A User's Manual for grpfs/grafs

Stanislav Hledík

`$Id: grpfs.tex,v 1.2 2009/10/20 08:45:03 lvs Exp $`

Abstract

grpfs/grafs numerically solves the free boundary problem that describes internal structure of general relativistic polytropic/adiabatic fluid spheres with a non-zero cosmological constant.

This document serves as an introductory manual for the grpfs/grafs package only. Neither plotting programs used for creating graphical presentation, nor the package noutils used for postprocessing the output are covered—in this point the user is referred to the documentation cited hereafter.

Contents

1 Introduction

grpfs/grafs is a command-line driven program that calculates the internal structure of general relativistic polytropic/adiabatic fluid spheres (from which the program name has been derived). Based on Tooper's papers [Tooper, 1964, Tooper, 1965], it implements the effect of a non-zero cosmological constant as a contribution to the physical facet and more reliable and effective numerical methods as a contribution to the computational facet of the problem.

Whether the program deals with polytropic or adiabatic case, depends on the way it is invoked. When invoked as grpfs, the polytropic case will be solved, when invoked as grafs, the adiabatic case will be solved. Because both cases are similar from the computational point of view, we will not distinguish between them if not stated otherwise, calling the program simply grpfs throughout the manual. The same applies for whole the package as well. The differences between grpfs and grafs are listed in Appendix A.

Together with standard Unix[1] utilities like *awk* or *sed* and with its redirection/piping facility, the program is pretty flexible. For these purposes one can use the noutils package [Ref].

Section 2 gives some information on how the program can be obtained, no warranty and installation guidelines follow in Sections 3 and 4. Although it is beyond the scope of this manual to deal with the underlying principles of the problem, one can find it useful to get some idea about the mathematics and numerics the program has to cope with. This is also necessary for fully appreciating all the features and helpful when trying to fix problems. Therefore, the basic underlying math is briefly discussed in Section 5. The description of usage includes Section 6 on basic tasks and Section 7 on dealing with advanced features. Section 8 gives an idea how to customize the program to match the user's needs; Section 9 mentions some potentially buggy aspects. Finally, Appendix A lists the differences between grpfs and grafs, and Appendix B presents the full program code.

2 Obtaining grpfs

The package grpfs can be obtained from the Computational Physics Laboratory homepage `http://uf.fpf.slu.cz/cpl/`. Since the source code contain routines from [Press et al., 1997] (see Appendix B for more info) that are subject to copyright to Numerical Recipes Software that forbids to distribute them on a free basis, the site is password-protected. In order to obtain the password, please contact the author at Email address `Stanislav.Hledik@fpf.slu.cz`.

1. Unix is a registered trademark of Unix Systems Laboratories.

The source code can be modified and distriubuted without prior author's permission, with the exception of the module `grp_node.c` (containing routines from [Press et al., 1997]) that can *solely* be used for studying purposes.

The preferred platform for the noutils package is a GNU Unix system. However, this package is fully portable on most Unix platforms.

3 No Warranty

The software described in this manual has *no warranty*, it is provided 'as is.' It is your responsibility to validate the behavior of the program and its ability to perform the task required using the source code provided.

4 Installation

All sources are distributed packed in the tarball `grpfs.tar.gz`. Download that file and save it in a working directory of your choice. Once downloaded, `cd` to this directory and use the command (the leading `$` represents the shell prompt hereafter)

```
$ gzip -dc grpfs | tar xvvf - ; cd grpfs
```

(On a GNU Unix system, the simpler command

```
$ tar xzvvf grpfs ; cd grpfs
```

also works.) This unpacks all files into the directory `grpfs` and moves you to it. Then edit subsection 'Installation' of the 'Customization section' in the file `Makefile` to match your needs. Unless you are a superuser, you will probably want to install into your `~/bin` directory; in such case there is no need to customize `Makefile` at all. Superusers should change the destination directory to another one, preferably to `/usr/local/bin`.

Once `Makefile` is customized, simply type

```
$ make
```

to compile, and

```
$ make install
```

to install the package. (The first command may be omitted, is this case it is performed automatically.)

In order to get rid of backup, object, log and similar files, use

```
$ make clean
```

The command

```
$ make veryclean
```

makes the same as the previous one, in addition, it deletes the executable files (or symbolic links pointing at them) in package directory, leaving it in original state. To uninstall, run

```
    $ make uninstall
```

It is recommended to install the package noutils [Ref] as well if you have not installed it yet.

If you are going to install on a system without the `make` program (like, e.g., M$ DOG), look into the `Makefile` and follow the compilation steps described therein.

5 Equations

The program grpfs starts its performance by numerical integration the initial value problem for the following set of six coupled first-order ordinary differential equations

$$\frac{\mathrm{d}\theta}{\mathrm{d}\xi} = \frac{\left(\frac{2}{3}\lambda\xi^3 - \sigma\xi^3\theta^{n+1} - v\right)(1+\sigma\theta)}{\xi^2}\, g_{rr}(\xi, v; n, \sigma, \lambda) \tag{1}$$

$$\frac{\mathrm{d}v}{\mathrm{d}\xi} = \xi^2\theta^n \tag{2}$$

$$\frac{\mathrm{d}\bar{\xi}}{\mathrm{d}\xi} = \sqrt{g_{rr}(\xi, v; n, \sigma, \lambda)} \tag{3}$$

$$\frac{\mathrm{d}e_0}{\mathrm{d}\xi} = \xi^2\theta^n\sqrt{g_{rr}(\xi, v; n, \sigma, \lambda)} \tag{4}$$

$$\frac{\mathrm{d}e_{0\mathrm{g}}}{\mathrm{d}\xi} = \frac{\xi^2\theta^n\sqrt{g_{rr}(\xi, v; n, \sigma, \lambda)}}{(1+\sigma\theta)^n} \tag{5}$$

$$\frac{\mathrm{d}z}{\mathrm{d}\xi} = \sqrt{g_{rr}(\xi, v; n, \sigma, \lambda) - 1} \tag{6}$$

where

$$g_{rr}(\xi, v; n, \sigma, \lambda) \equiv \frac{1}{1 - 2\sigma(n+1)\left(\frac{v}{\xi} + \frac{1}{3}\lambda\xi^2\right)} \tag{7}$$

for six functions $\theta(\xi)$, $v(\xi)$, $\bar{\xi}(\xi)$, $e_0(\xi)$, $e_{0\mathrm{g}}(\xi)$, and $z(\xi)$ subject to initial conditions

$$\theta(0) = 1 \qquad \bar{\xi}(0) = v(0) = e_0(0) = e_{0\mathrm{g}}(0) = z(0) = 0 \tag{8}$$

on an a priori unknown interval $0 \leq \xi \leq \xi_1$ with the right end ξ_1 being the first positive root of the equation[2]

$$\theta(\xi) = 0 \tag{9}$$

2. From this point of view, the problem should be termed as *free boundary problem* instead (see [Press et al., 1997]).

The parameters n (polytropic index), σ (relativity parameter), λ (cosmological parameter) may take values from the regions $\langle 0, \infty)$, $\langle 0, \infty)$, and $(-\infty, \lambda_{\max})$, respectively, where $\lambda_{\max}$ depends on n and σ. (This dependence is beyond the scope of this manual, however.) The limit $\sigma \to 0$ corresponds to Newtonian fluid spheres.

It can be shown [Tooper, 1964, Tooper, 1965] that as far as the parameter λ is small enough to keep the function (7) positive, the function $\theta(\xi)$ is decreasing, dropping to its first zero at $\xi = \xi_1$. Consequently, further task is to find the root ξ_1 and the corresponding values of the remaining five functions $v(\xi_1)$, $\bar{\xi}(\xi_1)$, $e_0(\xi_1)$, $e_{0\mathrm{g}}(\xi_1)$, and $z(\xi_1)$. Finding the root ξ_1 requires, of course, some user-supplied first estimate ξ_0; if it turns out to be too large or too small, the integration must resume with a better first estimate ξ_0.

All the quantities that the program calculates with are dimensionless. The true scaling can be obtained by multiplying the output with appropriate scaling factor for length, mass, energy, mass density, energy density, pressure, and velocity

$$\mathscr{L}^* = \sqrt{\frac{\sigma(n+1)c^2}{4\pi G\rho_\mathrm{c}}} \tag{10}$$

$$\mathscr{M}^* = 4\pi\mathscr{L}^{*3}\rho_\mathrm{c} = \frac{c^2}{G}\sigma(n+1)\mathscr{L}^* \tag{11}$$

$$\mathscr{E}^* = \mathscr{M}^*c^2 = \frac{c^4}{G}\sigma(n+1)\mathscr{L}^* \tag{12}$$

$$\mu^* = \rho_\mathrm{c} \tag{13}$$

$$\epsilon^* = \mu^*c^2 = \rho_\mathrm{c}c^2 \tag{14}$$

$$\pi^* = \sigma\mu^*c^2 = \sigma\rho_\mathrm{c}c^2 \tag{15}$$

$$c^* = c = 2.99792458 \times 10^{10}\,\mathrm{cm/s} \tag{16}$$

where (see [Misner et al., 1973])

$$c^2 = 0.89875518 \times 10^{21}\,\mathrm{cm^2/s^2} \tag{17}$$

$$\frac{c^2}{G} = 1.3469 \times 10^{28}\,\mathrm{g/cm} \tag{18}$$

$$\frac{c^4}{G} = 1.2106 \times 10^{49}\,\mathrm{g\,cm/s^2} \tag{19}$$

As one can easily see, to calculate true scales it is necessary to know, besides n, σ, and G/c^2, the central mass density ρ_c.

There is another differential equation to be integrated,

$$\begin{aligned}\frac{\mathrm{d}\tilde{z}}{\mathrm{d}\xi} &= (1+\sigma\theta)^n\left[2\sigma(n+1)g_{rr}(\xi_1, v_1; n, \sigma, \lambda)g_{rr}(\xi, v; n, \sigma, \lambda)\right]^{1/2} \\ &\quad \times Z^{1/2}(\xi, v; \xi_1, v_1; n, \sigma, \lambda)\end{aligned} \tag{20}$$

where

$$Z(\xi, v; \xi_1, v_1; n, \sigma, \lambda) = \left[1 + \sigma\left(\theta + (n+1)\xi\frac{\mathrm{d}\theta}{\mathrm{d}\xi}\right)\right]\left(\frac{v}{\xi} + \frac{1}{3}\lambda\xi^2\right)$$

$$-\xi\frac{\mathrm{d}\theta}{\mathrm{d}\xi}\left[1+\sigma\theta+\frac{1}{2}\sigma(n+1)\xi\frac{\mathrm{d}\theta}{\mathrm{d}\xi}\right] \tag{21}$$

on an interval (unification of intervals) determined by condition of optical-geometry embeddability

$$Z(\xi,v;\xi_1,v_1;n,\sigma,\lambda)\geq 0 \tag{22}$$

with initial conditions

$$\begin{aligned}\tilde{z}(\xi_{\mathrm{i},1}) &= 0\\ \tilde{z}(\xi_{\mathrm{i},2}) &= \tilde{z}(\xi_{\mathrm{f},1})\\ &\vdots\\ \tilde{z}(\xi_{\mathrm{i},n}) &= \tilde{z}(\xi_{\mathrm{f},n-1})\end{aligned} \tag{23}$$

where the indices 'i' and 'f' refer to initial and final boundary of the intervals in which (22) is satisfied.

Equation (20) has to be treated separately from the coupled set (1)–(6). The reason is based on the fact that the right hand side of Eq. (20) depends on the first zero point ξ_1 determined by Eq. (9) in a non-factorizable way, i.e., it cannot be decomposed as $s(\xi_1)\times($*the rest of rhs (20) not containing* $\xi_1)$, which would allow appending the Eq. (20) to the set (1)–(6) without the constant factor $s(\xi_1)$ and scaling the results with $s(\xi_1)$ afterwards. As the numerical integration requires evaluation of the right-hand side at an arbitrary point, an interpolation between the tabulated values of the solution of Eqs (1)–(6) must take place.

The most straightforward way how to implement the restriction imposed by limit (22) is redefining the function (21) as

$$Z\leftarrow Z_{+}\equiv\begin{cases}Z & \text{if}\quad Z\geq 0\\ 0 & \text{if}\quad Z<0\end{cases} \tag{24}$$

which guarantees satisfying the initial conditions (22) automatically.

The optical geometry embedding diagram is then given parametrically in cylindrical coordinates $(\chi,\alpha,\tilde{z})$ as $(\chi(\xi),\tilde{z}(\xi))$ with the dimensionless radius ξ being the parameter, and

$$\chi(\xi)=\xi(-g_{tt})^{-1/2}=\xi(1+\sigma\theta)^{n+1}\left\{1-2\sigma(n+1)\left[\frac{v_1}{\xi_1}+\frac{1}{3}\lambda\xi_1^2\right]\right\}^{-1/2} \tag{25}$$

6 Basic running grpfs

This Section serves as the core tutorial for grpfs. We shall focus on running grpfs solely, with no cooperating utilities, as are, e.g., noutils [Ref]. This topic is covered in Section 7.

6.1 Invocation

The basic invocation of the program grpfs is

```
$ grpfs ⟨options[modifiers]⟩
```

where recognized ⟨*options*⟩ are '`-i`', '`-b`', '`-f`', '`-a`', and '`-x`', legal *[modifiers]* for the '`-a`' options are '`K`', '`B`', '`R`', and '`S`'; the remaining options have no modifiers. Options can be given in arbitrary order and can be grouped. For example, the following commands lead to identical invocations:

```
$ grpfs -i -aK
$ grpfs -aK -i
$ grpfs -iaK
$ grpfs -aKi
```

However, notice that the modifier ('`K`' in the example) must immediately follow the option to be modified. Therefore,

```
$ grpfs -aiK
```

is wrong, because the modifier '`K`' is separated from the base option '`-a`' by the option '`-i`'. The options and modifiers will be fully described in the following Subsections in a free narrative way.

If an unrecognized option or modifier is given, as in

```
$ grpfs -iak
```

the program issues a message like

```
grpfs: k: unrecognized option, exiting to system
```

and exits. This example also illustrates that the options and modifiers are case-sensitive.

6.2 Input

Let us launch grpfs in the simplest way, with no option:

```
$ grpfs
```

Program will start in interactive mode, displaying the following text:[3]

```
This is grpfs, Revision 1.6 <2002-02-08 19:53:11+01> by S. Hledik
ODE solvers based on (C) Numerical Recipes Software.
Fixed stepsize 4th-ord. Runge-Kutta, double precision.
Please feed in the quantities asked for below:
  ---------------------------------------------------------------
  Quantity             Numer. type     Input/Status    Echo check
  ---------------------------------------------------------------
  (r)hoc [10^5 g/cm^3]  real,if>0:true
```

3. The revision number, date and time or other items in the banner line may differ from the actual program run.

The method that grpfs is going to exploit for numerical integration is fixed stepsize 4th-order Runge–Kutta. This is the default, but explicitly using of this method can be required by typing

```
    $ grpfs -f
```

where the option '`-f`' stands for 'fixed stepsize.' Similarly, interactive run might be explicitly specified by the option '`-i`', which is the default. Thus, invocation

```
    $ grpfs -if
```

works in exactly the same way as the previous ones. These options might seem to be superfluous, however, they can help the user to keep in mind what he/she is actually doing.

Now the program is expecting you to enter the central mass density ρ_c (which is intrinsically positive number) in the units indicated. If you enter zero (`0`, `0.`, `.0`, `0.0`, `.`), the response looks like this:

```
---------------------------------------------------------------
(r)hoc [10^5 g/cm^3]  real,if>0:true  0
                                      Dimensionless output opted
                                      no scales calculated
                                      Ignored         --
---------------------------------------------------------------
```

Supplying zero for the central mass density (which cannot represent a valid value), the program is not able calculate the true scales (10)–(15); hence, both the output will be in dimensionless quantities and the scaling factors (10)–(15) will not be provided. If you enter negative central mass density (which also cannot represent a valid value), the program accepts the absolute value as the central mass density, and the minus sign is understood as a command 'do calculate and show the scaling factors (10)–(15), but do *not* multiply the output with them, leave the output dimensionless instead.' In such case, the response looks like this:

```
---------------------------------------------------------------
(r)hoc [10^5 g/cm^3]  real,if>0:true  -1.0
                                      Dimensionless output opted
                                      but scales provided
                                      Accepted |r|    1.000E+00
---------------------------------------------------------------
```

Finally, if you enter a positive value, it will be accepted without changes and the dimensional output quantities will be multiplied by appropriate scale factors. In this case, the response looks like this:

```
---------------------------------------------------------------
(r)hoc [10^5 g/cm^3]  real,if>0:true  1.0
                                      True scale on output opted
                                      Accepted        1.000E+00
---------------------------------------------------------------
```

The second entry is the polytropic index $n \geq 0$. The input and grpfs's response looks like this:

```
-----------------------------------------------------------------
(p)olytropic index    real >= 0.0     1.0
                                      Accepted        1.000
-----------------------------------------------------------------
```

If you enter a value that is in conflict with the specification in the column '`Numer. type`', the program will response (and the user may correct his/her mistake) like this:

```
-----------------------------------------------------------------
(p)olytropic index    real >= 0.0     -.2
                                      Bad, retry:
                                      1.0
                                      Accepted        1.000
-----------------------------------------------------------------
```

Similarly, entering a wrong value at any prompt, it will be rejected and the user will be prompted to provide a correct value. This protection might be bypassed by using the batchmode (see Subsection 7.1), but the results would be nonsensical.

Now, assume that we entered $\rho_{\rm c} = 0$ at the first prompt. Then the rest of interaction grpfs–the user is self-explanatory, as the following example shows:

```
-----------------------------------------------------------------
(s)igma               real >= 0.0     .2
                                      Accepted        2.000E-01
-----------------------------------------------------------------
(l)ambda              real            1.0E-34
                                      Accepted        1.000E-34
-----------------------------------------------------------------
(f)inal xi guess      real > 0.0      4
                                      Accepted        4.000
-----------------------------------------------------------------
# of s(t)eps          integer > 2     25
                                      Accepted        25
-----------------------------------------------------------------
```

Here we entered σ, the dimensionless 'cosmological parameter' $\lambda = \rho_{\rm vac}/\rho_{\rm c}$, the estimate of the first positive root of (9), and the number of steps. Remember: the points at which the functions $\theta(\xi)$, $v(\xi)$, $\bar{\xi}(\xi)$, $e_0(\xi)$, $e_{0\rm g}(\xi)$, $z(\xi)$, and $\tilde{z}(\xi)$ will be evaluated, are enumerated $0, 1, \ldots, \langle$*number of steps*$\rangle$, their number being $\langle$*number of steps*$\rangle + 1$.

If a non-zero $\rho_{\rm c}$ (either positive or negative) has been entered at the first prompt, the prompting for the 'cosmological parameter' and the response that follows will be modified like this:

```
-----------------------------------------------------------------
(l|L)ambda [1|cm^-2]  real | *real    1.0E-34
                                      Accepted (l)    1.000E-34
-----------------------------------------------------------------
```

That means if an 'ordinary' number is entered as in the example above, the programs accepts it as 'cosmological parameter' λ and, for the user's information, will calculate the cosmological constant

$$\Lambda = \frac{8\pi G}{c^2}\rho_c\lambda \tag{26}$$

in units of cm^{-2}. On the other hand, if the number entered is preceded by a star '`*`' (no space between), the number will be accepted as the cosmological constant Λ (in units of cm^{-2}) and the 'cosmological parameter' λ will be calculated for the program's (and user's, of course) needs:

```
-----------------------------------------------------------------
(l|L)ambda [1|cm^-2]  real | *real    *1.1E-56
                                      Accepted (L)    1.100E-56
-----------------------------------------------------------------
```

Notice that if ρ_c is not given, the relationship (26) is lacking any sense, which grpfs is aware of.

Now let us try another invocation (the option '`-i`' may be omitted):

```
$ grpfs -iaK
```

The program writes out the following banner

```
This is grpfs, Revision 1.6 <2002-02-08 19:53:11+01> by S. Hledik
ODE solvers based on (C) Numerical Recipes Software.
Adaptive stepsize 5th-ord. Cash-Karp Runge-Kutta, double precision.
Please feed in the quantities asked for below:
```

indicating that now we have opted the adaptive stepsize 5th-order Cash–Karp Runge–Kutta method (see [Press et al., 1997]). The '`K`' modifier stands for 'Kutta' and is the default modifier, i.e., launching

```
$ grpfs -ia
```

behaves as if the '`K`' modifier were present. However, it is reasonable to specify the modifier explicitly.

The '`-a`' option is of higher precedence than '`-f`' option, i.e., if both these options would be given in any order, the program would behave as if only '`-a`' were specified.

The prompts and behavior up to the final ξ guess are identical, then the prompting and responses become different, as shown in the following example:

```
-----------------------------------------------------------------
(a)ccuracy            real > 2.3E-16  .000001
                                      Accepted        1.0E-06
-----------------------------------------------------------------
1st stepsize (g)uess  0.0<real<f      .1
                                      Accepted        1.0E-01
-----------------------------------------------------------------
ma(x) points to save  int 1, or >3    25
                                      Accepted        25
-----------------------------------------------------------------
(m)in sav. interval   real,if<0:|f/m| -25
                                      Acc|4.0/-25|    1.6E-01
-----------------------------------------------------------------
```

The user is asked for to supply the accuracy desired, then the first stepsize estimate, and maximum number of points (*not* steps) to save, and the minimum saving interval. If the last value is negative, the absolute value of (final ξ guess)/(minimum saving interval) will be used instead.

Replacing the '`K`' modifier with '`B`', '`R`', or '`S`' indicates using the adaptive stepsize Bulirsch–Stoer, 4th-order Rosenbrock, and Bulirsch–Stoer semi-implicit midpoint rule method, respectively (see [Press et al., 1997]). The modifier precedence can be symbolically expressed as `K` < `B` < `R` < `S`. The structure of input remains exactly the same for all these modifiers.

6.3 Output

General description of the output format can be found in [Ref]. Here we only summarize its main features and give more detailed account of some extras thrown in particularly for grpfs.

The output is separated into commentary part and the output data itself. Each line of the former part is preceded by comment sign '`#`', while the latter part (tabulated output data) are formatted in c space-separated columns.

The commentary part comprises the following sections.

`#M`m This line at the very beginning of the output marks the method used. The digit m takes value 0 for fixed stepsize method (option '`f`' or none) and 1 for adaptive stepsize methods (option '`a`' with no modifier or with modifiers '`K`', '`B`', '`R`', or '`S`').

`#ID_CHART` This sections accommodates various technical details concerning date and time, means of invocation, method used, host name and IP address, code version, compiler options and many others. the purpose of this section is to identify the run as exactly as possible. Because the actual contents may change, we shall not give it here explicitly. This section is omitted unless preprocessor option '`-DVERBOSE`' is set.

`#INPUT` In this section, the input data are echoed. Notice that the numbers are commented out by strings '`#!`' instead of '`#`'; see subsection 7.1 and manual [Ref] for explanation.

`#COMPUTATION_LOG` This section gives information about solving ODE's and auxiliary numerical routines (such as interpolation) calls. This section is omitted unless preprocessor option '`-DVERBOSE`' is set.

`#OUTPUT` ℓ This string starts tabulated output consisting of ℓ rows of data.

The sections described above are separated by single comment lines containing the comment sign '`#`' only.

The columns of tabulated output are described in the following. The first column shows the output row number. The term 'dimensionless' refers to quantities that can be scaled by the scaling factors (10)–(16), while the term 'relative' refers either to quantities that range from 0 through 1 since they are divided by the corresponding value on the surface or total quantity (at the point ξ_1), or to quantities related to another quantities (typically energies or energy densities). The value of a language C symbolic constant `NVAR` shows the number of Eqs (1)–(6) ($\texttt{NVAR} = 2, 3, \ldots, 6$) and (20) ($\texttt{NVAR} = 7$) that have to be taken into account for the particular column to be computed (it is padded with zero otherwise). The boldface **const** reminds that all rows are of constant value.

C1 Output row number, starting from 0 for fixed stepsize methods (option '`-f`') and from 1 for adaptive stepsize methods (option '`-a`'). [$\texttt{NVAR} \geq 2$]

C2 Dimensionless radius ξ (independent variable). To be multiplied by length scale factor (10) to get true radius, $r = \mathscr{L}^*\xi$. [$\texttt{NVAR} \geq 2$, stored in array `xp[]`]

C3 Total dimensionless radius ξ_1. To be multiplied by length scale factor (10) to get true total radius, $R = \mathscr{L}^*\xi_1$. [$\texttt{NVAR} \geq 2$, **const**]

C4 Relative radius $\xi/\xi_1 \in [0, 1]$. [$\texttt{NVAR} \geq 2$]

C5 If $\texttt{LOGDLESSDENS} = 0$, then $\sigma(n+1)\xi^2$ [$\texttt{NVAR} \geq 2$]. If $\texttt{LOGDLESSDENS} = 1$, then $\log_{10} \sigma(n+1)\xi_1^2$ [$\texttt{NVAR} \geq 2$, **const**].

C6 Dimensionless proper radius $\bar{\xi}$ governed by Eq. (3). To be multiplied by length scale factor (10) to get true proper radius, $\bar{r} = \mathscr{L}^*\bar{\xi}$. [$\texttt{NVAR} \geq 3$, stored in array `yp[3][]`]

C7 Total dimensionless proper radius $\bar{\xi}_1$. To be multiplied by length scale factor (10) to get total true proper radius, $\bar{R} = \mathscr{L}^*\bar{\xi}_1$. [$\texttt{NVAR} \geq 3$, **const**]

C8 Relative proper radius $\bar{\xi}/\bar{\xi}_1 \in [0, 1]$. [$\texttt{NVAR} \geq 3$]

C9 If $\texttt{LOGDLESSDENS} = 0$, then $\sigma(n+1)\bar{\xi}^2$ [$\texttt{NVAR} \geq 3$]. If $\texttt{LOGDLESSDENS} = 1$, then $\log_{10} \sigma(n+1)\bar{\xi}_1^2$ [$\texttt{NVAR} \geq 3$, **const**].

C10 Dimensionless mass parameter θ governed by Eq. (1), $\theta_1 \equiv \theta(\xi_1) = 0$. See next column. [$\texttt{NVAR} \geq 2$, stored in array `yp[1][]`]

C11 Dimensionless mass density θ^n, $\theta_1^n = 0$. To be multiplied by mass density scale factor (13) to get true mass density, $\rho = \mu^*\theta^n$. [$\texttt{NVAR} \geq 2$]

C12 Dimensionless pressure θ^{n+1}, $\theta_1^{n+1} = 0$. To be multiplied by pressure scale factor (15) to get true pressure, $p = \pi^*\theta^{n+1}$. [$\texttt{NVAR} \geq 2$]

C13 Dimensionless proper mass density of the gas $\theta_{\text{gas}} = [\theta/(1+\sigma\theta)]^n$, $\theta_{\text{gas},1} \equiv \theta_{\text{gas}}(\xi_1) = 0$. To be multiplied by mass density scale factor (13) to get true proper mass density of the gas, $\rho_{\text{gas}} = \mu^*\theta_{\text{gas}}$. [$\texttt{NVAR} \geq 2$]

C14 Dimensionless velocity of sound $c_{\text{s}} = [\frac{n+1}{n}\sigma\theta]^{1/2}$, $c_{\text{s},1} \equiv c_s(\xi_1) = 0$. To be multiplied by velocity scale factor (16) to get true velocity of sound,

$v_\mathrm{s} = c^* c_\mathrm{s}$. The true velocity is not implemented, since the velocity scale factor (16) is ρ_c-independent. [`NVAR` ≥ 2]

C15 Dimensionless mass v inside dimensionless radius ξ governed by Eq. (2). To be multiplied by mass scale factor (11) to get true mass $m = \mathscr{M}^* v$ inside true radius $r = \mathscr{L}^* \xi$. [`NVAR` ≥ 2, stored in array `yp[2][]`]

C16 Total dimensionless mass v_1 inside total dimensionless radius ξ_1. To be multiplied by mass scale factor (11) to get true total mass $M = \mathscr{M}^* v_1$ inside total true radius $R = \mathscr{L}^* \xi_1$. [`NVAR` ≥ 2, **const**]

C17 Dimensionless gravitational radius $\xi_\mathrm{grav} = 2\sigma(n+1)v$ at dimensionless radius ξ. To be multiplied by length scale factor (10) to get true gravitational radius $r_\mathrm{grav} = \mathscr{L}^* \xi_\mathrm{grav}$ at true radius r. [`NVAR` ≥ 2]

C18 Dimensionless gravitational radius at the surface $\xi_\mathrm{grav,1} = 2\sigma(n+1)v_1$. To be multiplied by length scale factor (10) to get true gravitational radius at the surface, $R_\mathrm{grav} = \mathscr{L}^* \xi_\mathrm{grav,1}$. [`NVAR` ≥ 2]

C19 Dimensionless Newtonian potential $\Phi_\mathrm{N} = \sigma(n+1)v/\xi = \xi_\mathrm{grav}/2\xi = Gm/c^2 r$ at dimensionless radius ξ. [`NVAR` ≥ 2]

C20 Dimensionless Newtonian potential $\Phi_\mathrm{N,1} = \sigma(n+1)v_1/\xi_1 = \xi_\mathrm{grav,1}/2\xi_1 = GM/c^2 R$ at the surface. [`NVAR` ≥ 2, **const**]

C21 Dimensionless proper Newtonian potential $\bar{\Phi}_\mathrm{N} = \sigma(n+1)v/\bar{\xi} = \xi_\mathrm{grav}/2\bar{\xi} = Gm/c^2 \bar{r}$ at dimensionless proper radius $\bar{\xi}$. [`NVAR` ≥ 3]

C22 Dimensionless proper Newtonian potential $\bar{\Phi}_\mathrm{N,1} = \sigma(n+1)v_1/\bar{\xi}_1 = \xi_\mathrm{grav,1}/2\bar{\xi}_1 = GM/c^2 \bar{R}$ at the surface. [`NVAR` ≥ 3, **const**]

C23 Density concentration factor $\rho_\mathrm{c}/\langle\rho\rangle = \xi^3/3v$ up to dimensionless radius ξ. [`NVAR` ≥ 2]

C24 Total density concentration factor $\rho_\mathrm{c}/\langle\rho\rangle_1 = \xi_1^3/3v_1$ (up to total dimensionless radius ξ_1). [`NVAR` ≥ 2, **const**]

C25 Spatial metric coefficient $g_{rr} = [1-2\sigma(n+1)(\frac{v}{\xi}+\frac{1}{3}\lambda\xi^2)]^{-1}$ at dimensionless radius ξ. [`NVAR` ≥ 2]

C26 Spatial metric coefficient $g_{rr,1} = [1-2\sigma(n+1)(\frac{v_1}{\xi_1}+\frac{1}{3}\lambda\xi_1^2)]^{-1}$ at the surface ($\xi = \xi_1$). [`NVAR` ≥ 2, **const**]

C27 Time metric coefficient $-g_{tt} = [(1+\sigma\theta)^{2(n+1)} g_{rr,1}]^{-1}$ at dimensionless radius ξ. To be multiplied by velocity scale factor (16) squared to get true time metric coefficient at true radius r. The true time metric coefficient is not implemented, since the velocity scale factor (16) is ρ_c-independent. [`NVAR` ≥ 2]

C28 Time metric coefficient $-g_{tt,1} = (g_{rr,1})^{-1}$ at the surface ξ_1 (because $\theta_1 \equiv \theta(\xi_1) = 0$). To be multiplied by velocity scale factor (16) squared to get true time metric coefficient at the surface. The true time metric coefficient at the surface is not implemented, since the velocity scale factor (16) is ρ_c-independent. [`NVAR` ≥ 2, **const**]

C29 Dimensionless proper energy density $\epsilon_0 = \theta^n \sqrt{g_{rr}}$ at dimensionless radius ξ. To be multiplied by energy density scale factor (14) to get true proper energy density $\mathscr{E}_0 = \epsilon^* \epsilon_0$ at radius r. At the surface the value vanishes. [`NVAR` ≥ 2]

C30 Dimensionless proper rest energy density $\epsilon_{0\mathrm{g}} = [\theta/(1+\sigma\theta)]^n \sqrt{g_{rr}}$ at dimensionless radius ξ. To be multiplied by energy density scale factor (14) to get true proper rest energy density $\mathscr{E}_{0\mathrm{g}} = \epsilon^* \epsilon_{0\mathrm{g}}$ at radius r. At the surface the value vanishes. [`NVAR` ≥ 2]

C31 Dimensionless difference $\epsilon_0 - \theta^n = \theta^n(\sqrt{g_{rr}} - 1)$ at dimensionless radius ξ. To be multiplied by energy density scale factor (14) to get true difference $\mathscr{E}_0 - \rho c^2 = \epsilon^*(\epsilon_0 - \theta^n)$ at radius r. At the surface the value vanishes. [`NVAR` ≥ 2]

C32 Relative difference $(\epsilon_0 - \theta^n)/\theta^n = \sqrt{g_{rr}} - 1$ at dimensionless radius ξ. [`NVAR` ≥ 2]

C33 Limit of relative difference $\lim_{\xi \to \xi_{1-}} [(\epsilon_0 - \theta^n)/\theta^n] = \sqrt{g_{rr,1}} - 1$, approaching the surface from below. [`NVAR` ≥ 2, **const**]

C34 Dimensionless difference $\epsilon_{0\mathrm{g}} - \theta^n = \theta^n[\sqrt{g_{rr}}/(1+\sigma\theta)^n - 1]$ at dimensionless radius ξ. To be multiplied by energy density scale factor (14) to get true difference $\mathscr{E}_{0\mathrm{g}} - \rho c^2 = \epsilon^*(\epsilon_{0\mathrm{g}} - \theta^n)$ at radius r. At the surface the value vanishes. [`NVAR` ≥ 2]

C35 Relative difference $(\epsilon_{0\mathrm{g}} - \theta^n)/\theta^n = \sqrt{g_{rr}}/(1+\sigma\theta)^n - 1$ at dimensionless radius ξ. [`NVAR` ≥ 2]

C36 Limit of relative difference $\lim_{\xi \to \xi_{1-}} [(\epsilon_{0\mathrm{g}} - \theta^n)/\theta^n] = \sqrt{g_{rr,1}} - 1$, approaching the surface from below. [`NVAR` ≥ 2, **const**]

C37 Dimensionless difference $\epsilon_0 - \epsilon_{0\mathrm{g}} = \theta^n \sqrt{g_{rr}}[1 - 1/(1+\sigma\theta)^n]$ at dimensionless radius ξ. To be multiplied by energy density scale factor (14) to get true difference $\mathscr{E}_0 - \mathscr{E}_{0\mathrm{g}} = \epsilon^*(\epsilon_0 - \epsilon_{0\mathrm{g}})$ at radius r. At the surface the value vanishes. [`NVAR` ≥ 2]

C38 Relative difference $(\epsilon_0 - \epsilon_{0\mathrm{g}})/\theta^n = \sqrt{g_{rr}}[1 - 1/(1+\sigma\theta)^n]$ at dimensionless radius ξ. The limit for approaching the surface from below vanishes. [`NVAR` ≥ 2]

C39 Relative difference $(\epsilon_0 - \theta^n)/\epsilon_{0\mathrm{g}} = (1+\sigma\theta)^n(1 - g_{rr}^{-1/2})$ at dimensionless radius ξ. [`NVAR` ≥ 2]

C40 Limit of relative difference $\lim_{\xi \to \xi_{1-}} [(\epsilon_0 - \theta^n)/\epsilon_{0\mathrm{g}}] = 1 - g_{rr,1}^{-1/2}$, approaching the surface from below. [NVAR ≥ 2, **const**]

C41 Relative difference $(\epsilon_{0g} - \theta^n)/\epsilon_{0g} = 1 - (1 + \sigma\theta)^n g_{rr}^{-1/2}$ at dimensionless radius ξ. [NVAR ≥ 2]

C42 Limit of relative difference $\lim_{\xi \to \xi_{1-}} [(\epsilon_{0g} - \theta^n)/\epsilon_{0g}] = 1 - g_{rr,1}^{-1/2}$, approaching the surface from below. [NVAR ≥ 2, **const**]

C43 Relative difference $(\epsilon_0 - \epsilon_{0g})/\epsilon_{0g} = (1 + \sigma\theta)^n - 1$ at dimensionless radius ξ. The limit for approaching the surface from below vanishes. [NVAR ≥ 2]

C44 Dimensionless proper energy e_0 inside dimensionless radius ξ governed by Eq. (4). To be multiplied by energy scale factor (12) to get true proper energy $E_0 = \mathscr{E}^* e_0$ inside radius r. [NVAR ≥ 4, stored in array `yp[4][]`]

C45 Total dimensionless proper energy $e_{0,1} \equiv e_0(\xi_1)$. To be multiplied by energy scale factor (12) to get true total proper energy $E_{0,1} = \mathscr{E}^* e_{0,1}$. [NVAR ≥ 4, **const**]

C46 Dimensionless gravitational potential energy $\omega \equiv e_0 - v$ inside dimensionless radius ξ. To be multiplied by energy scale factor (12) to get true gravitational potential energy $\Omega = E_0 - mc^2 = \mathscr{E}^*(e_0 - v)$ inside radius r. [NVAR ≥ 4]

C47 Total dimensionless gravitational potential energy $\omega_1 = e_{0,1} - v_1$. To be multiplied by energy scale factor (12) to get total true gravitational potential energy $\Omega_1 = E_{0,1} - Mc^2 = \mathscr{E}^*(e_{0,1} - v_1)$. [NVAR ≥ 4, **const**]

C48 Relative gravitational potential energy $\omega/v = e_0/v - 1$ inside dimensionless radius ξ. [NVAR ≥ 4]

C49 Total relative gravitational potential energy $\omega_1/v_1 = e_{0,1}/v_1 - 1$. [NVAR $\geq$ 4, **const**]

C50 Dimensionless proper rest energy e_{0g} inside dimensionless radius ξ governed by Eq. (5). To be multiplied by energy scale factor (12) to get true proper rest energy $E_{0g} = \mathscr{E}^* e_{0g}$ inside radius r. [NVAR ≥ 5, stored in array `yp[5][]`]

C51 Total dimensionless proper rest energy $e_{0g,1}$. To be multiplied by energy scale factor (12) to get true total proper rest energy $E_{0g,1} = \mathscr{E}^* e_{0g,1}$. [NVAR ≥ 5, **const**]

C52 Dimensionless binding energy $e_b \equiv e_{0g} - v$ inside dimensionless radius ξ. To be multiplied by energy scale factor (12) to get true binding energy $E_b = \mathscr{E}^* e_b$ inside radius r. [NVAR ≥ 5]

C53 Total dimensionless binding energy $e_{b,1} = e_{0g,1} - v_1$. To be multiplied by energy scale factor (12) to get true total binding energy $E_{b,1} = \mathscr{E}^* e_{b,1}$. [NVAR ≥ 5, **const**]

C54 Relative binding energy $e_\mathrm{b}/v = e_\mathrm{0g}/v - 1$ inside dimensionless radius ξ. **Caution:** the limit for $\xi \to 0_+$ is $(1+\sigma)^{-n} - 1$. [`NVAR` ≥ 5]

C55 Total relative binding energy $e_{\mathrm{b},1}/v_1$. [`NVAR` ≥ 5, **const**]

C56 Relative binding energy ω/e_0g inside dimensionless radius ξ. [`NVAR` ≥ 5]

C57 Total relative binding energy $\omega_1/e_{\mathrm{0g},1}$. [`NVAR` ≥ 5, **const**]

C58 Dimensionless kinetic energy $e_\mathrm{k} = e_0 - e_\mathrm{0g}$ inside dimensionless radius ξ. To be multiplied by energy scale factor (12) to get true kinetic energy $E_\mathrm{k} = \mathscr{E}^* e_\mathrm{k}$ inside radius r. [`NVAR` ≥ 5]

C59 Total dimensionless kinetic energy $e_{\mathrm{k},1} = e_{0,1} - e_{\mathrm{0g},1}$. To be multiplied by energy scale factor (12) to get true total kinetic energy $E_{\mathrm{k},1} = \mathscr{E}^* e_{\mathrm{k},1}$. [`NVAR` ≥ 5, **const**]

C60 Relative kinetic energy e_k/v inside dimensionless radius ξ. **Caution:** the limit for $\xi \to 0_+$ is $1 - (1+\sigma)^{-n}$. [`NVAR` ≥ 5]

C61 Total relative kinetic energy $e_{\mathrm{k},1}/v_1$. [`NVAR` ≥ 5, **const**]

C62 Relative binding energy $e_\mathrm{b}/e_\mathrm{0g}$ inside dimensionless radius ξ. **Caution:** the limit for $\xi \to 0_+$ is $1 - (1+\sigma)^n$. [`NVAR` ≥ 5]

C63 Total relative binding energy $e_{\mathrm{b},1}/e_{\mathrm{0g},1}$. [`NVAR` ≥ 5, **const**]

C64 Relative kinetic energy $e_\mathrm{k}/e_\mathrm{0g}$ inside dimensionless radius ξ. **Caution:** the limit for $\xi \to 0_+$ is $(1+\sigma)^n - 1$. [`NVAR` ≥ 5]

C65 Total relative kinetic energy $e_{\mathrm{k},1}/e_{\mathrm{0g},1}$. [`NVAR` ≥ 5, **const**]

C66 Dimensionless Euclidean vertical axis z of ordinary embedding diagram governed by Eq. (6) at dimensionless radius ξ. Scaling is lacking sense so it is not provided. [`NVAR` ≥ 6, stored in array `yp[6][]`]

C67 Value $z_1 = z(\xi_1)$ of dimensionless Euclidean vertical axis z of ordinary embedding diagram at the surface. Scaling is lacking sense so it is not provided. [`NVAR` ≥ 6, **const**]

C68 Relative Euclidean vertical axis $z/z_1 \in [0,1]$ of ordinary embedding diagram at dimensionless radius ξ. Scaling is lacking sense so it is not provided. [`NVAR` ≥ 6]

C69 Dimensionless Euclidean radial axis $\chi = (h_{\phi\phi})^{1/2} = \xi(-g_{tt})^{-1/2} = \xi(1 + \sigma\theta)^{n+1}\left\{1 - 2\sigma(n+1)\left[\frac{v_1}{\xi_1} + \frac{1}{3}\lambda\xi_1^2\right]\right\}^{-1/2}$ at dimensionless radius ξ. Scaling is lacking sense so it is not provided. [`NVAR` ≥ 2 (but has sense only with `NVAR` ≥ 7)]

C70 Dimensionless Euclidean vertical axis $\tilde{z}$ of optical embedding diagram governed by Eqs (20) and (22) at dimensionless radius ξ. Scaling is lacking sense so it is not provided. [`NVAR` ≥ 7, stored in array `yp[7][]`]

C71 Value $\tilde{z}_1 = \tilde{z}(\xi_1)$ of dimensionless Euclidean vertical axis $\tilde{z}$ of optical embedding diagram at the surface. Scaling is lacking sense so it is not provided. [`NVAR` ≥ 7, **const**]

C72 Relative Euclidean vertical axis $\tilde{z}/\tilde{z}_1$ of optical embedding diagram at dimensionless radius ξ. Scaling is lacking sense so it is not provided. [`NVAR` ≥ 7]

C73 Polytropic/adiabatic index n. [`NVAR` ≥ 2, **const**]

C74 Relativity parameter $\sigma = p_c/\rho_c c^2$. !!!! For ADIABATIC [`NVAR` ≥ 2, **const**]

C75 Cosmological parameter $\lambda = \rho_{vac}/\rho_c$. See also Eq. (26). [`NVAR` ≥ 2, **const**]

C76 Length scale factor $\mathscr{L}^*$ defined by Eq. (10). [`NVAR` ≥ 2, **const**]

C77 Mass scale factor $\mathscr{M}^*$ defined by Eq. (11). [`NVAR` ≥ 2, **const**]

C78 Energy scale factor $\mathscr{E}^*$ defined by Eq. (12). [`NVAR` ≥ 2, **const**]

C79 Mass density scale factor μ^* defined by Eq. (13). [`NVAR` ≥ 2, **const**]

C80 Energy density scale factor ϵ^* defined by Eq. (14). [`NVAR` ≥ 2, **const**]

C81 Pressure scale factor π^* defined by Eq. (15). [`NVAR` ≥ 2, **const**]

6.4 Examples

here is an example how to send grpfs's output to the file `output.dat` after an interactive run using the fixed-stepsize method:

```
$ grpfs -if > output.dat
```

To append another output to a previous one, use

```
$ grpfs -if >> output.dat
```

7 Advanced usage

This Section describes some features of grpfs that rely on Unix-like systems' capability of redirecting standard terminal input/output into a file or from a file and piping it between processes, as has been touched in the previous Section. Therefore, some basic knowledge of these features can be beneficial.

7.1 Feeding input from an output file

Although the title of this Subsection may sound rather weirdly, the output produced by grpfs can really be reused as the input. If you examined the output thoroughly, you must have noticed that certain comment lines start with strings

‘`#!`’ instead of mere ‘`#`’. These *metacomments* precede exactly those numerical strings that the program requires as input. So, if you redirect the grpfs’s standard input to a file written out beforehand and named, say, `output.dat`, like this,

```
$ grpfs -b < output.dat
```

the program will skip all lines commented out with ‘`#`’ but will load the numerical strings that follow the metacomment strings ‘`#!`’. Don’t forget the ‘`-b`’ option; although this omission has no influence on the function, the screen would be messed with nasty interactive prompts. Moreover, this option turns on the input data conformity check (see below in this Subsection), and turns off checking for meaningful values.

So, this capability makes it possible to rerun any calculation you have stored in a file. In fact, the output should exactly mimic the input (with the exception of some records in the `ID_CHART` section, of course), as you can convince yourself by giving the commands

```
$ grpfs -b < output.dat > output.new
$ diff output.dat output.new
```

The structure of grpfs’s input differs for fixed stepsize and adaptive stepsize methods. That is why each output is labeled with a ‘magic string’ at the very beginning: the fixed stepsize method has magic string ‘`#M0`’, while the adaptive stepsize ones have `#M`⟨*method number*⟩, where ⟨*method number*⟩ is `1` for 5th-order Cash–Karp Runge–Kutta method, `2` for the Bulirsch–Stoer method, `3` for 4-th order Rosenbrock method, `4` for Bulirsch–Stoer semi-implicit midpoint rule method. You can only input data conforming with the method selected by the ‘`-f`’ or ‘`-a`’ option (in other words, choose ‘`-f`’ or no option if the input is labeled ‘`#M0`’ and ‘`-a`’ if the input is labeled ‘`#M1`’ through ‘`#M4`’), otherwise the program stops, complaining

```
grpfs: input dataset: mismatch between option/stdin, exiting to system
```

There should be emphasized that all the adaptive stepsize outputs are interchangeable if used as input, however, the output data may differ.

7.2 Filtering output through noutils

See the package noutils [Ref] for more info.

7.3 Examples

Here will be examples on using *less*, *tee*, etc.

8 Configuration

There are three preprocessor options in the `Makefile` ‘Customization section,’ subsection ‘Preprocessor options.’

`DOPT1 = -DDBL_PREC` If set, grpfs will use double precision floating-point arithmetic—this is highly recommended since the single precision range is frequently not sufficient to hold values of cosmological constant and similar ones. This option is set by default. Unless set, grpfs will use single precision floating-point arithmetic.

`DOPT2 = -DVERBOSE` If set, grpfs runs in a very verbose mode; this option is recommended and set by default.

`DOPT1 = -DRAT_EXTR` If set, grpfs will use rational extrapolation subroutine instead of the polynomial one. For experimental purposes and disabled by default.

It is also worth mentioning the preprocessor symbolic constant `NVAR`. Setting it to $1, 2, \ldots, 7$ one can control the number of the equations to solve (see also Section 6). This constant can be set in the header file `grp.h`.

There is a lot of customizable symbolic constants in `grp.h`, `grp_io.h` and `grp_node.h`. In this point, the user is recommended to study [Press et al., 1997] and to read the source codes.

When using package noutils [Ref] and dealing with constant combination of columns for a long time, it is worth setting and exporting the environmental variable `PICKOLS`. See [Ref] for detailed information.

9 Bugs

Please send bug reports to `Stanislav.Hledik@fpf.slu.cz`. Indicate clearly grpfs-bugs in the subject field. Another possibility is to send complaints to `/dev/null`.

Acknowledgments

The author would like to thank to the Silesian University at Opava, Faculty of Philosophy and Science, Institute of Physics, for providing computing facilities that were used for developing the code and the manual.

References

[Misner et al., 1973] Misner, C. W., Thorne, K. S., and Wheeler, J. A. (1973). *Gravitation.* W. H. Freeman and Co, New York, San Francisco.

[Press et al., 1997] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1997). *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, Cambridge, 2nd edition.

[Tooper, 1964] Tooper, R. F. (1964). General relativistic polytropic fluid spheres. *Astrophys. J.*, 140(2):434–459.

[Tooper, 1965] Tooper, R. F. (1965). Adiabatic fluid spheres in general relativity. *Astrophys. J.*, 142(4):1541–1562.

A Differences between grpfs and grafs

To be finished!!!!

B Source code

The kernel of the system—the numerical routines that perform integration, interpolation and extrapolation—are adopted from [Press et al., 1997] and included in module `grp_node.c`. The program also utilizes the memory allocating and other utility routines from [Press et al., 1997] concentrated in the module `grp_nutl.c`, but the latter are freely distributable contrary to the former ones that are subject to copyright and cannot be placed in a freely accessible site.

The user can make some settings in the header file `grp.h`, especially in the section marked as 'customization section.'