Chapter 1
Why R, and Why Now?

Abstract For many years, the R language has had a reputation as a premier system for interactive data analysis. From a user's perspective, there are two main reasons for this. First, R is a language designed specifically for working with data, so it has important practical features (e.g. sensible treatment of missing values) that are not found in more general languages. Second, R comes with a vast array of high-quality packages, or libraries, that handle specialized tasks. The packages are contributed by experts in various fields, and tend to be tied closely to the literature—two facts that are relevant in an integrative field such as oceanography. The case for R has grown stronger in recent years, with a general movement to open-source software, and with specialized aspects of oceanographic data analysis becoming available in the oce package. Now is a good time for oceanographers to try R.

In a young scientific field, work is often carried out by postgraduate students whose thesis goals inspire new procedures intended for somewhat limited application. These procedures might be called practical (or provisional) operating procedures (POP), by analogy to the standardized operating procedures (SOP) used in more routine work. As fields mature, POP may be translated to SOP, expanding the range of application and permitting a shift in workload to technicians who do not need postgraduate training. According to this line of reasoning, new undergraduate programmes can be a sign of a maturing field. This is the state of oceanography today.[1]

The task of translating POP to SOP may be eased if similar tools are used in each, so it makes sense to consider the choice of tools carefully. In this spirit, Fig. 1.1

[1] For an example, the author was contributing to the development of a new undergraduate programme at Dalhousie University, while working on this book.

D. E. Kelley, *Oceanographic Analysis with R*,
https://doi.org/10.1007/978-1-4939-8844-0_1

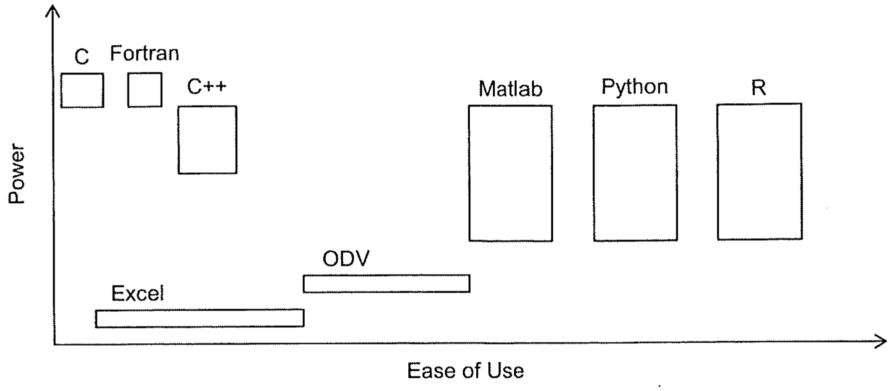

Fig. 1.1 Comparison of general-purpose computing languages or applications that may be used for oceanographic analysis

compares R with some other systems that might be used for oceanographic data analysis.[2] Some of these systems hold little promise, but it is worth touching on them all, if only for the excuse to bring up some general issues.

The diagram suggests that Excel scores poorly on both power and usability. While the first point is unlikely to be contested by anyone who has tried to use Excel on a large dataset, some readers might argue that Excel is easy to use. However, the context is important. Compared with its competitors, Excel is ill-suited to the particular calculations and graphical displays that oceanographers need. For example, it is easy to add columns in Excel, but considerably more difficult to correctly enter a formula for seawater density that contains dozens of numerical values specified to five or more digits. Also, the very thing that makes Excel popular for nontechnical work, its graphical user interface (GUI), is an impediment in technical work,[3] because a sequence of GUI operations is difficult to describe and reproduce.[4] A text-based approach is preferable to a GUI approach for all but the simplest of tasks. Those who switch from Excel to R should see benefits quickly, and should find the transition easy, because there are tools for combining the two systems (Heiberger and Neuwirth 2009).

To some extent, the box for Excel in Fig. 1.1 is a place-holder for other GUI systems, and so these need not be discussed in much detail, with one exception:

[2] This book deals more with data analysis than with statistics. For early thoughts on data analysis, see the influential paper by Tukey (1962), along with the recent historical commentary by Mallows (2006).

[3] GUI-based systems can be problematic for users with weak vision, with text-based systems such as R providing a better choice (Godfrey 2013; Godfrey and Erhardt 2014).

[4] Issues in reproducible research are discussed by Pebesma et al. (2012), while Herndon et al. (2013) detail problems particular to Excel.

Ocean Data Viewer.[5] ODV is a GUI-based system that offers good support for many oceanographic operations, including the equation of state, specialized graphing, etc. However, its power is limited by both its GUI-based design and the fact that the ODV source code is not available for inspection or modification.

All the other entries in Fig. 1.1 are languages, some compiled and others interpreted. Languages are well-suited for reproducible research, because the code used to solve a problem is, in and of itself, a full description of the processing procedure. Good coding practices make the transference of effort between tasks or work groups an easy matter, often involving little more than changing the name of a data file. Readers who are accustomed to the GUI approach will discover other benefits in adopting the language approach. Loops make it easy to carry out repetitive work. Conditional blocks handle changing circumstances. Functions and object-orientation yield specialization and simplicity of operation, without loss of generality. The only cost for these benefits is a learning process that starts with thinking beyond menus and icons.

Generally, compiled languages offer higher efficiency than interpreted ones, but they are much more difficult to use. This is why the compiled languages C, Fortran and C++ are placed on the left of Fig. 1.1. These are used in the most demanding of computing tasks, from operating systems to climate models. The relative positions of these languages on the diagram are debatable, since they depend on the nature of the work being carried out. C offers essentially the full power of the machine, but the language is difficult to use for oceanographic work, because of its weak support for matrices and other high-level data types. Fortran offers similar power, and has an advantage over C in its strong support for matrices. In some ways, C++ is even easier to use, with an object orientation model that reduces coding effort and facilitates collaboration, but its object orientation can impose efficiency penalties, if users rely on overly indirect algorithm expression.

Although compiled languages underpin all computing applications, and remain the best solution for large computing tasks such as numerical models, they have fallen out of favour for interactive work. This is particularly true for so-called "exploratory data analysis" as described in the seminal treatment of Tukey (1977) and more recently by, e.g., Velleman and Hoaglin (2004). Of many interpreted languages that might be discussed in the present context, three stand out: Matlab, Python, and R.

As with the compiled languages shown in Fig. 1.1, the relative merits of the interpreted languages depend on the work being done. The illustrated efficiency ranges are large because not all problems map well to the fastest components of the languages. For example, these three languages all provide strong low-level support for matrices, so that problems that can be cast in matrix form are handled with efficiency approaching that of compiled languages. Importantly, each also allows

[5] http://data.unep-wcmc.org.

advanced users to frame parts of their algorithms in compiled languages, yielding great improvements in speed.[6]

The most contentious aspect of Fig. 1.1 may be the ranking of Matlab, Python and R in terms of ease of use, for this is the sort of judgement that partly boils down to a matter of taste. The diagram expresses the author's opinion, based on years of experience, that Python is superior to Matlab, and that R is superior to both. This reflects several factors. First, both Python and R are popular in more diverse fields (at least outside oceanography), which means that users of these languages can benefit from the efforts of broad communities of experts. The popularity lies partly in the technical merits of the languages, and partly in their open-source licenses. Especially in a university setting, open-source systems attract talented people who have a habit of sharing their work, and this can lead to nonlinear improvements to the development process.[7] For Python and R, the shared efforts are organized through systems that bundle software code with documentation and test cases. The bundles are called packages in R. These packages are a significant factor in the present judgement of the superiority of R, since they provide the power to tackle a myriad of tasks that come up in oceanographic analysis.

An important package in the oceanographic context is `oce`. As discussed in Chap. 3 and throughout this book, `oce` handles dozens of specialized oceanographic data formats, and provides functions for calculations and graphical displays that are specific to oceanography. Its object-oriented approach lets novices get results quickly, without imposing undue limits on experts. Reproducible research is built into the foundation of the package, with a processing log being contained in all `oce` data objects. Few limitations are imposed on the scope of work done with `oce`, because the package integrates well with both the base R language and other packages.

Based on factors such as those listed above, the thesis statement of this book is that R is a powerful system for oceanographic analysis, with high potential for open-ended research and more routine technical work. Simply stated, it is a tool that works well, and fits comfortably in the hand.

There *is* a learning process in adopting R, and this book is designed to accelerate that process, in different ways for readers of different backgrounds. The author is a research scientist and an educator, not a salesman, and so the text points out the weaknesses of R, as well as strengths. For many readers, these strengths and weaknesses will be measured against Matlab, and so an early component of the tutorial provided in the next chapter is a brief comparison of the two languages.

[6] For example, the `oce` package (Kelley and Richards 2018) uses C to decode the binary data files produced acoustic Doppler instruments, reducing computation times by orders of magnitude compared with pure R.

[7] See Raymond (2001) for a general discussion of open-source development, Fox (2009) for comments in the R context, and Lowndes et al. (2017) for details of how using R and other open-source tools can enhance reproducibility in ocean science.