Astronomy 142 — Recitation #1

Prof. Douglass

January 19, 2024

Formulas to remember

First-order approximations

All the power series expansions of elementary functions are useful in making approximations, especially

$$\sin x = \sum_{i=0}^{\infty} (-1)^i \frac{x^{2i+1}}{(2i+1)!} = x - \frac{x^3}{6} + \frac{x^5}{120} - \cdots \cong x$$

$$\cos x = \sum_{i=0}^{\infty} (-1)^i \frac{x^{2i}}{(2i)!} = 1 - \frac{x^2}{2} + \frac{x^4}{120} - \cdots \cong 1$$

$$\tan x = \sum_{i=0}^{\infty} \frac{2^{2i+2}\left(2^{2i+2}-1\right) B_i x^{2i+1}}{(2i+2)!} = x + \frac{x^3}{3} + \frac{2x^5}{15} + \cdots \cong x$$

$$\tan^{-1} x = \sum_{i=0}^{\infty} (-1)^i \frac{x^{2i+1}}{2i+1} = x - \frac{x^3}{3} + \frac{x^5}{5} - \cdots \cong x$$

$$e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!} = 1 + x + \frac{x^2}{2} + \cdots \cong 1 + x$$

$$\ln(1+x) = \sum_{i=0}^{\infty} (-1)^i \frac{x^{i+1}}{i+1} = x - \frac{x^2}{2} + \frac{x^3}{3} - \cdots \cong x$$

$$(1+x)^n = \sum_{i=0}^{\infty} \frac{n!}{i!(n-i)!} x^i = 1 + nx + \frac{n(n-1)}{2} x + \cdots \cong 1 + nx$$

The right-most terms in each of these expressions are approximations correct to *first order* in x.

Recitation problems

Remember! These problems that you will do in groups in Recitation are a crucial part of the process of building up your command of the concepts important in ASTR 142 and subsequent courses. Therefore, do not do your work on scratch paper and discard it. Better for each of you to keep your own account of each problem in some sort of bound notebook.

1. Obtain first-order approximations for the following functions:
   (a) $d(x) = \frac{e^{ix}+e^{-ix}}{2}, \quad x \ll 1$
   (b) $f(x) = e^{-ik(r-x)}, \quad r \gg x$
   (c) $g(x) = \cos^2 x + \sin^2 x, \quad x \ll 1$
   (d) $h(x) = P(r+x) - P(r), \quad x \ll r, \quad P$ continuous and differentiable

2. *Using approximations to make differential equations solvable.* Consider a star of mass m that lies within a galaxy which has a mass density given by

$$\rho(r) = \rho_0 \left[1 - \left(\frac{r}{r_0}\right)^2\right]$$

   (a) Suppose that the star has no orbital angular momentum. (So how many variables are then in the equation of motion?) Use Newton's second law to generate the star's equation of motion.

   (b) Suppose that $r \ll r_0$. Make a suitable approximation of the equation of motion and describe its solution (the motion of the star within this galaxy).

   (c) Do you know how to solve the equation without the approximation?

Learn your way around the sky, lesson 1. (An *exclusive* feature of ASTR 142 recitations.) You may find the lab's celestial globes and the program Stellarium useful in answering these questions about the celestial sphere and the constellations.

3. This will be a review for those who have taken ASTR 111:

   (a) *Celestial, spherical, and Cartesian coordinates.* A star has right ascension and declination α and δ. (See Ryden pg. 1–9 if you are unfamiliar with the equatorial coordinate system in which these coordinates are used.) Draw a Cartesian, spherical, and celestial coordinate system with the x-axis pointing through $\alpha = \delta = 0$ and the z-axis pointing towards $\delta = 90°$. Write an expression for a unit vector pointing at the star in terms of the Cartesian unit vectors $\hat{x}$, $\hat{y}$, and $\hat{z}$.

   (b) Recall that the inner ("dot") product of two vectors $\mathbf{A}$ and $\mathbf{B}$ can be written

$$\begin{aligned}\mathbf{A} \cdot \mathbf{B} &= AB\cos\psi \\ &= A_x B_x + A_y B_y + A_z B_z\end{aligned}$$

   where A and B are the magnitudes of the two vectors, ψ is the angle between them, and the subscripts x-y-z indicate the components of the vectors along those Cartesian axes.

   Two stars have celestial coordinates α_1, δ_1 and α_2, δ_2. Use your result from part a to obtain an *exact* expression for the angle between (the directions toward) these stars. Remember this resulting equation: it is one of the two equations that are fundamental for understanding the angular distances along the celestial sphere.

   (Hint: For simplification, recall the trigonometric identity $\cos(a \pm b) = \cos a \cos b \mp \sin a \sin b$.)

4. Use the second order approximation for the cosine of a small angle and the exact solution for the angle between two celestial coordinates derived in the previous problem to obtain an *approximate* form of the angular difference, which bears a strong resemblance to the Pythagorean Theorem of plane geometry:

$$\psi \cong \left[(\delta_1 - \delta_2)^2 + (\alpha_1 - \alpha_2)^2 \cos\delta_1 \cos\delta_2\right]^{1/2} \qquad \text{if } \alpha_1 - \alpha_2, \delta_1 - \delta_2 \ll 1 \tag{1}$$

Intro to Python, lesson 1. (A feature *exclusive* of ASTR 142 recitations.)

5. *Installing Python.* Python is a programming language that is widely used in the scientific community for data analysis. While you can write python programs in text files (the file extension is .py), you will need to install various Python packages to be able to run the code on your computer. The easiest way to have all the various packages work in harmony on your computer is to install Anaconda.

   (a) Go to https://docs.anaconda.com/anaconda/install/ and click on your operating system. Follow the relevant instructions to install Anaconda on your computer. If you get stuck or are confused with something during the process, PLEASE ASK FOR HELP!

   (b) Once Python has been successfully installed, please open a terminal (Mac or Linux users) and type "ipython" at the prompt, or open Spyder (Windows users). Something like this should appear:

```
Python 3.10.13 |Anaconda, Inc.| (default, Oct 06 2023, 08:51:29)
Type 'copyright', 'credits' or 'license' for more information.
IPython 8.18.1 -- An enhanced Interactive Python. Type '?' for help.
```

We can type commands at the prompt.

(c) Let's start simple:

```
In [1]: 5 + 1
```

Does that give you what you would expect? How about:

```
In [2]: 9*2
```

One more:

```
In [3]: 7/3
```

(d) Python can also do algebra. For example:

```
In [4]: x = 4
In [5]: y = 3
In [6]: z = x**y
In [7]: print(z)
```

In this case, we used the power function ($x^y \rightarrow x**y$) as well as the print command. There are many more commands that you will learn with time.

(e) An object can belong to text (a "string") as well as a number:

```
In [8]: name = 'Douglass'
In [9]: print(name)
```

They can also point to more complicated objects known as **lists**:

```
In [10]: a = []
In [11]: a.append(5)
In [12]: a.append(9)
In [13]: print(a)
[5, 9]
In [14]: print(a[0])
5
```

We did two things here. The first was to create a list; the second, to use the "append" command to add an element to the end of the list. To access an element of the list, you need to input which position the element is. It is important to remember that **Python uses 0-based indexing**: that is, the first element in the list has index number 0, the second has index number 1, etc.

(f) The most useful packages for the astronomical community are astropy (http://www.astropy.org/), numpy (http://www.numpy.org/, and matplotlib (https://matplotlib.org/). We will use astropy to import and process data, numpy to do calculations on arrays, and matplotlib to display our data in plots. In order to use any of the functions in either package, we must import the package (or the individual functions) at the beginning of our code. For now, just make sure that you can import these packages without any problems:

```
In [15]: import astropy
In [16]: import numpy
In [17]: import matplotlib
```

Now, if we wanted to access the value of π, we would type

```
In [18]: numpy.pi
Out[18]: 3.141592653589793
```

It is common, though, to assign a nickname to the imported packages (so that we do not have to type the full package name every time). Numpy is often given the nickname "np". So, instead of just "import numpy", we can instead do

```
In [19]: import numpy as np
```

Now, to get π, all we need to type is

```
In [20]: np.pi
Out[20]: 3.141592653589793
```

Both astropy and matplotlib are what we call libraries — they contain many different packages. In order to access any of the functions within one of these contained packages, we need to import the package directly. For matplotlib, we will commonly use

```
In [21]: import matplotlib.pyplot as plt
```

(g) So far, we have imported an entire package and a package within a library. We can also import one specific function or class within a package (this minimizes the amount of memory we are using). Import just the "Table" class from astropy:

```
In [22]: from astropy.table import Table
```

Let's play around with the table functionality in astropy. Tables are "structured arrays" — similar to a list, but they have slightly different capabilities. Make an empty table:

```
In [23]: T = Table()
```

Now add some data:

```
In [24]: T['name'] = ['H_ALPHA', 'H_BETA', 'H_GAMMA']
In [25]: T['wavelength'] = [6563, 4861, 4340]
In [26]: T.pprint()
  name  wavelength
------- ----------
H_ALPHA       6563
 H_BETA       4861
H_GAMMA       4340
```

We have just created a table with two columns of data. Different columns can store different data types (string vs. integer vs. float), but the data type must be consistent *within* a column.

Feel free to continue to experiment with Python. If you want to look at the documentation for something, type

```
In [ ]: help(name_of_thing_you_want_documentation_for)
```

(h) Once you are finished, close out your terminal Python session by typing:

```
In [ ]: exit()
```

If you are on Windows and/or are using Spyder, just close the program using the "X" in the upper corner of the window.