PHY392, 2013
Problem Set 2
Assigned: Feb. 15th; Due: Mar. 5th

Computational Problem Set

Introduction

To complete this problem set, it would be helpful to have some experience in Python, hopefully in one of your previous physics courses! If you have not used Python before, go to the Department of Physics compwiki (`http://compwiki.physics.utoronto.ca/`) for an introduction, as well as installation instructions. Even if you have used Python before, the compwiki is a good resource for programming help. If you have Python-related questions about running the scripts in this problem set, you can contact the course TA, Oliver Watt-Meyer (`oliverwm@atmosp.physics.utoronto.ca`), for assistance. For evaluation you should hand in the requested plots together with your written answers to each question. This is probably most easily done in a LaTeX or Word document.

1 Runaway greenhouse effect

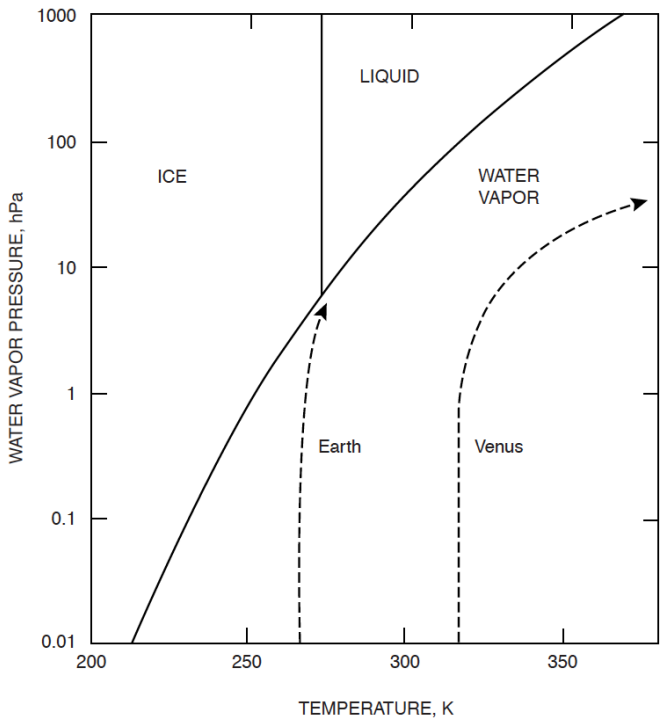
In the n-layer greenhouse model, $T_s = (n+1)^{1/4} T_e$, the increase of the surface temperature with the number of layers reflects the influence of increased abundances of greenhouse gases in the atmosphere on the surface temperature. Lets assume that the number of layers is a power of the surface temperature, $n(T_s) = \alpha T_s^{\beta}$. This captures, for example, the rapid increase of the saturation vapour pressure of water in the atmosphere with temperature.

(a) Determine the values of α appropriate for current conditions $T_s = 288K$ and $T_e = 255K$, for $\beta = 2$ and $\beta = 4$.

(b) Using these values of α, plot T_s as a function of T_e (over the range $150-350$ K) for $\beta = 2$ and $\beta = 4$. You can use Python or any plotting software to make these plots. Would you characterize the dependence of T_s on T_e for $\beta = 2$ as an example of a runaway greenhouse effect? What about for $\beta = 4$? The saturation vapour pressure of water actually increases exponentially with temperature.

(c) The high surface temperature on Venus is often described as an example of a runaway greenhouse. Calculate the emission temperature of Venus assuming a planetary albedo of 0.15, which might be expected in the absence of clouds early in the planets history.

(d) The figure below shows the evolution of temperatures in the early atmospheres of Venus and Earth, together with the phase diagram for water. Using this figure, explain why the Earth did not experience a runaway greenhouse as water accumulated in the atmosphere from outgassing from the planets interior.



2 Problem 3.34 from the textbook

In equilibrium, the absorbed solar radiation $L_o(1-\alpha)/4$, where α represents the planetary albedo, is balanced by the outgoing long-wave radiation (OLR) at the top of the atmosphere. Calculations with a complete radiative transfer calculation indicate that, for a CO_2 concentration of 300 ppmv, and with a atmosphere on the moist adiabat, a reasonable fit to the actual OLR curve in the range of 220 K to 310 K is the linear fit $OLR(T) = a + b(T - 220)$ where $a = 113$ W/m^2 and $b = 2.177$ W/m^2 K. In this problem you will compute the ice-albedo hysteresis diagram giving the set of equilibrium temperatures as a function of the solar constant L_o. Note that there is a simple trick for getting the bifurcation plot. The equation determining the equilibrium is $\frac{1}{4}L_o(1 - \alpha(T)) = OLR(T)$. Instead of specifying L_o and finding the T that satisfy the equation, we can rewrite the equation as

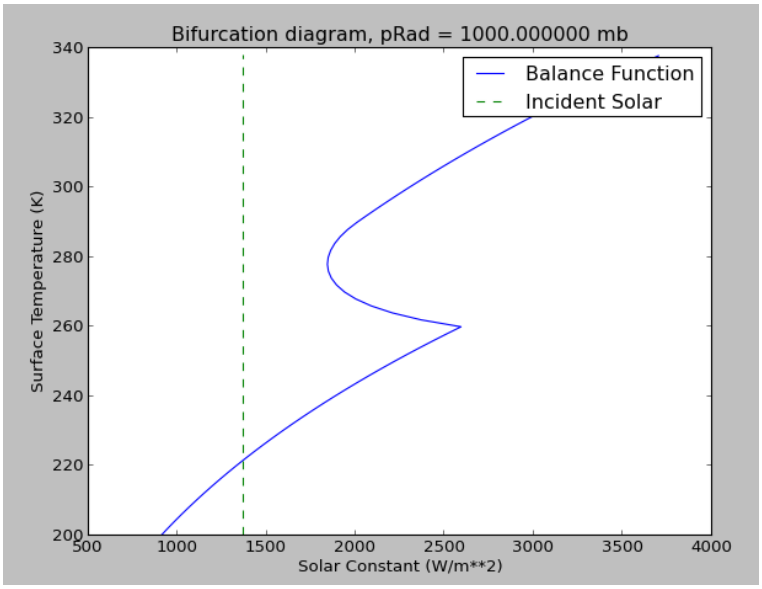
$$L_o = 4\frac{OLR(T)}{1 - \alpha(t)}.$$

Now, if we call the right hand side $G(T)$, then $G(T)$ gives the unique value of L_o which supports the temperature T. Hence, to get the bifurcation diagram, you can just plot $G(T)$ and then turn it sideways. The Python script `IceAlbedoZeroD.py` calculates and plots the hysteresis diagram using the albedo-temperature given in Equation (3.9) in the textbook, with an albedo of 0.2 for an ice-free Earth and 0.6 for an ice-covered Earth. The script also makes other plots that are useful in understanding the bifurcations. You will have to modify the script to use the linear relationship for $OLR(T)$ between $220 - 310$ K described above. Shown below is an example of the bifurcation plot produced by the script for P_{rad} of 1000 hPa. You will need to change P_{rad} (the variable **pRad**) in the script to a value that is appropriate for current atmospheric composition.

(a) Based on your calculations with `IceAlbedoZeroD.py`, if CO_2 were held constant how much would L_o have to be reduced from its modern value before Earth was forced to fall into an inevitable Snowball state?

(b) Using the inverse-square law for the intensity of the Sun, and assuming a circular orbit, compute how far out from the Sun the Earth would have to be displaced (relative to its present orbit) to achieve this solar constant.

(c) Conversely, how close to the Sun would you have to place the Earth before a Snowball state became impossible? How does this distance compare to the distant of Venus from the Sun?



3 The Budyko model

In this problem set you will use a Python implementation (by Oliver Watt-Meyer) of the energy balance model based on [1]. This model starkly demonstrates a possible non-linearity in the climate system due to the ice–albedo positive feedback. You should read [1] before completing this problem set. Briefly, Budyko proposes a simple one-dimensional (depending only on latitude) model that assumes the total energies absorbed and emitted by the Earth are equal. The Earth is assumed to be covered only by ocean and sea ice, and the values for a number of parameters such as the albedo for the ocean versus for sea ice and the solar insolation as a function of latitude are estimated. The result is an implicit equation for temperature as a function of latitude.

(a) Open and run the script `BudykoEnergyBalanceModel.py`. You should see a figure in which temperature as a function of latitude is plotted. This is the calculated temperature distribution based on the default reduction of solar insolation, -2%. Try adjusting the variable **delQp** to see what happens to the temperature distribution for different changes in the solar insolation. Does **delQp**=0.0 give a realistic temperature distribution for the present day Earth? You should make plots for **delQp** = 0%, -2%, -4%, -6% and -8% and comment on how the dependence of the temperature on **delQp**. Does the global mean temperature vary smoothly as you change **delQp**?

(b) Set **delQp** equal to zero. Now, try adjusting the values of **albedoOcean** and **albedoIce**. How do the surface temperatures respond? You should Include these plots in your solutions.

(c) Calculate the mean global temperature and ice edge boundary as a function of **delQp**, the change in solar insolation. Essentially, we want to recreate Figure 5 from [1]. The script `BudykoEnergyBalanceModel.py` calculates the variables **Tp** and **iceEdge** which are the global mean temperature and latitude of sea

ice edge respectively. You should adapt this script so that it calculates these variables for an array of **delQp** values (say, from 0% to -8%, with intervals of 0.1%), and then plots them.

To do this, create the array of changes in insolation by calling something like

```
delQp_list = arange(0,-8.1,-0.1)
```

and add a `for` loop around the main part of the code that solves for the temperature distribution. You should make sure to save the values of **Tp** and **iceEdge** for each value of **delQp** so that you can make the appropriate plots.

(d) Include plots of both **Tp** and **iceEdge** as a function of **delQp**. At what percentage decrease in solar insolation does the ice edge go to the equator, i.e. we get "snowball Earth" conditions? Does this agree with the value found in [1]? How does this compare with the value found in Problem 2a?

(e) Now, try adjusting the albedos as in Part (b) and see what effect this has on the figures from the previous question. Comment on the differences.

References

[1] Budyko, M. I. (1969), The effect of solar radiation variations on the climate of the Earth, *Tellus*, **21**, 611-619.