{x} = \left(\sigma - \lambda^{\mathrm{s}}\right)\boldsymbol{x}. \tag{9}$$

Rearranging to

$$\left(\boldsymbol{A} - \lambda^{\mathrm{s}}\boldsymbol{I}\right)^{-1}\boldsymbol{x} = \left(\sigma - \lambda^{\mathrm{s}}\right)^{-1}\boldsymbol{x} \tag{10}$$

gives us a new eigenproblem where the eigenvalue $(\sigma - \lambda^{\mathrm{s}})^{-1}$ will have a far higher magnitude than any others. ARPACK will locate this eigenvalue far more effectively than it would for the original formulation (8).

The program `krcmrnif` takes two values of R_m as inputs which must lie on either side of R_m^{c}. It then iterates towards a critical magnetic Reynolds number, modifying R_m in order to produce a value for σ_{r} which has a magnitude below a specified tolerance criterion, `DTOL`. The program is specific to the Kumar Roberts flow ([KR75]). $\boldsymbol{u}_{\mathrm{KR}}$ is defined as

$$\boldsymbol{u}_{\mathrm{KR}} = \nabla \times \nabla \times \left[\, {}^{P}_{\mathrm{KR}}v(r,t,\theta,\phi)\; \boldsymbol{r} \;\right] \;+\; \nabla \times \left[\, {}^{T}_{\mathrm{KR}}v(r,t,\theta,\phi)\; \boldsymbol{r} \;\right], \tag{11}$$

where the only non-zero radial functions for the velocity are

$${}^{T}_{\mathrm{KR}}v_1^{0c}(r) = \varepsilon_0 \left[r(1-r^2) + \Lambda r\right], \tag{12}$$

$${}^{P}_{\mathrm{KR}}v_2^{0c}(r) = \varepsilon_1 r^5 (1-r^2)^3, \tag{13}$$

$${}^{P}_{\mathrm{KR}}v_2^{2s}(r) = \varepsilon_2 r^3 (1-r^2)^2 \cos(pr) \tag{14}$$

and

$$ {}^{P}_{\rm KR}v_2^{2c}(r) = \varepsilon_3 r^3 (1-r^2)^2 \sin(pr). \qquad (15)$$

The geometry is a sphere with outer boundary at $r = 1$: this means $r_{\rm i} = 0$ and $r_{\rm o} = 1$.

The relative strengths of different components of the velocity are given by the parameters ε_0, ε_1, ε_2 and ε_3.

The stand-alone source code version of the program is compiled by typing

```
make krcmrnif
```

within this directory. Once the executable is created, begin execution by typing

```
krcmrnif < inputfile
```

The inputs file must have the following format.

```
* input file for krcmrnif
*
example_aOUTPUT                            : ROOT
0.0  1.0  10.0   4   20   7   0.001  : RI, RO, DRSV, NEV, NCV, NOITM, DTOL
*----------------------------------------------------------------------------
* NR LH ITRI ISF IOF ISP NBNO  E0   E1    E2    E3  PPAR  RM1      RM2
*----------------------------------------------------------------------------
* For the following truncation, the program krssgeps gave
*  Rmc = 3877.796
*----------------------------------------------------------------------------
  50  8  0    1   0   1   4  1.00  0.03  0.04  0.04  3   3860.0  3890.0
*----------------------------------------------------------------------------
* For the following truncation, the program krssgeps gave
*  Rmc = 3878.208
*----------------------------------------------------------------------------
  50 10  0    1   0   1   4  1.00  0.03  0.04  0.04  3   3860.0  3890.0
*----------------------------------------------------------------------------
* For the following truncation, the program krssgeps gave
*  Rmc = 3878.224
*----------------------------------------------------------------------------
  50 12  0    1   0   1   4  1.00  0.03  0.04  0.04  3   3860.0  3890.0
*----------------------------------------------------------------------------
```

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 following arguments are common to all of the runs carried out by the execution:

- `ROOT`: First characters in output files to be generated by current run.
- `RI`. Inner boundary radius: essentially always set to zero.
- `RO`. Outer boundary radius: essentially always set to 1.0.
- `DRSV`. The real shift - this is $\lambda_{\mathrm{r}}^{\mathrm{s}}$ in Equation (10).
- `NEV`. The number of eigenvalues requested.
- `NCV`. The length of the Arnoldi factorisation. See [LSY98]) for details.
- `NOITM`: The maximum number of iterations (values for R_m tried) permitted in order to find R_m^{c}.
- `DTOL`: Stopping criterion. $|\sigma_{\mathrm{r}}|$ must be less than `DTOL` for convergence to have occured.

All of the (uncommented) lines which follow describe a single run and contain the following variables:

- `NR`. Number of radial grid nodes.
- `LH`. Highest spherical harmonic degree, l requested.
- `ITRI`. Determines whether or not triangular truncation is used. The normal truncation of spherical harmonics is to include only terms with l up to including `LH`. This option is selected by `ITRI = 0`. If `ITRI = 1` then only radial functions with $(l + m)$ up to including `LH` are selected.
- `ISF`. Symmetry selection flag. This essential selects the seed field. Due to the symmetry of the flow (equations 12 to 15) there are four distinct symmetries of magnetic field which decouple. Each is determined by a poloidal magnetic seed field with a single spherical harmonic with degree l_{Seed} and order m_{Seed}. The options are:
  `ISF = 1`: $\rightarrow$ axial dipole. $l_{\mathrm{Seed}} = 1,\ m_{\mathrm{Seed}} = 0$.
  `ISF = 2`: $\rightarrow$ equatorial quadrupole. $l_{\mathrm{Seed}} = 2,\ m_{\mathrm{Seed}} = 0$.
  `ISF = 3`: $\rightarrow$ axial dipole. $l_{\mathrm{Seed}} = 1,\ m_{\mathrm{Seed}} = 1$.
  `ISF = 4`: $\rightarrow$ equatorial quadrupole. $l_{\mathrm{Seed}} = 2,\ m_{\mathrm{Seed}} = 1$.
  See ([Sar94]) and ([GZ93]) for details on symmetry.
- `IOF`. Output file flag. Options are:
  `IOF = 0`: $\rightarrow$ no output of eigenvectors.
  `IOF = 1`: $\rightarrow$ only output eigenfunctions corresponding to the eigenvalue with the largest real part.
  `IOF = 2`: $\rightarrow$ output all eigenfunctions.

Note that for `krcmrnif`, options `IOF = 1` and `IOF = 2` only ever produce a single eigenvector.

- `ISP`. Radial grid node spacings flag.
  `isp = 1` forces evenly spaced grid nodes from `ESNAAS` and `isp = 2` forces Chebyshev zero spaced nodes from `ZCPAAS`.
- `NBNO`. Number of bounding nodes for derivatives of velocity radial functions. Limited by the integer parameter `NBNOMX` in the source code. Whereas, for example, David Gubbins' code uses analytic expressions for the velocity radial functions and their derivatives, this code differentiates the functions numerically. The main reason for this approach was so that the subroutines used could handle general flows, e.g. steady flows locked by thermal boundary heating. However, it also has the advantage that (once thoroughly tested) any velocity may be applied without fear of making algebraic errors in the differentiation. The clear disadvantage is that it is not exact, although `NBNO` may be increased arbitrarily with little detriment to the program, other than the time required to compute these derivatives. Experiment to see what effect this has. `NBNO` must be atleast 3.
- `E0`. Parameter ε_0 in Equation (12).
- `E1`. Parameter ε_1 in Equation (13).
- `E2`. Parameter ε_2 in Equation (14).
- `E3`. Parameter ε_3 in Equation (15).
- `PPAR`. p/π where p is defined in Equations (14) and (15). `PPAR = 3.0` $\rightarrow p = 3\pi$.
- `RM1`. A value for the magnetic Reynolds number which is known to be below critical.
- `RM2`. A value for the magnetic Reynolds number which is known to be below critical.

1.1 Subprograms required for krcmrnif

SUBS subroutines

```
fopen.f krvhmf.f esnaas.f zcpaas.f svkrvf.f kdthsr.f
svfdcf.f ontppf.f gauwts.f schnla.f kdcmrn.f hmfwt.f
xarrwt.f svfwt.f fclose.f fnamer.f radvlf.f cntric.f
ldgnmf.f gfdcfd.f vcpcc.f matop.f rv0mfa.f amlp.f
```

```
mfseps.f evecex.f shvect.f vfcp.f  vf2qst.f invmft.f
vicexr.f innlca.f amdlt.f amlica.f amta.f  amsdea.f
vecop.f asvta.f asvcpl.f cubeop.f dvecz.f fftrlv.f
asvdr.f matind.f bmrcop.f powtwo.f
```

SUBS double precision function

```
emmult.f pmm.f   pmm1.f  plm.f   dpmm.f  dpmm1.f
dplm.f  sqrll1.f
```

SUBS integer function

```
indfun.f indshc.f
```

BLAS double precision function

```
dnrm2.f ddot.f  dasum.f
```

BLAS integer function

```
idamax.f
```

BLAS subroutines

```
daxpy.f dgemm.f dtrsm.f dgemv.f dswap.f dcopy.f
dger.f  dscal.f dtbsv.f dtrmm.f drot.f  dtrmv.f
```

ARPACK subroutines

```
dnaupd.f dneupd.f dnaup2.f dvout.f ivout.f second.f
dstatn.f dmout.f dgetv0.f dnaitr.f dnconv.f dneigh.f
dngets.f dnapps.f dlaqrb.f dsortc.f
```

LAPACK subroutines

```
dgetrf.f dgetri.f dgbtrf.f dgbtrs.f dgetf2.f dlaswp.f
xerbla.f dtrtri.f dgbtf2.f dlahqr.f dgeqr2.f dlacpy.f
dlaset.f dorm2r.f dtrevc.f dtrsen.f dtrti2.f dlabad.f
dlanv2.f dlarfg.f dlarf.f dlaln2.f dlacon.f dtrexc.f
dtrsyl.f dlarnv.f dlascl.f dlartg.f dlassq.f dladiv.f
dlaexc.f dlasy2.f dlaruv.f dlarfx.f
```

LAPACK double precision function

```
dlapy2.f dlamch.f dlanhs.f dlange.f
```

LAPACK integer function

```
ilaenv.f
```

LAPACK logical function

```
lsame.f
```

1.2 Run-time limitations

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

```
 INTEGER NRMAX, LHMAX, NHMAX, NPHMAX, NTHMAX, KLMAX,
1        NBNDMX, NDCS, NDRVM, ISVMAX, NPMAX, LHLH2M, NCFM,
2        NBN, NBNOMX, NRUNM, NCFMOM, NVMAX, IVELMX, NDCS0,
3        MAXNVI, NCVM
 PARAMETER ( NRMAX = 100, LHMAX = 20, MAXNVI = 30000, NRUNM = 50,
1            LHLH2M = LHMAX*(LHMAX+2), NBN = 2, NCVM = 25,
2            NHMAX = LHLH2M/2, NPHMAX = 64, NTHMAX = LHMAX + 2,
3            KLMAX = (NBN+1)*NHMAX-1, NBNDMX = 3*KLMAX+1 )
 PARAMETER ( NBNOMX = 5, NDCS = LHMAX + 1, NDRVM = 2,
1            NCFM = 2*NBN + 1, NCFMOM = 2*NBNOMX + 1,
2            ISVMAX = NRMAX*NHMAX, NVMAX = 4,
3            IVELMX = NVMAX*NRMAX, NDCS0 = 1,
4            NPMAX = (LHMAX+1)*(LHMAX+2)/2 )
```

If the values are insufficient, then change them and recompile. (Note that `NDRVM` and `NDCS0` are not size dependent and should not be changed.)

- `NRMAX` is the maximum permitted number of radial grid nodes.
- `LHMAX` is the highest permitted spherical harmonic degree, l.
- `MAXNVI` is the maximum number of spherical harmonic interactions. These are pre-calculated and stored in an array. There is no real way of knowing how many of these will be required without simply working them out. This number is therefore rather trial and error based: it clearly increases with `LH`. (This is probably quite an inefficient way of calculating non-linear interactions. Anybody is welcome to think up new ways. However, since the majority of time in this program is at the linear algebra stage, this aspect of the code never seemed worth examining.)
- `NRUNM` is the maximum number of independent runs permitted.
- `NBN` is the number of bounding nodes on either side of the finite difference stencil. `NBN = 2` is recommended value. Since the matrix describing the advective term couples distinct spherical harmonic radial functions (i.e. we are using the **IFORMF = 3** option, the bandwidth of the matrix soon becomes very large as the number of radial functions increases. `NBN = 3` therefore is a far higher computational cost than `NBN = 2`. (`NBN = 3` is necessary for the vorticity equation, for example in `linons1`, because of the fourth derivatives for the poloidal velocity. However, for the induction equation, `NBN = 2` gives fourth order accuracy.)

- `NPHMAX` is the maximum permitted number of grid nodes in ϕ for the Fast Fourier Transforms.
- `NBNOMX` is the upper limit for `NBNO`.
- `NVMAX` is the maximum number of velocity spherical harmonic radial functions. `NVMAX = 4` is quite adequate for the Kumar and Roberts flow (which has exactly 4 spherical harmonics). This will need to be changed if this program is to be modified to deal with other flows.

1.3 Outputs from KRCMRNIF

If the filename stem "root" was specified in the input file, the file `root.log` will be generated, along with any output solution vectors.

For each iteration of each run demanded from the input file, the following information is given in the file `root.log`:

```
---------------------------------------------
Iteration    2:   4 eigenvalues converged.
Rm value =         3890.0000000
---------------------------------------------
Eval  1 (     -35.7171267,       0.0000000) Res =        0.0000000
Eval  2 (       0.0229733,       0.0000000) Res =        0.0000000
Eval  3 (     -37.6180736,     -42.6454814) Res =        0.0000000
Eval  4 (     -37.6180736,      42.6454814) Res =        0.0000000
```

This simply returns the eigenvalues found along with their direct residuals. Once a critical magnetic Reynolds number has been found, the following is output:

```
RM=    3.8782252D+03 Real=    1.8658348D-06 Imag=    0.0000000D+00
ri:    0.0000000D+00 ro:    1.0000000D+00 nr:    50 isp:  1 nbn0:  4
lh:  12 itri:  0 Field sym: Axial_dipole
e0:    1.0000000D+00 e1:    3.0000000D-02 e2:    4.0000000D-02
e3:    4.0000000D-02 pp:    3.0000000D+00 rmc:    3.8782252D+03
osc:    0.0000000D+00
```

Here `RM` is the critical magnetic Reynolds number, accurate to whatever precision was specified by `DTOL`.

1.4 Sample runs of krcmrnif

The directory

```
$LEOPACK_DIR/SAMPLERUNS/KRCMRNIF
```

contains example input files and model output. Do not under any circumstances edit these files, as these examples should serve as a control for the correct working of the code. After compiling the program, copy the `.input` files to another directory, run the code and confirm that the output agrees with that in the directory.

1.4.1 Example a

The program `krssgeps` calculates critical magnetic Reynolds numbers using the generalised eigenvalue problem. It's results have been very nicely benchmarked against those of ([Sar94]) - and so we will test this code by simply reproducing results of `krssgeps`.

The input file has as comments the results obtained from `krssgeps` for these resolutions. Typing

```
grep RM example_aOUTPUT.log
```

reveals

```
RM=    3.8779671D+03 Real=    2.7827164D-06 Imag=    0.0000000D+00
RM=    3.8782084D+03 Real=    1.5640779D-06 Imag=    0.0000000D+00
RM=    3.8782252D+03 Real=    1.8658348D-06 Imag=    0.0000000D+00
```

which would appear to be in very good agreement.

References

[Arn51] W. E. Arnoldi. The principle of minimized iterations in the solution of the matrix eigenvalue problem. *Quart. J. Appl. Math.*, 9:17–29, 1951.

[GZ93] D. Gubbins and K. Zhang. Symmetry properties of the dynamo equations for paleomagnetism and geomagnetism. *Phys. Earth Planet. Inter.*, 75:225–241, 1993.

[KR75] S. Kumar and P. H. Roberts. A three-dimensional kinematic dynamo. *Proc. Roy. Soc. Lond. A*, 344:235–258, 1975.

[LSY98] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *Arpack users guide: Solution of large scale eigenvalue problems by implicitly restarted Arnoldi methods.* SIAM, 1998.

[Sar94] G. R. Sarson. *Kinematic Dynamo Calculations for Geomagnetism.* PhD thesis, University of Leeds, 1994.

[Sor92] D. C. Sorensen. Implicit application of polynomial filters in a k-step Arnoldi method. *SIAM J. Matrix Analysis and Applications*, 13:357–385, 1992.