Stellar Atmospheres Radiative Transfer Computer Exercises

Mandy Hagenaar, Thijs Krijger and Alfred de Wijn

June 10, 2003

Contents

Chapter 1

Getting Started

1.1 VALIII Atmosphere

We consider the temperature structure of a VALIII atmosphere. VALIII atmospheric parameters can be found at `http://www.astro.uu.nl/~dwijn/teaching/afyc/`.

1.1 Copy the files `valiii.txt` and `valiii_rw.pro` to your home-directory and read it by running `VALIII_READ` in `valiii_rw.pro`. Which variables did you restore from the file?

1.2 Make a plot of temperature as a function of height. Specify an x-range from 2500 to -500 km and a logarithmic y-axis. Add titles to the axes, expressing the temperature in reasonable units.

1.3 What is the minimum temperature?

1.4 Determine the height where the temperature is minimal, using the `WHERE` statement. Indicate this height in your figure with the command `OPLOT`. Indicate photosphere, chromosphere and corona in your plot.

1.5 What's the corona's temperature range?

1.2 Ca II K line

A strong line in the chromosphere is the Ca II K line (3933 Å). Why we can observe the chromosphere (and the photosphere) in this line will become clear throughout this course.

The Ca II K line is used often to trace magnetic flux tubes from the photosphere to the chromosphere and higher altitudes. The line does not discriminate between magnetic polarity. An example of an observation is available in `caiik.txt` in `http://www.astro.uu.nl/~dwijn/teaching/afyc/`.

Figure 1.1 shows an image from the same sequence of observations in Ca II K, and a Kitt Peak magnetogram, taken at the same time.

1.6 Copy this file and the program `caiik_rw.pro`.

1.7 What are the dimensions of the image (in pixels)? Show the image on your screen.

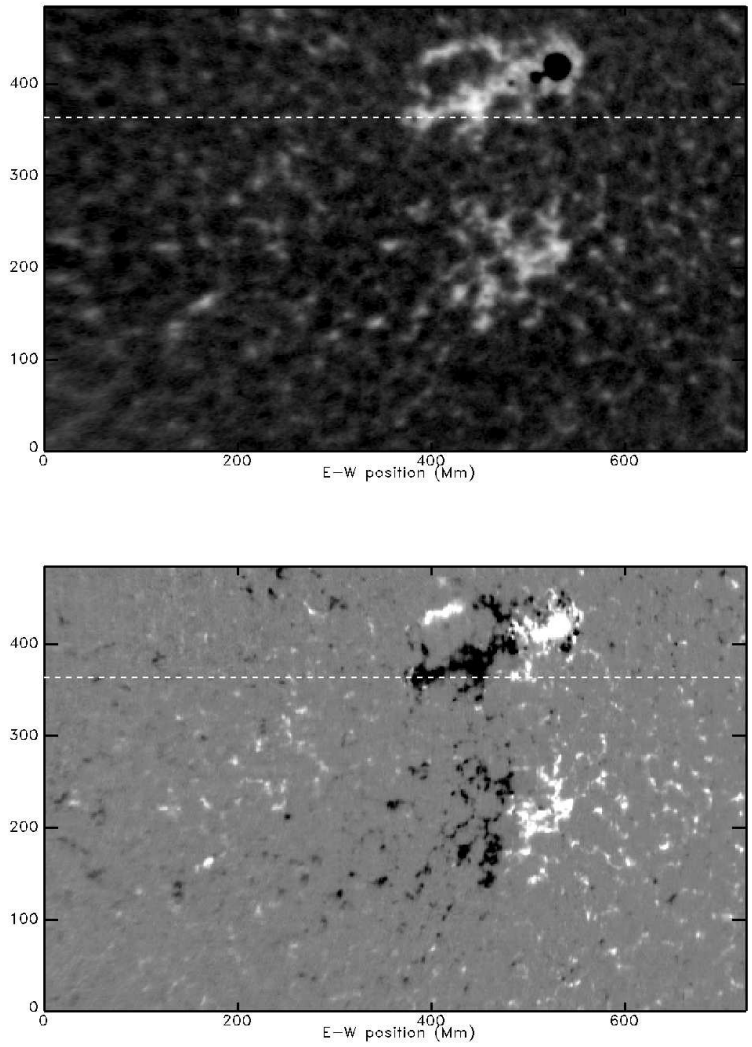

Figure 1.1: Upper panel: Ca II K filtergram, remapped to disk center, observed at 17:30 UT on Dec 21, 1994. The horizontal and vertical axes are aligned with the E–W and N–S directions, respectively. The dashed line represents the equator. Lower panel: Remapped NSO/Kitt Peak magnetogram corresponding to the region analyzed in the Ca II K image sequence. The full-disk magnetogram from which this field was extracted was observed on Dec 21, 1994, between 17:01:22 and 17:56:01 UT. The remapping procedure developed for the Ca II K filtergrams was also used on this magnetogram.

1.8 This observation has been recorded on a CCD, from the geographic south pole in December 1994. Besides two active regions of enhanced magnetic field or *plage*, the chromospheric network is visible.

1.9 The pixels of the CCD are 1550×1550 km^2. Express this in arc-seconds.

1.10 What are the dimensions of the image in km and in heliographic degrees? Assume the sun has a radius of 6.96×10^8 m.

The contrast of the image is very low. To determine the minimum and maximum brightness you can use the IDL commands `MIN(image)` and `MAX(image)`.

1.11 Plot the histogram of brightnesses `y0` with the IDL function `y0=HISTOGRAM(image)`. How many elements has `y0`?

1.12 Now define `y1=HISTOGRAM(image,min=0)`. How many elements has `y1`? Can you explain the difference?

1.13 What is the most occurring value? Determine this automatically with the `WHERE` statement in combination with `HISTOGRAM(image,min=0)`.

The applied filter transmits a 1 nm bandpass centered on the Ca II K line. The line-center is chromospheric, the wings are photospheric. Fluctuations in the photospheric component in time and space have been filtered out so that the remaining component is homogeneous. This component contributes a constant brightness in each pixel.

1.14 Consider every signal below the threshold value 1970 as photospheric. Show a *thresholded* image on your screen. Vary the threshold until you think the contrast is optimal.

1.15 The chromospheric network is found to correspond to the boundaries of convection cells (i.e., centers of upflow and cell boundaries, where the gas flows down) that can be recognized in the photosphere: the supergranulation. Can you recognize some of these cells? Estimate the order of size of these supergranular cells.

Chapter 2

The Inversion Problem

From Lecture Notes by Phil Judge, Summer School on Radiative Transfer and Radiation Hydrodynamics, Institute of Theoretical Astrophysics, Oslo, 1995.

In this exercise we construct a line profile in a simple but realistic model atmosphere and then compute emergent intensities. Next we assume the intensity is given and try to reconstruct the temperature profile of the atmosphere.

2.1 Gray Atmosphere

A Milne-Eddington atmosphere is an atmosphere in which radiative equilibrium is valid. We assume gray opacities, LTE for the source function and use the Eddington approximation to find

$$T^4(\tau) = \frac{3}{4} T_{\rm eff}^4 \left(\tau + \frac{2}{3} \right) , \tag{2.1}$$

$$S(\tau) = J(\tau) = B(\tau) = \frac{\sigma_{\rm R}}{\pi} T^4 , \tag{2.2}$$

where $T_{\rm eff} = 5770$ K and $\sigma_{\rm R} = 5.67 \times 10^5$ erg cm^{-2} sec^{-1} deg^{-4}.

2.1 Construct an array of 51 gray optical depth points distributed logarithmically uniform between $\tau = 10^{-4}$ and $\tau = 10$.

2.2 Define $T(\tau)$ and $S(\tau)$ as functions of τ.

2.3 Plot S as a function of τ.

2.2 Line Profile

2.4 Define a frequency variable y describing the distance in Doppler widths from line center, varying from -5, -4.5, -4, ..., 5. Use $\phi(y) = {\rm e}^{-y^2}$ for the Doppler line profile function normalized to unity at line center.

2.5 Plot the profile function.

The non-gray optical depth is now defined as

$$\begin{aligned} \tau_2(x,y) &= 10\,\phi(y)\,\tau(x) + \tau(x) \\ &= \big(10\,\phi(y) + 1\big)\tau(x)\,, \end{aligned} \tag{2.3}$$

so that the opacity in the line center (at $y = 0$) is eleven times greater than the gray opacity $\tau(x)$.

2.6 In IDL, define a matrix `tau2` containing the non-gray optical depth.

2.7 What is the formal integral solution of the transport equation

$$\frac{\mathrm{d}I_\nu}{\mathrm{d}\tau_\nu} = I_\nu - S_\nu \tag{2.4}$$

for the emergent intensity?

2.8 Write the integral equation as a summation of differences[1].

Remember, we assume that the source function depends on temperature and *gray* optical depth only, and is frequency-independent.

2.9 Derive the general form

$$\mathbf{I} = \mathbf{K}\ \mathbf{S}\,, \tag{2.6}$$

where $\mathbf{I}$ is the denoting emergent intensity as a function of frequency, $\mathbf{S}$ is the source function as a function of depth, and $\mathbf{K}$ is the so-called *kernel* matrix.

2.10 Compute the emergent intensity as a function of frequency. Plot $\mathbf{I}$ as a function of frequency.

2.3 The Eddington-Barbier Approximation

In this exercise we pretend that the source function is unknown (as in real life) and that all we know is the outgoing surface intensity. We want to find the source function from the intensity and Eq. 2.6. A first approximation is the Eddington-Barbier relation:

$$S_\nu(\tau_\nu = \mu) \approx I_\nu^+(\tau_\nu = 0,\, \mu)\,, \tag{2.7}$$

which is exact when S_ν varies linearly with τ_ν. Here we assume $\mu = 1$ so that $S_\nu(\tau_\nu = 1) \approx I_\nu^+(\tau_\nu = 0)$.

τ_2 is a function of gray optical depth τ and frequency ϕ. If we look at a certain gray optical depth τ there might be a frequency ϕ for which the corresponding line optical depth $\tau_2(\tau, \phi) \approx 1$.

[1] Use:

$$\begin{aligned} \Delta\tau_2(x,y) &\approx \frac{\mathrm{d}\tau_2(x,y)}{\mathrm{d}\tau(x)}\,\frac{\mathrm{d}\tau(x)}{\mathrm{d}\log_{10}\tau(x)}\,\Delta\left(\log_{10}\tau(x)\right) \\ &= (10\,\phi(y)+1)\,\tau(x)\,\log_{\mathrm{e}}(10)\,\Delta = \log_{\mathrm{e}}(10)\,\tau_2(x,y)\,\Delta\,, \end{aligned} \tag{2.5}$$

where $\Delta = \Delta\left(\log_{10}\tau(x)\right) = 0.1$ is constant.

2.11 Find this frequency for each individual gray optical depth if possible. For the approximation, assume any τ_2 between 0.5 and 1.5 is valid[2]. What is the corresponding emerging intensity at this frequency? There are two frequencies because the line profile is symmetric. Take the lower frequency.

2.12 Our source function is gray and only depends on the continuum optical depth τ. Apply the Eddington-Barbier relation and plot the gray optical scale depth against the derived source function, in units of 10^{20} erg cm^{-2}. Overplot the original source function. Add titles to the axes with the correct units, and save your plot as a postscript file.

The emerging intensity across the line profile originates from different depths. If we consider the intensity at a certain frequency ν, we know this intensity roughly equals the source function at a line optical depth $\tau_\nu = 1$, with a corresponding continuum optical depth τ. The gray source function at this gray optical depth is representative for the emerging intensity.

2.4 Inversion

Another way to derive the source function from the emerging intensity is by inversion of relation 2.6. However, because $\mathbf{K}$ is not a square matrix it has no inverse. We circumvent this problem by multiplying both sides of Eq. 2.6 with the transpose of $\mathbf{K}$, $\mathbf{K}^{\mathrm{T}}$. The matrix $\mathbf{K}^{\mathrm{T}}\,\mathbf{K}$ is square and invertible.

2.13 Express $\mathbf{S}$ in terms of $\mathbf{K}$ and $\mathbf{I}$ and perform this calculation in IDL. Use the standard IDL functions `TRANSPOSE` and `INVERT`.

2.14 Is multiplying with $\mathbf{K}^{\mathrm{T}}$ the only possibility in order to get a square matrix, i.e., is the solution unique?

2.15 Plot the 'inverted' source function over the original source function and explain the difference.

[2]You might use something that looks like

```
; the y index (of phi) we are searching for
yind=INTARR(51)
; create difference matrix with absolute difference between tau2 and 1
tau1=ABS(tau2-1)
; check for all 51 tau-indices where the difference matrix is at minimum
; and less then .5 (thus between .5 and 1.5)
FOR i=0,50 DO $
    yind[i]= MIN(WHERE((tau1[i,*] EQ $
        MIN(tau1[i,*])) AND $
        (tau1[i,*] LT .5)))
; take only tau indices where you found frequencies
tauind=WHERE(yind GE 0)
; take tau's for which you found a frequency
foundtau=tau(tauind)
; take intensity at the corresponding frequency
sedd=int(yind(tauind))
; plot the 'found' source function as function of 'found' tau
PLOT,foundtau,sedd,/PSYM
```

We see that small errors in $\mathbf{I}$ can give rise to high frequency instabilities in the solution. The problem with the obtained solution is that it is not unique: $\mathbf{K}$ could be multiplied by any matrix with the same dimensions as $\mathbf{K}^{\mathrm{T}}$, and the amount of information contained in the resulting square matrix is limited to the number of independent measurements. A whole family of results is possible. The results may look very instable and have large fluctuations, which is caused by a lack of data. An optimum inversion depends on the properties of the unknown solution, which can not be determined from the data alone (see Craig & Brown, 1986, Inverse Problems in Astronomy, Adam Hilger Ltd, Bristol, UK).

2.5 Regularization

In the method of regularization the extra information required to stabilize the inversion is introduced by way of forcing an arbitrary 'smoothness condition' on the source function, requiring that $\mathbf{H}\,\mathbf{S}$ is minimal, with $\mathbf{H}$ some suitable chosen linear operator. To accept a found $\mathbf{S}$ as a good solution we also want the difference between $\mathbf{K}\,\mathbf{S}$ and $\mathbf{I}$ to be minimal. The original ill-posed problem $\mathbf{K}\,\mathbf{S} = \mathbf{I}$ is thus replaced by the minimization of

$$\|\mathbf{K}\,\mathbf{S} - \mathbf{I}\|^2 + \lambda\,\|\mathbf{H}\,\mathbf{S}\|^2\,, \tag{2.8}$$

where λ is a positive regularization parameter. However we can still choose $\mathbf{H}$ freely and thus find many possible solutions. A suggested choosing for $\mathbf{H}$ is to minimize the norm of the second derivative so that $\mathbf{H}\,\mathbf{S} = \mathbf{S}''$. This choice is very natural since by considering only the most smooth function (hence the name 'smoothness condition') we ignore oscillatory or discontinuous solutions. This yields

$$\|\mathbf{K}\,\mathbf{S} - \mathbf{I}\|^2 + \lambda\,\|\mathbf{S}''\|^2 = \text{minimal}\,. \tag{2.9}$$

For a Euclidean vector norm a discretized approximation to Eq. 2.9 is given by

$$\sum_{i=1}^{m}\left(\sum_{j=1}^{n} K_{ij}\,S_j - I_i\right)^2 + \lambda\sum_{j=1}^{n}(S_{j+1} - 2S_j + S_{j-1})^2 = \text{minimal}\,. \tag{2.10}$$

with $\mathbf{S}''$ replaced by second order differences (see Lecture Notes). Differentiating Eq. 2.10 with respect to S_k and expressing the result in matrix form yields

$$\left(\mathbf{K}^{\mathrm{T}}\,\mathbf{K}\right)\,\mathbf{S} - \mathbf{K}^{\mathrm{T}}\,\mathbf{I} + \lambda\,\mathbf{H}\,\mathbf{S} = 0\,, \tag{2.11}$$

where some constants have been included into λ. The regularization matrix $\mathbf{H}$ can now be determined since

$$\frac{\partial}{\partial S_k}\sum_{j=1}^{n}(S_{j+1} - 2S_j + S_{j-1})^2 = \lambda\mathbf{H}\,\mathbf{S} \tag{2.12}$$

For $k = 1$, 2, $n-1$ and n, end conditions apply, so that

$$\frac{\partial}{\partial S_1}\left(S_3 - 2S_2 + S_1\right)^2 = 2(S_3 - 2S_2 + S_1) \qquad k = 1\,, \tag{2.13}$$

$$\frac{\partial}{\partial S_2}\left[\left(S_3 - 2S_2 + S_1\right)^2 + \left(S_4 - 2S_3 + S_2\right)^2\right] = 2\left(S_4 - 4S_3 + 5S_2 - 2S_1\right) \qquad k = 2\,, \tag{2.14}$$

$$\frac{\partial}{\partial S_{n-1}}\left[\left(S_n - 2S_{n-1} + S_{n-2}\right)^2 + \left(S_{n-1} - 2S_{n-2} + S_{n-3}\right)^2\right] = 2(-2S_n + 5S_{n-1} - 4S_{n-2} + S_{n-3}) \qquad k = n-1\,, \tag{2.15}$$

$$\frac{\partial}{\partial S_n}\left(S_n - 2S_{n-1} + S_{n-2}\right)^2 = 2(S_n - 2S_{n-1} + S_{n-2}) \qquad k = n\,, \tag{2.16}$$

while for $2 \le k \le n-1$ we have

$$\frac{\partial}{\partial S_k}\left[\left(S_k - 2S_{k-1} + S_{k-2}\right)^2 + \left(S_{k+1} - 2S_k + S_{k-1}\right)^2 + \left(S_{k+2} - 2S_{k+1} + S_k\right)^2\right] = 2(S_{k+2} - 4S_{k+1} + 6S_k - 4S_{k-1} + S_{k-2})\,. \tag{2.17}$$

2.16 Write down the regularization matrix $\mathbf{H}$ and write an IDL procedure that returns $\mathbf{H}$ as a function of its dimension.

2.17 Invert the regularized matrix $\mathbf{K}^{\mathrm{T}}\,\mathbf{K} + \lambda\mathbf{H}$ in your IDL program. Vary λ from 8×10^{-5} to 8×10^{-4} and find the best value for λ (e.g., through the χ^2-method).

2.18 Plot the real source function against τ and then overplot the inverted source functions.

2.19 Repeat this exercise twice, adding 1% and 10% random noise of to each intensity. Comment upon the three solutions obtained. What is the effect of adding some noise to the 'data'? Finally, think about what we needed to impose on the inverted solutions in order simply to obtain any inverted solution at all.

Chapter 3

The Newton-Raphson Method

In this exercise we study the Newton-Raphson method. This method is based on Taylor expansion

$$\begin{aligned} f(x) &= \sum_{n=0}^{\infty} \frac{1}{n!} f^{(n)}(x_0)\,(x-x_0)^n \\ &= f(x_0) + \frac{\partial f}{\partial x}(x_0)\,(x-x_0) + \mathcal{O}\left((x-x_0)^2\right)\,, \end{aligned} \tag{3.1}$$

where $f^{(n)}(x_0)$ denotes the n-th derivative of $f(x)$ evaluated in x_0. To first order, this yields

$$f(x) - f(x_0) \approx \frac{\partial f}{\partial x}(x_0)\,(x-x_0)\,, \tag{3.2}$$

or

$$f(x_0+\delta x) - f(x_0) \approx \frac{\partial f}{\partial x}(x_0)\,\delta x_0\,. \tag{3.3}$$

Suppose we have a first estimate x_0 for a certain solution of $f(x) = C$ and we want to find δx for which $f(x_0+\delta x)$ approaches C. Substitution of C in Eq. 3.3 yields

$$C - f(x_0) \approx \frac{\partial f}{\partial x}(x_0)\,\delta x_0\,, \tag{3.4}$$

so that, to first order,

$$\delta x_0 = \frac{C - f(x_0)}{\frac{\partial f}{\partial x}(x_0)}\,. \tag{3.5}$$

Now we have a second estimate $x_1 = x_0 + \delta x$. Iterating yields a δx_1 such that

$$\begin{aligned} x_2 &= x_1 + \delta x_1 \\ &= x_1 + \frac{C - f(x_1)}{\frac{\partial f}{\partial x}(x_1)}\,. \end{aligned} \tag{3.6}$$

In many cases the estimate converges to a stable solution after several iterations. For example, take $f(x) = x^2$. Suppose we want to know the solution x for $f(x) = 2$, which is of course $x = \sqrt{2} \approx 1.414213562$. We start with a rough estimate $x_0 = 1$.

3.1 Use the Newton-Raphson method to calculate $\sqrt{2}$ up to the 5th decimal. How many iterations do you need?

3.2 Now start with a first estimate not even close to $\sqrt{2}$, for example $x_0 = 100$. How many steps do you need in this case to approach the real value up to the 5th decimal?

3.3 Repeat this exercise for two values of $x_0 < 0$. What happens and why? Can you think of a way to solve this?

3.4 A classic example in chaos theory is the application of Newton's method to find the roots of $x^4 = 1$. How many solutions are there in the complex space? Write down Eq. 3.5 for this function.

Now consider the whole complex space. We start with a first guess $x_0 = a + b\,i$, for the solution of $x^4 = 1$. This point can be represented in a plane with $(x, y) = (a, b)$. In `newt_raphs.pro`, which is available from `http://www.astro.uu.nl/~dwijn/teaching/afyc/`, you find a procedure to calculate the values of each point in the complex plane after several iterations.

3.5 Copy this procedure and, after studying and understanding the code, run it. The procedure pauses every 5 iterations and draws those points that have converged to one of the exact solutions[1]. The color of each point corresponds to the solution to which it converges.

3.6 Change the procedure so that the number of points and the range for which the result is plotted are variable.

3.7 Look only at one quadrant, and study it using a finer grid. Is the result unexpected? What happens if you 'zoom in' again and take an even finer grid?

The Newton-Raphson method is very powerful: the number of decimals usually doubles with every iteration. For higher-order polynomials, especially those with multiple complex solutions, the situation is far more complicated. Intuitively one expects a first estimate x_0 to converge to the closest solution, so that two points infinitesimally close to one another converge to the same solution. In reality this is not true. We have seen that two very close points in complex space can behave extremely different. Some points converge to one of the solutions after a small number of iterations, while others seem to jump around randomly or even end in a loop and never reach a solution. This is one of the subjects of chaos theory. A nice fractal picture of the behavior of complex space under a Newton approximation of the function in the exercise can be found in J. Gleick, 1987, Chaos: making a new science, Viking, New York. The point we make here is that the Newton approximation is very useful, but we should be aware of possible nonlinear effects.

[1]Note that the number of steps depends on the accurateness of IDL. Even though the procedure uses *double-precision* variables, the precision is finite. Values close to a solution will be found as converged to a that solution.

Chapter 4

The Feautrier Method

4.1 Feautrier Method by hand

In this exercise we apply the Feautrier method by hand, to become familiar with this method to solve the transport equation.

In static, plane-parallel media the transfer equation can easily be put into the Feautrier form

$$\frac{\mathrm{d}^2 P}{\mathrm{d}\tau^2} = P - S\,, \tag{4.1}$$

where S is the source function, $P = \frac{1}{2}(I_\nu^+(\tau,\mu) + I_\nu^-(\tau,\mu))$ and τ is the monochromatic optical depth along a ray at angle μ.

4.1 Write Eq. 4.1 as a difference equation. Assume a homogeneous distribution of τ_i, i.e., $\Delta\tau_{i+\frac{1}{2}} = \tau_{i+1} - \tau_i = \delta$.

4.2 Write the first-order difference equation for boundary conditions $\tau \approx 0$ and $\tau \gg 1$ (see Lecture Notes).

4.3 Express P_1 in P_2 (and P_2 in P_3 if you have time left) and see that the equations for P_i become complicated. This method can however easily be applied with a computer.

4.2 Feautrier Program

Straightforward discretization of the transfer equation for an optical depth grid $\tau_i = 1, \ldots, n$, leads to the tridiagonal system (see Lecture Notes)

$$-A_i\, P_{i-1} + B_i\, P_i - C_i\, P_{i+1} = S_i\,. \tag{4.2}$$

The coefficients for $i = 1$ and $i = n$ depend on the chosen boundary conditions.

We use IDL to evaluate monochromatic intensities I^+ and I^- and emergent intensity along a ray with given optical depth. The IDL function in `feautrier.pro` by Han Uitenbroek (1991) (from `http://www.astro.uu.nl/~dwijn/teaching/afyc/`) follows a second order Feautrier method and a bidiagonal system, described below.

The program introduces auxiliary variables K_i, F_i, and Z_i in order to preserve numerical precision[1] (see Appendix A of G.B. Rybicki & D.G. Hummer, 1991, A&A 245, 171–181 and next section). The discretized transport equation is transformed into a bidiagonal system that can be solved iteratively

$$P_i = (1 + F_i)^{-1} P_{i+1} + Z_i \,. \tag{4.3}$$

A simple improvement to first-order boundary conditions can increase accuracy by a factor up to 40, by taking into account second-order terms (see L. Auer, 1967, ApJ 150, L53–55). Instead of the upper boundary condition for $\tau \ll 1$,

$$\frac{\mathrm{d}P}{\mathrm{d}\tau}(\tau_0) = P(\tau_0) \,, \tag{4.4}$$

we use a second order Taylor approximation for P_1.

4.4 Give this second order Taylor approximation if you consider P_1 to be P_0 with a small deviation. Then express P_0' and P_0'' in I_0^- and S_0 (see Lecture Notes), so that P_1 can be expressed in I_0^-, S_0 and P_0.

In the IDL program[2] the quantities R_0 (`r0`), H_0 (`h0`), R_n (`rn`), and H_n (`hn`) represent the boundary conditions at respectively $\tau(0)$ and $\tau(n-1)$. The values depend on the geometry of the problem you want to solve and the illumination at $\tau = \tau(0)$ and $\tau = \tau(n-1)$.

R_0 and H_0 dictate I_0^-, the ingoing intensity $I_0^- = R_0\, I_0^+ + H_0$, at $\tau(0)$, i.e., at the first grid point. With $P_0 = \frac{1}{2}\,(I_0^+ + I_0^-)$, we can eliminate I_0^+ and find

$$I_0^- = \frac{2R_0\, P_0 + H_0}{1 + R_0} \,. \tag{4.5}$$

Equivalently, R_n and H_n dictate I_n^+, the outgoing intensity at the last grid point $\tau(n-1)$, $I_n^+ = R_n\, I_n^- + H_n$.

4.5 Follow the same argumentation as above and express I_n^- in P_n, H_n and R_n.

Assume the radiation is thermalized and isotropic at the deepest point in a half-infinite plane-parallel space.

4.6 What does this mean for I^+ and I^- at the deepest point $\tau(n-1)$? What are R_n and H_n in this case?

4.7 What are R_0 and H_0 if the medium is illuminated at $\tau = \tau(0)$ with illumination I_{ill}?

If the slab is symmetrical with the same illumination at both sides, it is only necessary to consider half of the slab, and assume the middle of the layer is at $\tau(n-1)$.

4.8 What does this mean for H_n and R_n?

4.9 Use the optical depth scale and source function from Sect. 2.4. Use the Feautrier program to determine the emergent intensity for a half-infinite isotropic layer without illumination.

4.10 Why is the emergent intensity computed with the Feautrier program not equal to $I^+(0)$?

[1] Small optical depth increments $\Delta\tau$ can cause numerical problems in the B_i coefficient when the first term becomes lost to machine precision compared to the second term, which is of order $\Delta\tau^{-2}$.

[2] The `R` used in this program is ***not*** the R used in the Lecture Notes.

Chapter 5

Λ-Iteration

The Laplace transform of the formal solution of the transfer equation defines the **Λ**-operator (see Lecture Notes). In the following exercises, the transfer equation is solved by **Λ**-iteration with the Feautrier method as a **Λ**-operator.

5.1 Angle-Quadrature

In the previous exercise the Feautrier transport equation

$$\mu^2 \frac{\mathrm{d}^2 P(\tau,\mu)}{\mathrm{d}\tau^2} = P(\tau,\mu) - S(\tau)\,, \tag{5.1}$$

with τ the monochromatic optical depth along a ray at a certain angle μ, had the form

$$\frac{\mathrm{d}^2 P(\tau)}{\mathrm{d}\tau^2} = P(\tau) - S(\tau)\,. \tag{5.2}$$

5.1 Which transformation, taking different angles into account, changes τ such that the Feautrier transport equation 5.2 returns the solution of Eq. 5.1? Remember that the source function only depends on depth.

The quantity $P = \frac{1}{2}\left[I(\tau,\mu) + I(\tau,-\mu)\right]$ is J-like in character: numerical integration gives $J(\tau)$ from $P(\tau,\mu)$ if the latter is known for a set of m outward directions μ_j,

$$J(\tau) = \int_0^1 P(\tau,\mu)\mathrm{d}\mu \approx \sum_{j=1}^{m} a_j P(\tau,\mu_j)\,, \tag{5.3}$$

where a_j are the numerical integration weights. If we find **P** for an appropriate number of angles μ, we can numerically calculate **J**.

We write the summation in Eq. 5.3 as a closed Newton-Cotes 5-point quadrature, *Bode's rule*,

$$\int_{x_1}^{x_5} f(x)\mathrm{d}x = h\left[\frac{14}{45}f(x_1) + \frac{64}{45}f(x_2) + \frac{24}{45}f(x_3) + \frac{64}{45}f(x_4) + \frac{14}{45}f(x_5)\right] + \mathcal{O}\left(h^7 f^{(6)}\right)\,, \tag{5.4}$$

where $x_1, \ldots, x_5$ are equally spaced.

5.2 Write down Eq 5.3 using Bode's rule.

5.3 Write an IDL subroutine `QUADRATURE` with output a and μ. In order to avoid $a_i\mu_i = 0$, take $0.05 \leq \mu \leq 0.95$ with homogeneously distributed μ_i. Normalize so that $\int_0^1 \mu \mathrm{d}\mu = \sum_1^m a_i\mu_i$[1].

5.2 Λ-Matrix

An alternative for the previously defined integration scheme

$$-A_iP_{i-1} + B_iP_i - C_iP_{i+1} = S_i\,, \tag{5.5}$$

is the bidiagonal system

$$P_i = (1 + F_i)^{-1}P_{i+1} + Z_i \quad P_n = 0\,, \tag{5.6}$$

where the elimination scheme is

$$F_i = \left(K_i + \frac{A_iF_{i-1}}{1 + F_{i-1}}\right)\frac{1}{C_i}\,, \tag{5.7}$$

$$F_0 = \frac{K_0}{C_0}\,, \tag{5.8}$$

$$Z_i = \frac{S_i + A_iZ_{i-1}}{C_i(1 + F_i)}\,, \tag{5.9}$$

$$Z_0 = \frac{S_0}{B_0}\,, \tag{5.10}$$

with $K_i = -A_i + B_i - C_i$.

5.4 Express $\mathbf{Z}$ as a vector given by $\mathbf{Z} = \mathbf{X}\,\mathbf{S}$, i.e., a matrix multiplication of a certain matrix X_{ij} and the source function. Use the above elimination scheme. Determine the exact expressions for the first three rows of the matrix X_{ij}, starting with the boundary condition for Z_0.

5.5 Express the discretized transport equation in terms of a matrix $\mathbf{T}$ and the vectors $\mathbf{P}$ and $\mathbf{S}$, and give an expression for the last two rows of this matrix, using $P_n = 0$ and the fact that $\mathbf{X}$ and $\mathbf{F}$ are known.

The IDL function `LMBD_MATRIX` returns this matrix $\mathbf{T}$ as a function of angle μ and optical depth τ. Basically, it is the Feautrier program, but instead of producing the emergent intensity as a function of depth, the output is the matrix X_{ij}, as a function of μ and τ.

Consider the equation $\mathbf{P} = \mathbf{T}\,\mathbf{S}$. This equation is valid for one angle μ, and we write $P(\tau, \mu) = T(\tau, \mu)\,S(\tau)$.

5.6 Combine this equation and Eq. 5.3 in order to express $\mathbf{J}$ in terms of $\mathbf{T}$ and $\mathbf{S}$.

[1] A better solution is to choose not only variable weighting coefficients, but also varying the space between the points at which the function is to be evaluated (for extended description, see Numerical Recipes, W. H. Press).

We have determined the matrix of the $\mathbf{\Lambda}$-operator:

$$J(\tau) = \mathbf{\Lambda}\left[S(\tau)\right]. \tag{5.11}$$

5.7 Write an IDL function `LMBD`, with input `TAU`, that returns the whole $\mathbf{\Lambda}$ matrix[2]. Use your `QUADRATURE` and the function `LMBD_MATRIX`, available from `http://www.astro.uu.nl/~dwijn/teaching/afyc/`.

In the IDL program $\mathbf{J}$ and $\mathbf{S}$ are vectors and we write $\mathbf{J} = \mathbf{\Lambda}\,\mathbf{S}$.

5.8 Define τ as a floating-point array of size 101, with grid points spread homogeneously between 10^{-4} and 10^{6}. Suppose the source function equals the Planck function at each depth point. Plot $\mathbf{J}$ and $\mathbf{S}$ in one figure, assuming $B(\tau) = 1 - \mathrm{e}^{-\tau}$.

5.3 Inversion

5.9 Substitute the expression for $\mathbf{J}$ in Eq. 5.11 in the two-level coherent scattering source function $\mathbf{S} = (1-\epsilon)\mathbf{J} + \epsilon\,\mathbf{B}$ and determine the direct solution of the source function.

5.10 Assume $\epsilon = 0.1$ is constant throughout the atmosphere. Now plot, $\mathbf{B}$, $\mathbf{S}$ and $\mathbf{J}$ in one figure.

5.4 Λ-Iteration

You probably noticed that matrix inversion is computationally expensive. The classical alternative to inversion is $\mathbf{\Lambda}$-iteration,

$$\mathbf{S}_{n+1} = (1-\epsilon)\,\mathbf{\Lambda}[\mathbf{S}_n] + \epsilon\mathbf{B}\,, \tag{5.12}$$

starting with a first guess $\mathbf{S}_0$, for example $\mathbf{S}_0 = \mathbf{B}$.

The IDL procedure `LAMBDA_ITERATE`, to be found in `http://www.astro.uu.nl/~dwijn/teaching/afyc/`, returns $\mathbf{J}$, $\mathbf{S}$, the number of iterations (`STEP`), and the time needed for the iteration (`TIME`).

5.11 Study the program. When is the iteration stopped?

5.12 Vary the value of epsilon, from 10^{-3} to 0.9, and study the number of steps needed for the iteration to converge.

5.13 Plot $\mathbf{S}$, $\mathbf{B}$ and $\mathbf{J}$ in one figure, with different line styles, for different epsilon.

5.14 Change the program such that it plots $\mathbf{S}_n$ in one figure every 10 steps, for $\mathbf{B} = 1 - \mathrm{e}^{-\tau}$ and $\epsilon = 0.01$. Save this plot as a postscript file and change the program back to the old version.

[2]Your program should calculate the sum of 5 $\mathbf{\Lambda}$ matrices with an appropriate weighing factor, following from the quadrature.

5.5 Accelerated Λ-Iteration

The process of $\mathbf{\Lambda}$-iteration can be accelerated by splitting $\mathbf{\Lambda}$,

$$\mathbf{\Lambda} = \mathbf{\Lambda}^* + (\mathbf{\Lambda} - \mathbf{\Lambda}^*)\,, \tag{5.13}$$

with $\mathbf{\Lambda}^*$ the *approximate* $\mathbf{\Lambda}$*-operator*. The iteration scheme becomes

$$\mathbf{S}_{n+1} = (1-\epsilon)\,\mathbf{\Lambda}^*[\mathbf{S}_{n+1}] + (1-\epsilon)\,(\mathbf{\Lambda} - \mathbf{\Lambda}^*)[\mathbf{S}_n] + \epsilon\mathbf{B}\,. \tag{5.14}$$

The new estimate is then found from matrix inversion

$$\mathbf{S}_{n+1} = \left(1 - (1-\epsilon)\,\mathbf{\Lambda}^*\right)^{-1}\left[(1-\epsilon)\,(\mathbf{\Lambda} - \mathbf{\Lambda}^*)[\mathbf{S}_n] + \epsilon\mathbf{B}\right]\,. \tag{5.15}$$

By making a smart choice of $\mathbf{\Lambda}^*$ the inversion in Eq. 5.15 is highly simplified. In the IDL program below the diagonal of $\mathbf{\Lambda}$ is taken as $\mathbf{\Lambda}^*$, and the first guess for the source function is $\mathbf{S}_0 = \mathbf{\Lambda}^*\,\mathbf{B}$.

5.15 How is the inverse $\left(1 - (1-\epsilon)\,\mathbf{\Lambda}^*\right)^{-1}$ easily calculated? What is the main difference with the classical inversion?

Below we develop an IDL procedure `LAMBDA_AP_ITERATE`, returning $\mathbf{J}$, $\mathbf{S}$, the number of iterations and the time needed for the iteration.

5.16 Firstly, save the `LAMBDA_ITERATE` program as `LAMBDA_AP_ITERATE`.

5.17 Define the approximate $\mathbf{\Lambda}$-operator, the inverse approximate $\mathbf{\Lambda}$-operator, and the identity matrix as two-dimensional floating-point arrays of the same size as τ[3], instead of taking a first guess $\mathbf{S} = \mathbf{B}$.

5.18 Define the $\mathbf{\Lambda}$-operator. It is a function of τ.

5.19 In one loop, fill in the identity matrix, the approximate $\mathbf{\Lambda}$-operator and the inverse approximate $\mathbf{\Lambda}$-operator. Note that they are all diagonal matrices so that you have to define only elements (i, i).

5.20 Next, define the initial matrix $\mathbf{\Delta_\Lambda} = (1-\epsilon)(\mathbf{\Lambda} - \mathbf{\Lambda}^*)$.

5.21 The first guess is now $\mathbf{S} = \mathbf{\Lambda}^*\,\mathbf{B}$.

5.22 Now use the above defined operators in a new iteration scheme.

If you have changed and implemented the necessary changes, the approximate $\mathbf{\Lambda}$-operator should now be ready.

5.23 Repeat the exercises from the previous section with the accelerated $\mathbf{\Lambda}$-operator.

5.24 Plot the number of steps needed for the source function to converge using $\mathbf{\Lambda}$-iteration and accelerated $\mathbf{\Lambda}$-iteration as a function of epsilon, in one figure.

[3] For example:

```
nd=n_elements(tau)
Lambda_ap=fltarr(nd,nd)
```

5.6 Optional

5.25 Write an IDL procedure to plot the time needed for the inversion as a function of the length of τ[4].

5.26 Plot the time needed for determination of the source function as a function of the dimension of τ for the three different methods (inversion, $\mathbf{\Lambda}$-iteration and accelerated $\mathbf{\Lambda}$-iteration) in one figure.

[4]To calculate the system-time in seconds, use

```
t=systime(0)
dt=long(strmid(t,11,2))*3600+long(strmid(t,14,2))*60 $
+long(strmid(t,17,2))
```

The variable `systime(0)` is a string, for example

```
Fri Feb 9 12:00:10 1996
```

We want the time in seconds, i.e., the number of hours times 3600 plus the number of minutes times 60 plus the number of seconds. The number of hours are the 11th and the 12th characters of the string, converted into integers and so on.