What's next for atomic and molecular physics software?

Brown, A. (2019). What's next for atomic and molecular physics software? *Nature Reviews | Physics*, *1*(12). https://doi.org/10.1038/s42254-019-0139-y

COMMENT

What's next for atomic and molecular physics software?

Andrew Brown

Centre for Theoretical Atomic, Molecular and Optical Physics (CTAMOP), Queen's University Belfast, UK

E-mail: andrew.brown@qub.ac.uk

There are several atomic, molecular and optical physics computer codes. Knowing which one to use for what and how is a challenge. Andrew Brown surveys the available software packages and discusses how code development practices in academia could be improved.

A rather daunting task awaits any newcomer to the field of computational atomic physics, and it has nothing to do with the complexities of the science. Instead, one is inundated with incomprehensible acronyms. I recently had to unpick some confusion over CPP, CPC, PCCP and CCPQ (which denote a research cluster, two journals and a collaborative project in the UK respectively). When it comes to the methods and computer codes the picture is no less confusing: just in my own field of strong-field atomic physics one has to learn the subtle differences between CASSCF, RASSCF, MCHF, CIS, DFT... not to mention their time-dependent counterparts (prepend 'TD' as in TD-CASSCF and so on).

If you are interested in atomic molecular and optical physics, your first step should be to give yourself a solid grounding in the history of the numerical methods that are still used after nearly 100 years of development [1]. Then, for some simple problems, you might be able to engineer a pen-and-paper solution or even a short piece of computer code. However, for real, heavyweight atomic molecular and optical physics research, one has to navigate a veritable jungle of software packages— each with their own unique capabilities, limitations, and obligatory acronymic name.

[H1] Time-independent problems

The fundamental problem of atomic, molecular and optical physics is to solve the Schrödinger equation, in both its time-dependent and independent forms. Until quite recently, researchers were primarily concerned with the latter: describing the structure of atoms and molecules and considering any dynamics primarily from the perspective of energy. This is why time-independent codes are used to describe apparently time-dependent processes such as photoionization or scattering; the concern is not with the timing, but rather the initial and final energies of the constituent particles. And although research into these processes started more than 50 years ago, this remains a vibrant field where well-established methods continue to produce new insights with real applications for astrophysics and fusion energy.

For treating atomic systems, the standard is set by the atomic R-matrix codes, which come with their own long history and an amicable rivalry between the two leading developer groups in Queen's University, Belfast and Drake University, Iowa. The codes developed by both groups are freely available and well documented and although there is a substantial overlap in applications, there are differences in the implementation. For instance, the *B*-spline atomic R-matrix codes (BSR) codes [2] from the Iowa group undoubtedly offer enhanced functionality, but they are probably more useful for experienced users. The Belfast codes come with a handy set of Perl scripts which make the process of running them far less opaque

(for basic problems at least). Unfortunately, the simple problems have all been done: for anything realistic you're going to have to roll your sleeves up and dive into the documentation.

For molecular systems there are, broadly speaking, two choices: R-matrix methods and Kohn variational methods. Kohn variational methods has been used extensively in recent years for electron-molecule scattering problems. However, if you want to do this kind of calculation, you're likely going to have to write the code yourself as there aren't any codes that have been made available for general use.

By contrast, the molecular R-matrix approach is represented almost uniquely by the UKRMol+ code which is freely available, and used by several groups around the world. Like the Belfast atomic codes, the suite contains Perl scripts for running basic calculations. However, a beginner will not be able to do much, except run the basic test calculations, and the documentation for these codes is not as well developed as for the atomic codes. There is, however, a very active group of users and developers, and the pay-off for learning to use the code is that UKRmol+ is unrivalled for describing electron correlations.

An alternative approach is the convergent close coupling method which is developed specifically for atoms and small molecules (meaning it is not as general as UKRmol+, but may be better suited for specific calculations). Although the convergent close coupling codes are not publicly available, they are accessible through the AMP gateway project which hosts the Convergent close coupling, UKRMol+, BSR and other codes and makes them available for use on supercomputing facilities in the US.

[H1] Time-dependent problems

Traditionally, it was thought that solving the time-dependent Schrödinger equation was not tractable for systems involving more than a few electrons, but with the introduction of ultrashort laser pulse sources in the early 1990s there was a critical need for theory which could describe general multi-electron systems in a time-dependent fashion. This became possible thanks to advances in methodology and an explosion in the availability and power of computational resources.

Excitingly for the new user, there are several different open-source packages available online. These include R-matrix with time (RMT) and the Jena Atomic Calculator (JAC). The JAC is an incredibly full-featured, interactive system for modelling various kinds of atomic processes (laser-driven dynamics, structure calculations and so on). It is well documented, built with usability in mind, and has the added advantage that it provides an interface to perform individual steps of larger calculations (such as angular momentum algebra, for instance) which larger packages such as RMT bury deep in the source code. That said (and here I must confess a serious bias) RMT is the most fully-featured and general code out there, and it is open source and freely available. It can describe atomic and molecular dynamics in arbitrarily polarised laser pulses, with a full account of multi-electron effects, and even includes capability to describe semi-relativistic effects in atoms. Again, as with the time-independent packages, the cost is a very steep learning curve. Although the documentation for RMT is better developed than most other codes of this type, it takes input from other, time-independent R-matrix packages, and thus performing even simple calculations requires a lot of setup. Thus, for most users JAC will be a better option in the short term.

[H1] Outlook

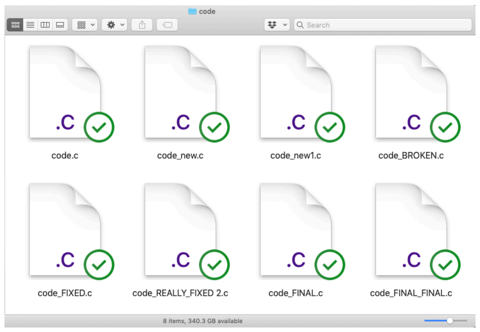

Figure 1 The typical anatomy of a code development project

In truth, the practices of researchers and programmers working in this field are still behind their counterparts in other areas of academia– such as quantum chemistry or molecular dynamics– and far behind the practices of non-academic code developers. A typical academic computer program evolves in a lurching, haphazard way as illustrated in Figure 1. Many are initially developed by a lone-genius to solve a very specific problem, before being 'enhanced', 'extended', 'refactored' or 'integrated' by a number of helpful research assistants. This process yields 7 versions of the same code, all of which do the same thing slightly differently (or perhaps 7 different things with approximately the same method). One might hope that only the best of these would survive in some sort of Darwinian selection, but invariably what happens is that all of them cling to life hidden away on someone's old laptop only to be found and resuscitated when something goes inexplicably wrong with `code-latest-working-1.1-NEWEST-VERSION.exe`.

I think there are three simple things academic developers can do to change this culture. First, make codes available to the community. There are academic journals for publishing codes, such as *Computer Physics Communications*, or you might just put the source code in an online repository: there are plenty of free services such as github, gitlab or bitbucket. Inevitably, if you actively encourage other people to see and use your code they will spot things you missed and this will give you a chance to make improvements. Second, spend some time on the documentation. This is the bit of code development that nobody has time to do, but there's very little chance that someone will use or contribute to your code if there aren't even basic instructions for doing so. The last thing is to start using some sort of version-control tool like SVN, mercurial or git. This will allow others to contribute in a manageable way, without running into the aforementioned problem of producing multiple, divergent copies of a single source code.

There are signs that the culture is changing. All of the R-matrix and JAC codes, for instance, are hosted in version-controlled, open access repositories, replete with comprehensive documentation. Additionally, the aforementioned AMP gateway project is starting to gain traction and this bodes well for the whole field of computational atomic, molecular and optical physics. One can only hope that the practice of making code available to the community catches on in other countries as it has existed in the UK for some time.

Moving from our current paradigm to one where communities of developers share their codes freely and sustainably, offering documentation and other support for users is the only way to ensure that the legacy of the last 100 years of computational atomic, molecular and optical physics is more than just a handful of academic papers. With a solid grounding in the underlying theory and a sustainable approach to the use, development and distribution of the resulting scientific software, we should have very high hopes for the next hundred years.

References

1. Schneider, B. & Gharibnejad, G. Numerical methods every atomic and molecular theorist should know. *Nature Reviews Physics* (2019).
2. Zatsarinny. O. BSR: *B*-spline atomic *R*-matrix codes. *Comp. Phys. Comm.* **174**, 273 (2006)

Related links

Parallel R-matrix codes. https://connorb.freeshell.org

UK-AMOR code repository. https://gitlab.com/uk-amor/

The AMP gateway project. https://ampgateway.org

The Jena Atomic Calculator. https://github.com/OpenJAC/JAC.jl