Special functions in R: introducing the gsl package

Robin K. S. Hankin

Abstract

This vignette introduces the **gsl** package of R utilities for accessing the functions of the Gnu Scientific Library.

An earlier version of this document was published as Hankin (2006).

Keywords: R, special functions.

1. Introduction

The Gnu Scientific Library (GSL) is a collection of numerical routines for scientific computing (Galassi *et al.* 2005). The routines are written in C and constitute a library for C programmers; the source code is distributed under the GNU General Public License. One stated aim of the GSL development effort is the development of wrappers for high level languages.

The R programming language (R Development Core Team 2008) is an environment for statistical computation and graphics. It consists of a language and a run-time environment with graphics and other features.

Here I introduce **gsl**, an R package that allows direct access to many GSL functions, including all the special functions, from within an R session. The package is available on CRAN, `http://www.cran.r-project.org/`; the GSL is available at `http://www.gnu.org/software/gsl/`.

2. Package design philosophy

The package splits into two parts: the special functions, written by the author; and the **rng** and **qrng** functionality, written by Duncan Murdoch. These two parts are very different in implementation, yet follow a common desideratum, namely that the package be a transparent port of the GSL library. The package thus has the advantage of being easy to compare with the GSL, and easy to update verifiably.

In this paper, the Airy functions are used to illustrate the package. They are typical of the package's capabilities and coding, and are relatively simple to understand, having only a single real argument. A brief definition, and an application in physics, is given in the appendix.

The package is organized into units that correspond to the GSL header file. Thus all the Airy functions are defined in a single header file, `gsl_sf_airy.h`. The package thus contains a corresponding C file, `airy.c`; an R file `airy.R`, and a documentation file `Airy.Rd`. These three files together encapsulate the functionality defined in `gsl_sf_airy.h` in the context of an R package. This structure makes it demonstrable that the GSL has been systematically

and completely wrapped.

Functions are named such that one can identify a function in the GSL manual, and the corresponding R command will be the same but with the prefix[1] and, if present, the "`_e`" suffix, removed. In the case of the special functions, the prefix is "`gsl_sf_`". Thus, GSL function `gsl_sf_airy_Ai_e()` of header file `gsl_sf_airy.h` is called, via intermediate C routine `airy_Ai_e()`, by R function `airy_Ai()`. Documentation is provided for every function defined in `gsl_sf_airy.h` under `Airy.Rd`.

The **gsl** package is not intended to add any numerical functionality to the GSL, although here and there I have implemented slight extensions such as the Jacobian elliptic functions whose R ports take a complex argument.

2.1. Package documentation

The **gsl** package is unusual in that its documentation consists almost entirely of pointers to the GSL reference manual (Galassi *et al.* 2005), and Abramowitz and Stegun (1965). This follows from the transparent wrapper philosophy. In any case, the GSL reference manual would strictly dominate the `Rd` files of the **gsl** package.

3. Package gsl in use

Most functions in the package are straightforwardly and transparently executable:

```
> airy_Ai(1:3)

[1] 0.13529242 0.03492413 0.00659114
```

The online helpfiles include many examples that reproduce graphs and tables that appear in Abramowitz and Stegun. This constitutes a useful check on the routines. For example, figures 1 and 2 show an approximate reproduction of their figures 10.6 and 10.7 (page 446).

4. Summary

The **gsl** package is a transparent R wrapper for the Gnu Scientific Library. It gives access to all the special functions, and the quasi-random sequence generation routines. Notation follows the GSL as closely as reasonably practicable; many graphs and tables appearing in Abramowitz and Stegun are reproduced by the examples in the helpfiles.

Acknowledgments

I would like to acknowledge the many stimulating and helpful comments made by the R-help list over the years.

References

[1] Some functions, such as `gsl_sf_sin()`, retain the prefix to avoid conflicts. A full list is given in `Misc.Rd`.

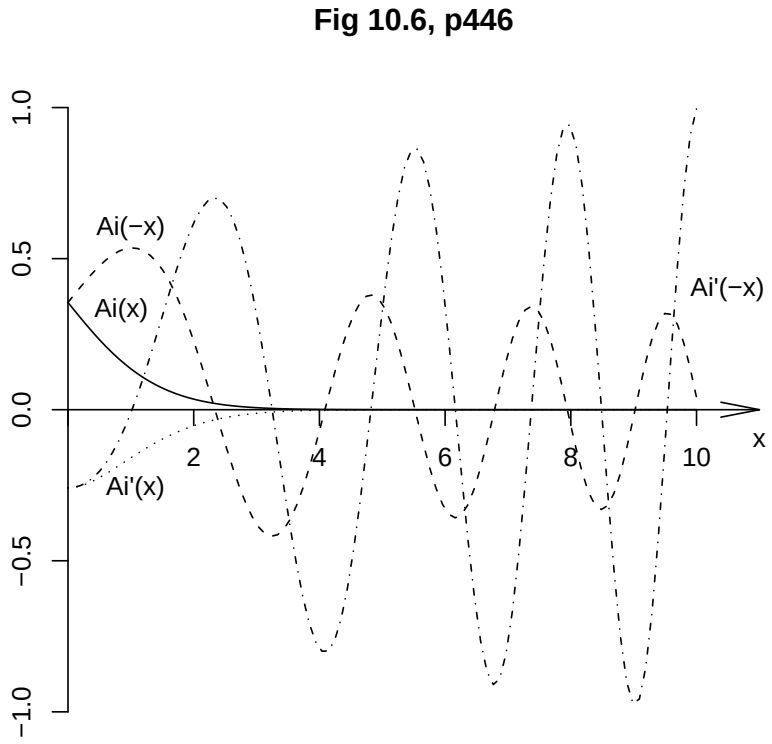

Figure 1: Functions $\mathrm{Ai}(\pm x)$ and $\mathrm{Ai}'(\pm x)$ as plotted in the helpfile for `airy_Ai()` and appearing on page 446 of Abramowitz and Stegun (1965)

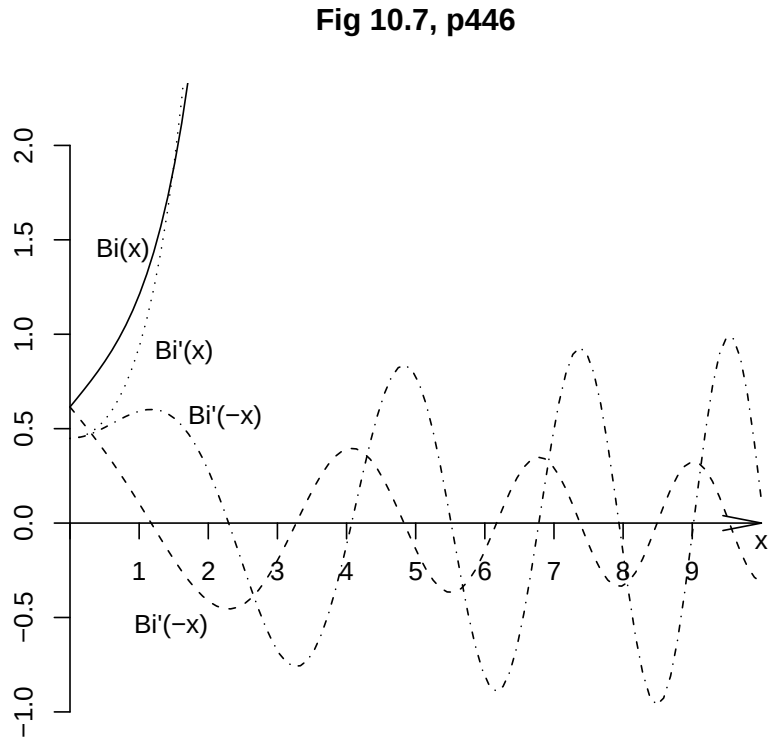

Figure 2: Functions $\mathrm{Bi}(\pm x)$ and $\mathrm{Bi}'(\pm x)$ (Abramowitz and Stegun 1965)

Abramowitz M, Stegun IA (1965). *Handbook of mathematical functions*. New York: Dover.

Galassi M, *et al.* (2005). *GNU Scientific Library*. Reference Manual edition 1.7, for GSL version 1.7; 13 September 2005.

Hankin RKS (2006). "Special functions in R: introducing the **gsl** package." *R News*, **6**(4), 24–26.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

Vallée O, Soares M (2004). *Airy functions and applications to physics*. World Scientific.

Appendix: The Airy function and an application in quantum mechanics

The Airy function may not be familiar to some readers; here, I give a brief introduction to it and illustrate the **gsl** package in use in a physical context. The standard reference is Vallée and Soares (2004).

For real argument x, the Airy function is defined by the integral

$$\mathrm{Ai}(x) = \frac{1}{\pi}\int_0^\infty \cos\left(t^3/3 + xt\right)\, dt \tag{1}$$

and obeys the differential equation $y'' = xy$ (the other solution is denoted $\mathrm{Bi}(x)$).

In the field of quantum mechanics, one often considers the problem of a particle confined to a potential well that has a well-specified form. Here, I consider a potential of the form

$$V(r) = \begin{cases} r & \text{if } r > 0 \\ \infty & \text{if } r \leq 0 \end{cases} \tag{2}$$

Under such circumstances, the energy spectrum is discrete and the energy E_n corresponds to the n^{th} quantum state, denoted by ψ_n. If the mass of the particle is m, it is governed by the Schrödinger equation

$$\frac{d^2\psi_n(r)}{dr^2} + \frac{2m}{\hbar^2}\left(E_n - r\right)\psi_n(r) = 0 \tag{3}$$

Changing variables to $\xi = \left(E_n - e\right)\left(2m/\hbar\right)^{1/3}$ yields the Airy equation, viz

$$\frac{d^2\psi_n}{d\xi^2} + \xi\psi_n = 0 \tag{4}$$

with solution

$$\psi_n(\xi) = N\mathrm{Ai}\left(-\xi\right) \tag{5}$$

where N is a normalizing constant (the $\mathrm{Bi}\left(\cdot\right)$ term is omitted as it tends to infinity with increasing r). Demanding that $\psi_n(0) = 0$ gives

$$E_n = -a_{n+1}\left(\hbar^2/2m\right)^{1/3}$$

where a_n is the n^{th} root of the Ai function [`Airy_zero_Ai()` in the package]; the off-by-one mismatch is due to the convention that the ground state is conventionally labelled state zero, not state 1. Thus, for example, $E_2 = 5.5206\left(\hbar^2/2m\right)^{1/3}$.

The normalization factor N is determined by requiring that $\int_0^\infty \psi^*\psi\, dr = 1$ (physically, the particle is known to be somewhere with $r > 0$). It can be shown that

$$N = \frac{(2m/\hbar)^{1/6}}{\mathrm{Ai}'(a_n)}$$

[the denominator is given by function `airy_zero_Ai_deriv()` in the package] and the full solution is thus given by

$$\psi_n(r) = \frac{(2m/\hbar)^{1/6}}{\mathrm{Ai}'(a_n)}\mathrm{Ai}\left[\left(\frac{2m}{\hbar}\right)^{1/3}(r - E_n)\right]. \tag{6}$$

Figure 3 shows the first six energy levels and the corresponding wave functions.

Affiliation:

Robin K. S. Hankin
University of Cambridge
19 Silver Street
Cambridge CB3 9EP
United Kingdom
E-mail: hankin.robin@gmail.com

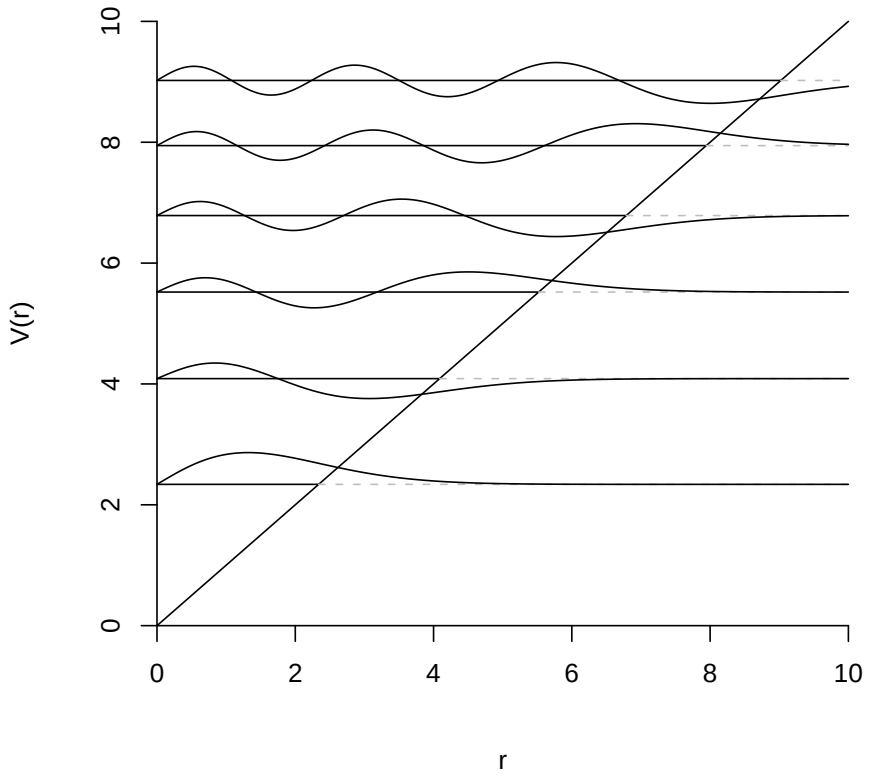

Figure 3: First six energy levels of a particle in a potential well (diagonal line) given by equation 2