Computational Physics: Homework 2

Matija Medvidović

9,5/10

October 2019

This homework was done using the Julia programming language.

Problem 1

In problem 6.10 in Newman we were asked to solve the following equation:

$$x = 1 - e^{-cx} \tag{1}$$

After coding up the overrelaxation algorithm, we found the solution $x_0 \approx 0.796812$. Solving the equation for an array of values $0 < c < 3$, we obtain the following plot:

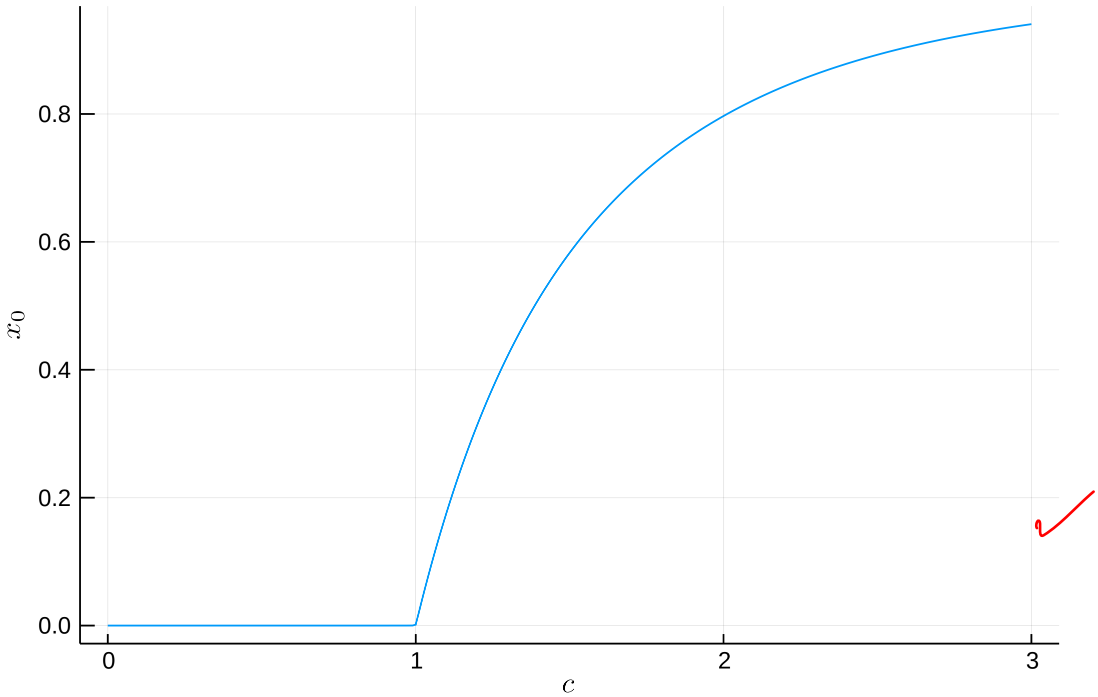


Figure 1: Dependence of the solution x_0 of Eq. 1 on parameter c. A "phase transitiion" is clearly visible at $x_0 = 1$.

Problem 6.11 (a)

The overrelaxation update rule:

$$x' = (1+\omega)f(x) - \omega x \tag{2}$$

To quantify the error ϵ, note that when x is close to the solution x^*:

$$\tilde{x} = (1+\omega)\left(f(x^*) + (x - x^*)f'(x^*)\right) - \omega x \tag{3}$$

where $f(x^*) = x^*$. Introducing $\epsilon \equiv x - x^*$ and $\tilde{\epsilon} \equiv \tilde{x} - x^*$, we obtain

$$\tilde{\epsilon} = \left[(1+\omega)f'(x^*) - \omega\right]\epsilon \tag{4}$$

which, with $x^* = x + \epsilon = \tilde{x} + \tilde{\epsilon}$, implies that

$$x + \frac{\tilde{\epsilon}}{(1+\omega)f'(x^*) - \omega} = \tilde{x} + \tilde{\epsilon} \tag{5}$$

Rearranging this equation, we get:

$$\tilde{\epsilon} = \frac{x - \tilde{x}}{1 - \frac{1}{(1+\omega)f'(x)-\omega}} \tag{6}$$

if we approximate x^* with x.

Problem 6.11 (b) & (c)

We return to Eq. 1. Using the overrelaxation algorithm, the number of iterations required to reach an accuracy of $\epsilon = 10^{-6}$ was plotted against ω defined by the overrelaxation update rule in Eq. 2. The result can be found on Fig. 2.

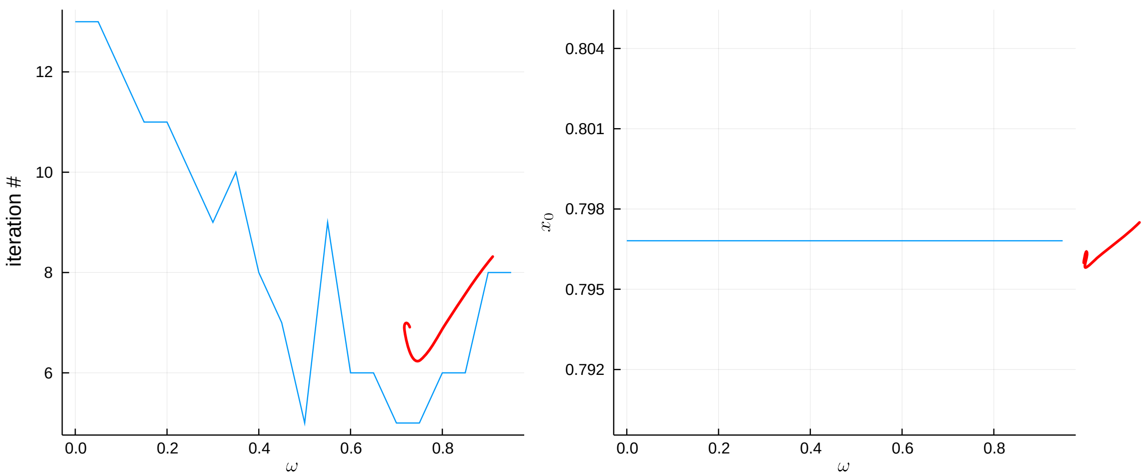


Figure 2: Overrelaxation performance as a function of ω. We can see that the optimal choice og ω is ≈ 0.7 at which point the overrelaxation algorithm needs $\approx 30\%$ as many iterations as the regular relaxation algorithm (14 @ $\omega = 0$). The right panel shows that we do reach the same solution regardless of our choice of ω.

Problem 6.11 (d)

Theoretically a negative ω value could improve performance in cases when $f(x)$ is not well-behaved. if we view the update rule (2) as a convex sum between x and $f(x)$, this would essentially mean that we take "more" of x than $f(x)$.

Problem 2

We were given Planck's formula:

$$I(\lambda) = \frac{2\pi hc^2}{\lambda^2} \frac{1}{e^{hc/\lambda k_B T} - 1} \tag{7}$$

so we get

$$\frac{\mathrm{d}I(\lambda)}{\mathrm{d}\lambda} = \frac{2\pi hc \left(e^{\frac{ch}{k_B \lambda T}} (hc - 5k_B\lambda T) + 5k_B \lambda T \right)}{k_B \lambda^7 T \left(e^{\frac{hc}{k_B \lambda T}} - 1 \right)^2} \overset{!}{=} 0\,. \tag{8}$$

Substituting $x = {}^{hc}/_{xkT}$

$$\frac{2\pi k_B^6 T^6 \left(e^x (x-5) + 5\right) x^6}{(hc)^5 \left(e^x - 1\right)^2} = 0 \quad \Leftrightarrow \quad e^x (x-5) + 5 = 0 \tag{9}$$

Running the code written for this problem, we solved this equation using binary search with accuracy $\epsilon = 10^6$. The solution reads $x \approx 4.96511$ which means

$$b = \frac{hc}{xk_B T} \approx 2.89708 \times 10^6 \text{ nm K} \quad \Rightarrow \quad T_\odot = \frac{b}{502 \text{ nm}} \approx 5771.07 \text{ K} \tag{10}$$

Problem 3

"Test" function

First we test our gradient descent algorithm on a simple function:

$$f(x,y) = (x-2)^2 + (y-2)^2 \tag{11}$$

We obtain $(x,y) = (2,2)$. The result can be seen on Fig. 3.

Actual χ^2

The Schechter function

$$n(M_{\text{gal}}) = \phi_* \left(\frac{M_{\text{gal}}}{M_*} \right)^{\alpha+1} e^{-M_{\text{gal}}/M_*} \log 10 \tag{12}$$

contains three free parameters we want to obtain as the minimum of:

nice! but what happened here?

Since steps are small, plot every 10th?

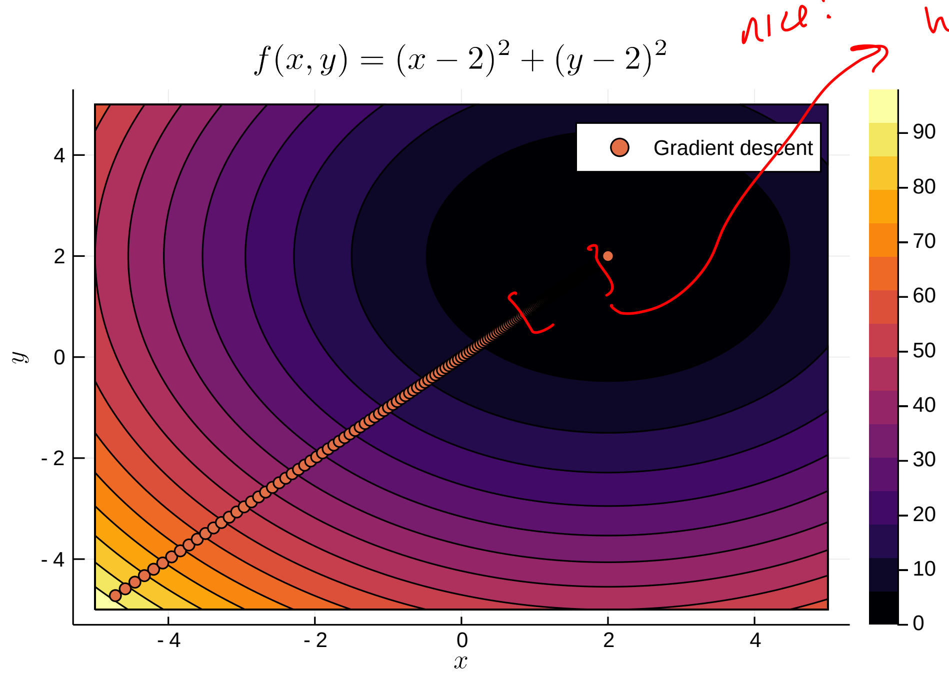


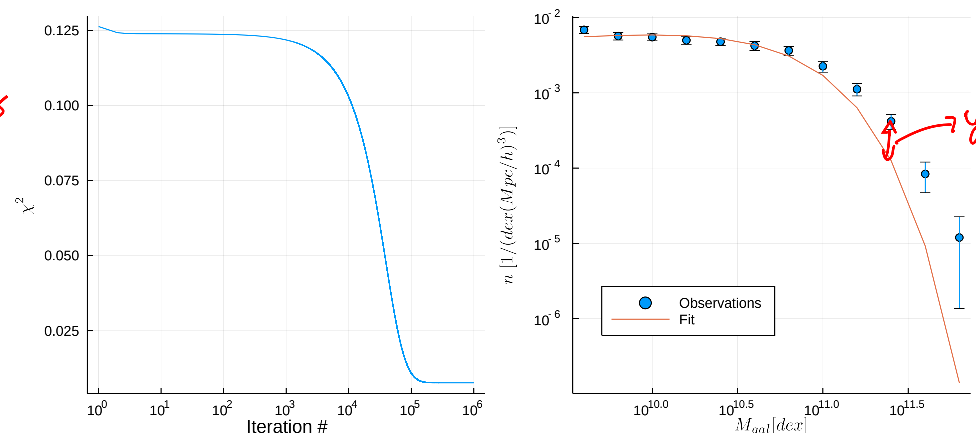
Figure 3: Gradient descent for the simple test function.

$$\chi^2(\phi_*, \alpha, M_*) = \sum_{\text{observations } i} \frac{(n(\phi_*, \alpha, M_*) - n_i)^2}{\sigma_i^2} \tag{13}$$

After running the code on χ^2, we obtained Fig. 4:

hmm... these χ^2 values are clearly not right. Too small.

$\chi^2_{min} \simeq 2$



yeah χ^2 must be larger than 1.

$\sim 1/2$

Figure 4: Gradient descent results for the χ^2 function of the Schechter model. The fit did not turn out so well, but I couldn't make it better by tweaking parameters. The model was **not** robust under $\sim$ 10 initializations of the appropriate order of magnitude but the parameters are at least of similar order of magnitude.

Representative fit parameters are:

negative? then n would be negative.

$$\phi_* = -0.0041 \, ^1/_{\text{vol}\times\text{dex}} \tag{14}$$
$$\alpha = -0.8210 \tag{15}$$
$$M_* = 5.4543 \times 10^{10} M_\odot \tag{16}$$