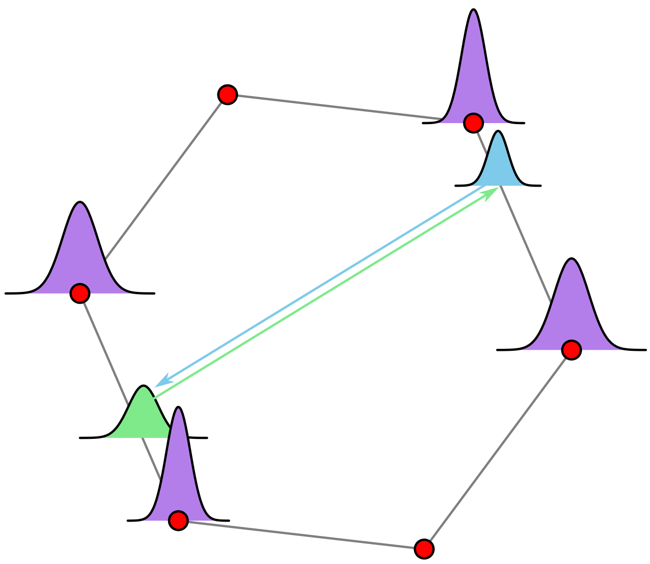
hello

FINAL REPORT

The Art of Scientific Computing: Introduction to Quantum Chemistry

Author:
Insert name here

Subject Handbook code:
COMP-90072



Faculty of Science
The University of Melbourne

1st March 2018

Subject co-ordinators: A/Prof. Roger Rassool & Prof. Harry Quiney

Contents

Chapter 1

Formatting options

The best way to view this document is in a split-view format, with the `.tex` file open and the corresponding portion of the `.pdf` output also visible.

1.1 Equations and tables

You can write an equation without a label by using an asterisk:

$$f(x) = (x + a)(x + b).$$

Alternatively, you can number and label your mathematical expressions:

$$\mathcal{L}_{\text{EM}} = \frac{1}{2}\left(\epsilon_0 \left|\boldsymbol{E}\right|^2 + \frac{1}{\mu_0}\left|\boldsymbol{B}\right|^2\right) - \phi\varrho_{\text{free}} + \boldsymbol{A}\cdot\boldsymbol{J}_{\text{free}} + \boldsymbol{E}\cdot\boldsymbol{P} + \boldsymbol{B}\cdot\boldsymbol{M}. \tag{1.1}$$

This way, we can later reference something we wrote earlier, like the electromagnetic Lagrangian density in non-relativistic vector notation shown in Equation (1.1).

The equation environment allows you to construct a number of objects, such as arrays and aligned sets of equations. For instance, we might want to write a general matrix:

$$\boldsymbol{c} = \left(\begin{array}{ccccc} c_{11} & c_{12} & c_{13} & \cdots & c_{1n} \\ c_{21} & c_{22} & c_{23} & \cdots & c_{2n} \\ c_{31} & c_{32} & c_{33} & \cdots & c_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & c_{m3} & \cdots & c_{mn} \end{array}\right). \tag{1.2}$$

We might alternatively want to write a set of simultaneous equations. If future descriptions require for all of them to have unique labels, try using the 'subequations' environment as well as the 'eqnarray' environment.

$$3x + 5y + z = 5 \tag{1.3a}$$

$$7x - 2y + 4z = 3 \tag{1.3b}$$

$$-6x + 3y + 2z = 1. \tag{1.3c}$$

This will allow you to talk about the set of simultaneous equations as a whole by referring the reader to Equation (1.3), or else to a specific statement like Equation (1.3b).

Sometimes you'll want to derive a result, which in its clearest form requires more than one line of algebra. There are a few ways to accomplish this, and the best choice will depend on the task at hand and your personal taste. You can instruct the compiler *not* to give a label to one or more of the lines as you go.

$$\begin{aligned} & R_x(\varepsilon)R_y(\varepsilon) - R_y(\varepsilon)R_x(\varepsilon) \\ &= \left(1 - \frac{iJ_x\varepsilon}{\hbar} - \frac{J_x^2\varepsilon^2}{2\hbar^2}\right)\left(1 - \frac{iJ_y\varepsilon}{\hbar} - \frac{J_y^2\varepsilon^2}{2\hbar^2}\right) - \left(1 - \frac{iJ_y\varepsilon}{\hbar} - \frac{J_y^2\varepsilon^2}{2\hbar^2}\right)\left(1 - \frac{iJ_x\varepsilon}{\hbar} - \frac{J_x^2\varepsilon^2}{2\hbar^2}\right) \\ &= \left(1 - \frac{iJ_z\varepsilon^2}{\hbar}\right) - 1 + \mathcal{O}(\varepsilon^3). \end{aligned} \tag{1.4}$$

A table of data is sometimes necessary, for which there is a dedicated environment. An example of this is given in Table (1.1) – note that the counter for tables is separate from the counter for equations. If this isn't to your liking, you can change the enumeration style of one or both. (Google has plenty of suggestions).

description		value		
electronic charge	e	$1.6021766 \times 10^{-19}$C		
proton mass	m_p	$1.6726218 \times 10^{-27}$kg	or	938.27203MeV$/c^2$
electron mass	m_e	$9.1093822 \times 10^{-31}$kg	or	510.99891keV$/c^2$
permittivity of vacuum	ϵ_0	$8.8541878 \times 10^{-12}$F/m		
permeability of vacuum	μ_0	1.2566371×10^{-6} H/m	or	$4\pi \times 10^{-7}$H/m
speed of light in vacuum	c	$2.9979246 \times 10^{+8}$ m/s		
electrostatic constant	k_e	$8.9875518 \times 10^{+9}$ m/F	or	$1/4\pi\epsilon_0$
Planck constant	h	$6.6260696 \times 10^{-34}$ J $\cdot$ s	or	$4.1356675 \times 10^{-15}$eV $\cdot$ s
reduced Planck constant	$\hbar$	$1.0545717 \times 10^{-34}$ J $\cdot$ s	or	$6.5821193 \times 10^{-16}$eV $\cdot$ s

Table 1.1: A table of measured constants.

1.2 Pictures and figures

Try to stick to a few data formats if possible. Vector and pdf data types are ideal, because they minimise the overall size of the document and you can zoom/print to any quality without pixelisation.

You can write a program that returns data or a visual representation of it (graph, coloured array, surface plot etc.), and in the latter case, you can take the resulting image and copy to your 'figures' folder (this keeps the internal address conventions neat and tidy) and link the document directly to your report, like so:

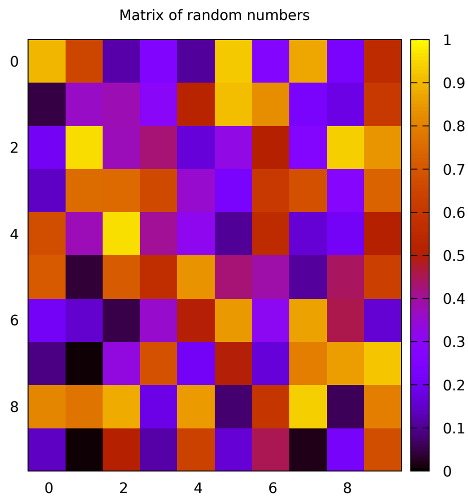


Figure 1.1: A random square matrix, generated by a `Fortran` program.

If you need to provide a diagram of your own creation, there are several options available. You can feel free to choose whichever method your experience has endowed you with, or else represents the best compromise between effort spent and professional appearance of the final product. For instance, you could hand-draw your diagram, scan or take a photo of it, upload the resulting image to the correct directory, crop/filter it and add to your document, as shown in the LHS of Figure (1.2).

Alternatively, you could use some available computer software to get the job done. Here we will give you an example made in Inkscape, because it is an open-source package which is available for most operating systems. Here you will make your diagram (there are plenty of tutorials and examples available online), adjust the document borders to your satisfaction, save a vector-type copy for later editing purposes (for instance, '`diagram.svg`', and then also save a pdf version ('`diagram.pdf`'). It is the second file-type which you can link to your LaTeXdocument with the usual features. The comparison between a hand-drawn and computer-rendered image is fairly stark – see the RHS of Figure (1.2).

If you wish to take the last example further, try exploring some of Inkscape's features. It allows you to plot functions, write mathematical expressions (although this option does not always work) and vectorise/edit a surprisingly large assortment of existing images. If you would like your final document to look seamlessly

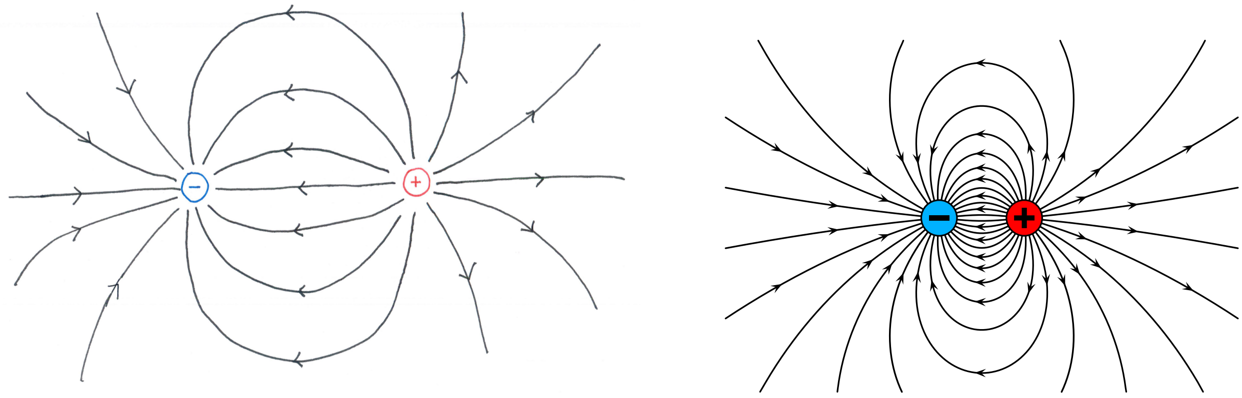


Figure 1.2: A sketch of some electric field lines around an ideal electric dipole.

consistent, you can even make diagrams with text and equations of the same formatting. (This involves writing everything in a dummy LaTeXdocument, exporting and vectorising it online, importing that vector document into Inkscape, playing around with character size compared to final drawing size in the document, and then exporting that diagram as a pdf to be placed back into LaTeX... possibly not a worthwhile pursuit for the purposes of this subject.)

1.3 Computer code

1.3.1 Code input/output

Talk about the language that you used, maybe referencing an integral in [1] or a numerical algorithm in [2] if you used `Fortran`.

1.3.2 Code written verbatim

This can be done with the `listings` and `verbatim` packages. You should modify the preamble document so that these packages correctly interpret the programming language that you used.

```
      FUNCTION PINEWT(N)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C*********************************************************************C
C     PINEWT EVALUATES THE INFINITE SERIES REPRESENTATION             C
C     FOR PI AS A TRUNCATED SUM.                                      C
C*********************************************************************C
C
C     ENSURE N IS A LEGAL VALUE
      IF(N.LT.0) THEN
        WRITE(6,*) 'In PINEWT: illegal value for N.',N
        RETURN
      ELSEIF(N.EQ.0) THEN
        PINEWT = 2.0D0
      ENDIF
C
C     INITIALISE SOME COUNTERS
      TWOPOW = 1.0D0
      FACKNM = 1.0D0
      FACKDM = 1.0D0
      PINEWT = TWOPOW*FACKNM*FACKNM/FACKDM
C
C     LOOP OVER SERIES ORDER N TIMES, ADDING EACH TERM TO TOTAL
      DO I=1,N
        TWOPOW = 2.0D0*TWOPOW
        FACKNM = DFLOAT(I)*FACKNM
        FACKDM = DFLOAT(2*I+1)*DFLOAT(2*I)*FACKDM
        PINEWT = PINEWT + TWOPOW*FACKNM*FACKNM/FACKDM
      ENDDO
C
C     DOUBLE THE RESULT (SEE WIKIPEDIA FORMULA)
      PINEWT = 2.0D0*PINEWT
C
      RETURN
      END
```

Appendix A

Program flowchart

A flowchart is a conceptual aid which highlights input and output that a program should have, as well as the consequences of any decisions that it must make. Sometimes, these will include *pseudocode*, which mimics some of the more common coding structures in a universal way. (For instance, you might say 'loop over all array addresses'.)

This particular flowchart indicates a program that reads three numbers from the terminal, and then produces the largest of the numbers back to the terminal. The colour scheme is not mandatory, but helps the reader to know what is going on at a glance.

This flowchart was drawn with the free 'Lucidcharts' extension to Google Docs, from which the result was exported as an `svg` file, and any unnecessary features were then cropped out with Inkscape. Choose any method you wish to create your own flowchart!

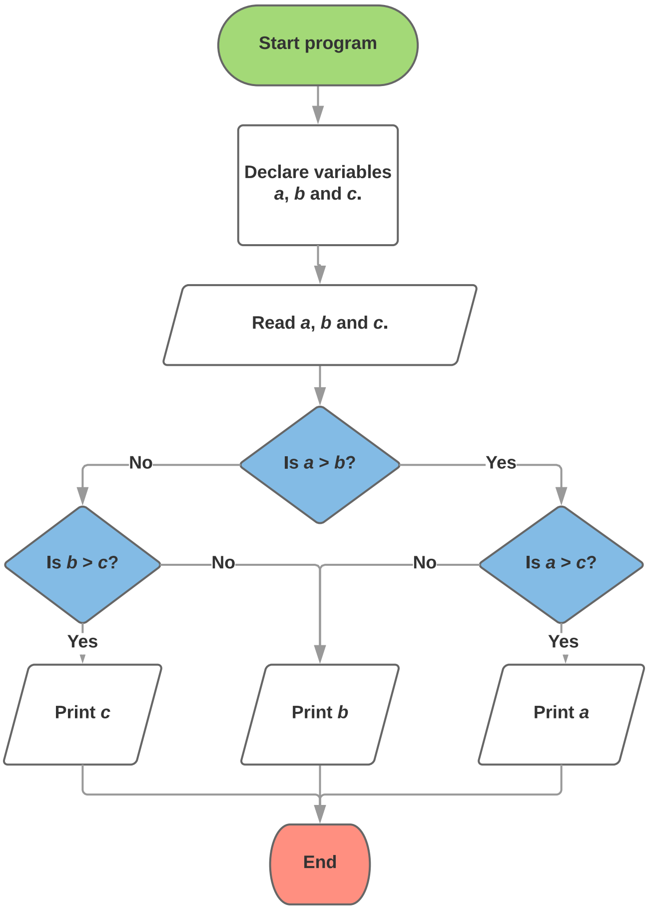


Figure A.1: A flowchart for the determination of the largest of three user-selected numbers.

Bibliography

[1] M. Abramowitz and I. Stegun. *Handbook of Mathematical Functions.* Dover Publishing Inc. New York, 1970.

[2] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes in Fortran (Cambridge).* Cambridge Univ. Press, 1992.