Lab 10 Supplement

1 Hey, you! Read the next sentence

Anything in the text in typewriter font, like this: `print, 'I love IDL'` is a command or line of code you can type right into IDL; if there is a long sequence of commands they will be shown as a separate chunk of text, but still in the same font.[1]

2 Arrays in IDL

Consider the array B shown below

$$B = \begin{pmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \end{pmatrix}. \tag{1}$$

In the notation we have been using in the course, the element B_{23} would mean second row, third column, whose value is 5. In other words, if I write B_{ij}, the first index i is the row number and the second index j is the column number.

To enter the matrix B in IDL you would type the following

```
B = [ [0, 1, 2] $
      [3, 4, 5] $
    ]
```

Notice that I have continued the definition of B across several lines. That is simply to make the definition easier to read.

Unfortunately, if you entered the matrix B as it is written above into IDL and typed `print, B[2,3]` IDL would generate an error message. Understanding why isn't too hard once you know how IDL refers to matrices. In IDL the matrix B is represented using two rules:

1. The first index is the column number and the second index is the row number.

[1] So how did you like that grammatical slight of hand I pulled with the semicolon? The section heading made you think you would only have to read one sentence, but by taking two sentences and joining them with a semicolon I got you to read both. Even better you have now read this entire footnote, an additional three sentences.

2. Numbering for the rows an columns starts at 0, not at 1.

With these rules in mind, we can figure out what `B[2,3]` meant to IDL; it meant look in the third column, fourth row. Why third column? Because the numbering starts at 0, so 2 is the third column. Similar reasoning gets you to the fourth row. There is no fourth row, so IDL was justifiably unhappy.

So how would we print out the element whose value is 5? It is in the third column, which is numbered 2 in IDL, and in the second row, which is numbered 1 in IDL, so the command `print, B[2,1]` does the trick. Note again that we list the column number first and then the row number.

3 Matrix Multiplication in IDL

In what comes next we will consider the matrix B above and the matrix C defined by

$$C = \begin{pmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{pmatrix}, \tag{2}$$

which you can enter into IDL by typing

```
C = [ [0,1], $
      [2,3], $
      [4,5] ]
```

If you multiply B by C you get the matrix

$$BC = \begin{pmatrix} 10 & 13 \\ 28 & 40 \end{pmatrix} \tag{3}$$

The way to get this result in IDL is to type `B ## C`, which you can check with the statement `print, B ## C`. Note that the result you get is still labeled with column first then row, so that if you define `D = B ## C` then `D[0,1]` will print the number 28, not the number 13.

4 Multiplication of a vector and a matrix

Consider the matrix

$$A = \begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} 0 & \\ 0 & 0 & 1 \end{pmatrix} \tag{4}$$

and the vector

$$v = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}. \tag{5}$$

Then the product of the two is

$$Av = \begin{pmatrix} 0 \\ \sqrt{2} \\ 0 \end{pmatrix}. \tag{6}$$

The way to make this happen in IDL is to write `A ## v`.

5 Defining a function in IDL

Last week was your first experience defining a function in IDL. Here I will lay out a bit more formally the steps, and give some advice for writing functions. In this section I will write a function called *RotationY* that returns the matrix representing rotation by angle θ about the y-axis.

First some advice. Put each function you write for IDL into a separate file, name that file the same thing function is called but with '`.pro`' on the end, and only use lower case letters in the file name. Using that convention I would make a file called *rotationy.pro* to write the function we want. Create a new file called *rotationxy.pro* (In xemacs this is done by going to the file menu, then open file, and typing in the name *rotationy.pro*. In the IDL Workbench, you can do this by going to the File menu and selecting "New → IDL Source File.").

The function below, which can be downloaded from the web site, takes one argument, the angle θ in radians.

```
FUNCTION RotationY,  theta
; the angle theta should be in radians


; Note the liberal use of continuation symbols $ to spread the code
; over multiple lines so it is easier to read.
    Ry = [ $
           [ cos(theta),  0,  -sin(theta)], $
           [     0,       1,       0     ], $
```

```
          [ sin(theta),  0,   cos(theta)] $
        ]

    RETURN,  Ry
END
```

To use our new function, try first calculating the matrix for a rotation of $\pi/4$,

```
IDL> ry = rotationy(!pi/4)
IDL print, ry
     0.707107     0.000000    -0.707107
     0.000000      1.00000     0.000000
     0.707107     0.000000     0.707107
```

and then try rotating the vector

$$v = \begin{pmatrix} 1/\sqrt{2} & 0 & 1/\sqrt{2} \end{pmatrix} \tag{7}$$

by that angle. We should end up with a vector along the x-axis, so let's see what IDL says:

```
IDL> v2=[1/sqrt(2),0,1/sqrt(2)]
IDL> print, ry # v2
      1.00000     0.000000     0.000000
```

so it would appear this worked.

The reason for giving the file and the function the same name is that if you run IDL and it encounters the function name *RotationY* it will search for a file of the same name (but ending .pro) and assume the definition of the function is in that file.