Computational Physics Homework 5

Hao Li[†]

December 5, 2019

Problem 1

a)

Origional ODE:

$$\frac{\mathrm{d}^2x}{\mathrm{d}t^2} = -g \tag{1}$$

Now replace the second derivative with:

$$x''(t) = \frac{x(t+h) - 2x(t) + x(t-h)}{h^2} \tag{2}$$

Put $x(t)$ on the left side and others on the right, and we will get the relaxation method equation:

$$x^*(t) = \frac{1}{2}[x(t-h) + x(t+h)] + \frac{1}{2}gh^2 \tag{3}$$

[†]hl3270@nyu.edu UID:N12137527

b)

The code for calculating the boundary value problem with equation above and boundary values $x(0) = x(10) = 0$ is available in “iteration.h”, attached in the folder.

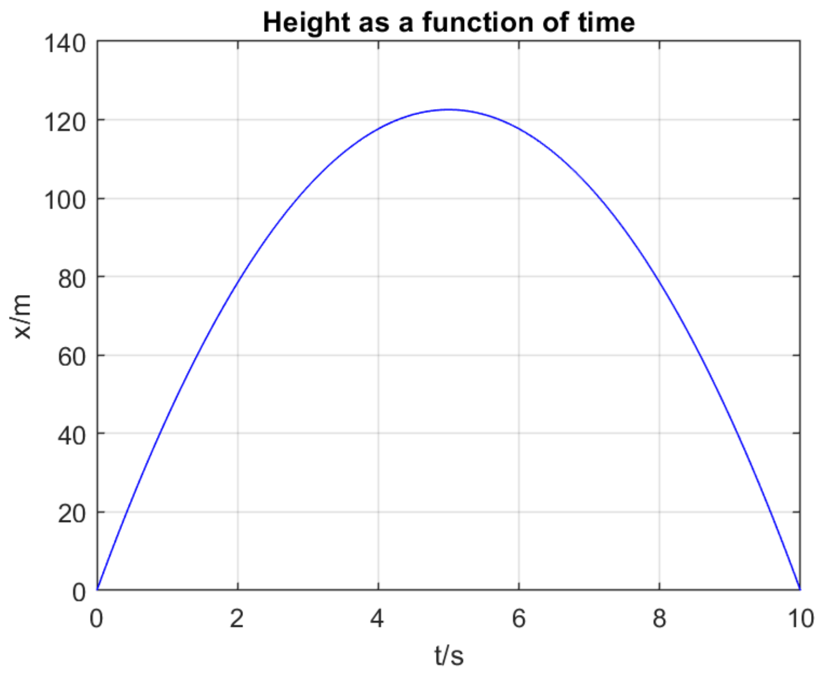


Fig. 1: *The trajectory of a ball calculated by one dimensional relaxation method. Boundary values $x(0) = x(10) = 0$, number of time grid cells $n_{\text{cell}} = 100$, target maximum difference is set to be 10^{-6}. The result agrees well with the parabolic trajectory it should be.*

By inputing the boundary values $x_{\text{left}}(t_{\text{left}})$, $x_{\text{right}}(t_{\text{right}})$,the number of grid cells n_{cell}, and the target maximum difference for all grid points between two steps max_dif_tar, the code will iterate using relaxation method in 1D with the relaxation equation 3 above, saves and shows the final values for each grid point $x(t_{\text{i}})$.

The plot for the given boundary value problem is as shown in Figure.1, which obviously agrees with the parabolic trajectory it should be. The data for this plot is saved in "relax_1D_trajectory.txt".

Problem 2

(a)

The cloud-in-cell (CIC) technique: Devide the domain of definition into grid cells centerd by grid points, and draw cells of the same shape centered by particles or points. Add the volume of particle cells (the clouds of the particles) overlapped by a grid cell to the amound for that center point, as shown in Figure. 2. After adding all the particles in this way, the amount distribution of grid cells shows the CIC approximate for particle density.

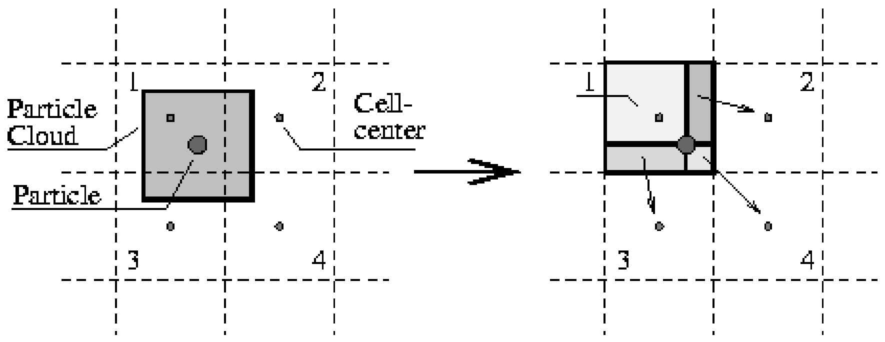


Fig. 2: *Cloud in cell technique. This picture shows an example in two dimensions. After dividing the whole area into grids, attach a cloud of the same shape and size of grid cell to each particle, centered by itself. The volume (area) of the cloud overlapped by a grid cell is added to the amount of particles of that cell centered by a grid point. After adding all the weighted volume, we will get a cloud-in-cell approximation of the density field of particles.*

The code for CIC technique is available in "particle_density.h" attached in the folder. The image for the distribution and the charge density field of the particles (x, y) given in "particles.txt" (renamed for convenience) are respectively ahown in Figure. 3 and 4. The data for the charge density field using CIC technique is saved in "particle_density_two_dim_CIC.txt".

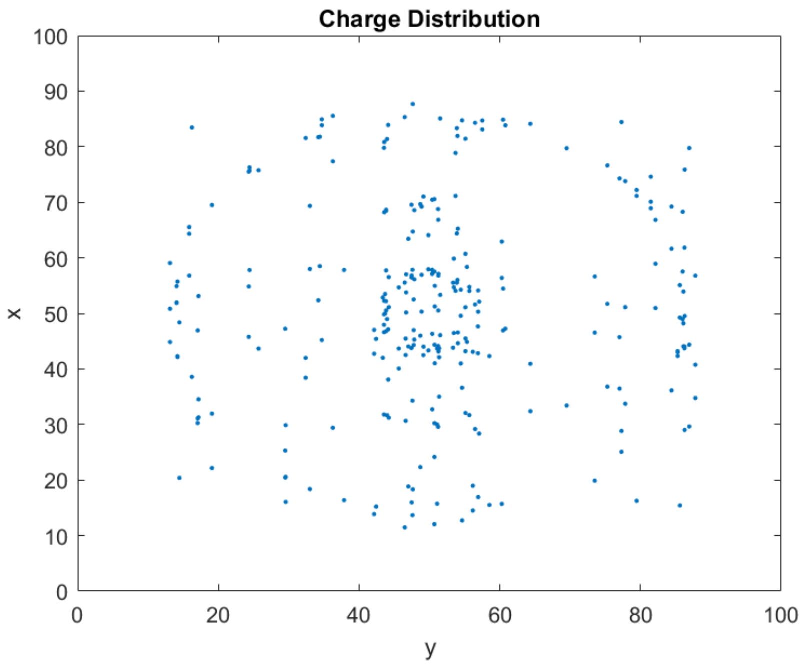


Fig. 3: *The position distribution of particles in "particles.txt".*

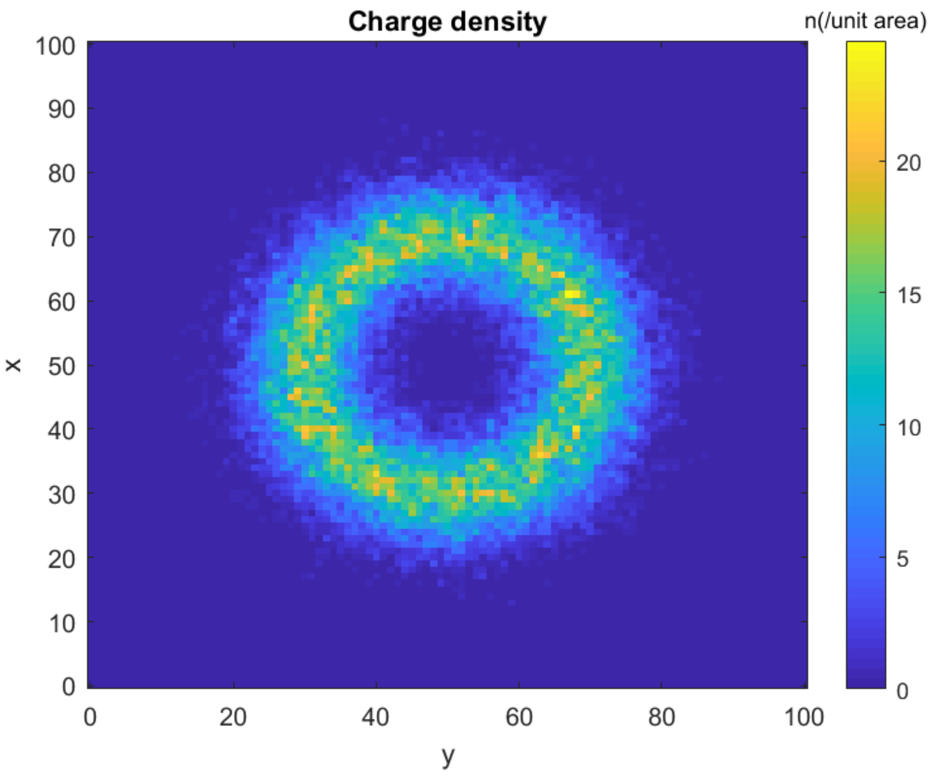


Fig. 4: *The charge density field of particles in "particles.txt". The size of a grid cell is* 1×1*, with* 100×100 *grids. Each charge has the same charge* e *as an electron.*

(b)

Simmilar to Problem 1, in two dimensions, replace the Laplace operator with

$$\begin{aligned}\nabla^2\phi(x,y) &= \frac{\partial^2\phi(x,y)}{\partial x^2} + \frac{\partial^2\phi(x,y)}{\partial y^2} \\ &= \frac{\phi(x+h_x,y) - 2\phi(x,y) + \phi(x-h_x,y)}{h_x^2} + \frac{\phi(x,y+h_y) - 2\phi(x,y) + \phi(x,y-h_y)}{h_y^2}\end{aligned} \tag{4}$$

Here we use different size for x and y since they are not necessarily the same. Now plug Eq. 4 into Poisson's equation Eq. 5 (in SI)

$$\nabla^2\phi = -\frac{\rho}{\epsilon_0} \tag{5}$$

and then we will get a relaxation equation with $\phi(x, y)$ on the left side only, shown as Eq. 6

$$\begin{aligned}\phi^*(x,y) =& \frac{h_y^2}{2(h_x^2+h_y^2)}[\phi(x+h_x,y)+\phi(x-h_x,y)] + \frac{h_x^2}{2(h_x^2+h_y^2)}[\phi(x,y+h_y)+\phi(x,y-h_y)] \\ &+ \frac{h_x^2 h_y^2}{2(h_x^2+h_y^2)}\frac{\rho(x,y)}{\epsilon_0}\end{aligned} \tag{6}$$

In this subsection, we will use the standard relaxation method similar to Problem 1 using Eq. 6, by refreshing $\phi(x_i, y_j)$ after an entire loop is finished. In this problem, all four sides are grounded so the boundary conditions are

$$\phi(x=0) = \phi(x=100) = \phi(y=0) = \phi(y=100) = 0 \tag{7}$$

The code for relaxation method to solve two dimensional boundary value problems is available in "iteration.h". By inputing the boundary of the box, the boundary conditions (if not simple, may input by function or file), number of grids for each side n_x, n_y, and the convergence criterion, i.e. the maximum difference for any cell in the grid between the current and prior step max_dif, the code will return the number of iterations, and the iteration result for each grid points, saved in a file.

In this problem, $x \in [0, 100]$, $y \in [0, 100]$, $n_x = M = 100$, $n_y = M = 100$, the boundary conditions are Eq. 7, $\text{max\_dif} = 1.0\text{E}-10$, and $\rho(x, y)$ given by (a).

The image for the relaxation method result of the Poisoon's equation is shown in Figure. 5 and 6. In Figure. 6 we can see clearly see the contour of potential goes from circular at the center to rectangular near boundary, agrees with our intuition. The data for this image is saved in “relax_2D_potential.txt”. It takes 12056 iterations to converge to the target difference.

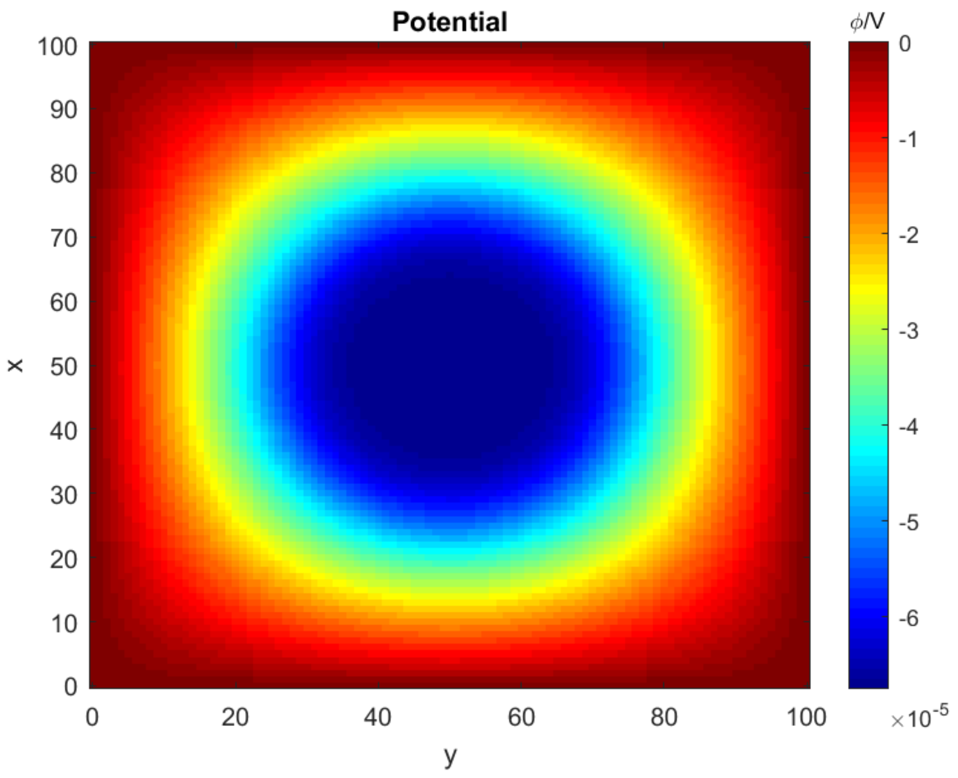


Fig. 5: *The potential $\phi(x, y)$ for the charge density field of particles in “particles.txt”, calculated using standard relaxation method.*

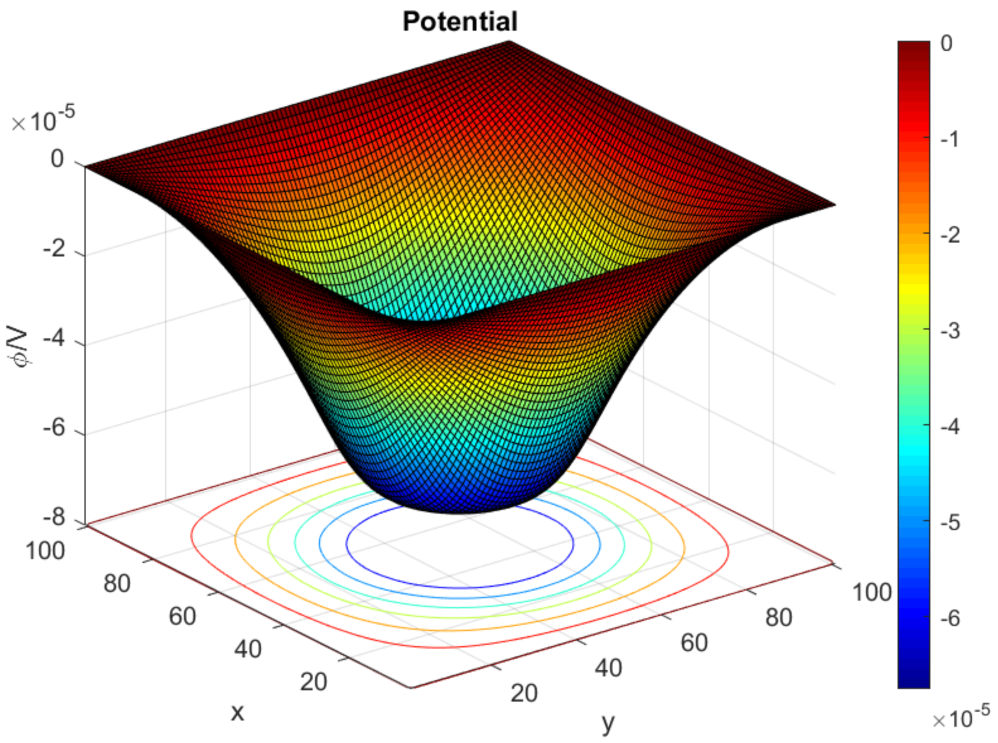


Fig. 6: *The potential $\phi(x, y)$ for the charge density field of particles in "particles.txt", calculated using standard relaxation method, shown with contours of potential.*

(c)

Similar to the standard relaxation method, Gauss-Seidel relaxation method refreshes the value at each point immediately when a new one is got. It does not ensure convergence, but sometimes this method is much faster than the standard one. The result of the same problem using Gauss-Seidel relaxation method is saved in "GS_relax_2D_potential.txt", and it takes 6697 iterations to converge to the target difference, which is nearly twice faster than the standard method.

In this subsection, we use Gauss-Seidel overrelaxation method to solve the same Poisson's equation as in (b). Label the relaxation equation Eq. 6 as $\phi'(x, y)$, so the overrelaxation equation is

$$\begin{aligned}\phi_\omega(x, y) =&(1+\omega)\phi'(x, y) - \omega\phi(x, y)\\ =&\frac{(1+\omega)h_y^2}{2(h_x^2+h_y^2)}[\phi(x+h_x, y)+\phi(x-h_x, y)] + \frac{(1+\omega)h_x^2}{2(h_x^2+h_y^2)}[\phi(x, y+h_y)+\phi(x, y-h_y)]\\ &+\frac{(1+\omega)h_x^2h_y^2}{2(h_x^2+h_y^2)}\frac{\rho(x, y)}{\epsilon_0}+\omega\phi(x, y)\end{aligned} \tag{8}$$

For standard overrelaxation method, refresh $\phi(x, y)$ after a whole iteration, and for Gauss-Seidel overrelaxation method, refresh as soon as a new one is calculated. The range of parameter $\omega \in (-1, 1)$. When $\omega = 0$, it is the just the normal relaxation method. For $\omega \in (0, 1)$, it is called overrelaxation method. For $\omega \in (-1, 0)$, it is called underrelaxation method, which may not speed up calculation. When $\omega \to \pm 1$, the calculation is instable. For some particular values of ω, the number of iterations are significantly reduced, the best of which is the optimal value of the overrelaxation parameter ω_b.

To find the optimal value of the parameter, we may use some root finding techniques, since the relationship of the number of iterations and the value of parameter is relatively smooth. Here golden ratio search is taken as an example. Firstly, for $\omega > 0.985$, the calculation is too slow to tell if it could actually converge, so we choose the maximum of $\omega$for searching to be 0.984, and the minimum to be 0. Considering the slope of $n_{\text{iterations}}(\omega)$, to get a clearer plot of it, the range is separated into two parts, [0, 0.981], and [0.98, 0.984]. Golden ratio search is used in the first part to get the optimal value parameter, and in the second for more details of the large slope part. Search stops when reaching the target precision, here is set to be 0.001.

The process of the golden ratio search is :

(1) Start with two boundaries ω_1 and ω_4. Calculate the two golden ratio points in the range $[\omega_1, \omega_4]$:

$$\begin{aligned} \omega_2 &= \omega_1 + \frac{1+\sqrt{5}}{2}(\omega_4 - \omega_1) = \frac{3-\sqrt{5}}{2}\omega_1 + \frac{1+\sqrt{5}}{2}\omega_4 \\ \omega_3 &= \omega_4 - \frac{1+\sqrt{5}}{2}(\omega_4 - \omega_1) = \frac{1+\sqrt{5}}{2}\omega_1 + \frac{3-\sqrt{5}}{2}\omega_4 \end{aligned} \tag{9}$$

(2) Calculate the value of n for each ω, and find the minimum of these four n's.

(3) If the minimum is at ω_1 or ω_2, then narrow the range of ω to $[\omega_1, \omega_3]$. So the new parameters

$$\begin{aligned} \omega_4^* &= \omega_3 \\ \omega_3^* &= \omega_2 \\ \omega_2^* &= \frac{3-\sqrt{5}}{2}\omega_1 + \frac{1+\sqrt{5}}{2}\omega_4 \end{aligned} \tag{10}$$

If the minimum is at ω_3 or ω_4, then narrow the range of ω to $[\omega_2, \omega_4]$. So the new parameters

$$\begin{aligned} \omega_1^* &= \omega_2 \\ \omega_2^* &= \omega_3 \\ \omega_3^* &= \frac{1+\sqrt{5}}{2}\omega_1 + \frac{3-\sqrt{5}}{2}\omega_4 \end{aligned} \tag{11}$$

(4) Calculate the value of n for the new ω_2^* or ω_3^* added.

(5) Repete step (3) and (4) until reaching the target precision $\omega_4 - \omega_1 < \Delta\omega_{\text{tar}}$

The data for these two calculations are saved in "GS_overrelax_2D_potential.txt"and "GS_overrelax_2D_potential(2).txt". Figure. 7 shows the relationship of the number of iterations with the value of overrelaxation parameter ω. In this figure, we can see that the number of iterations drops gradually but significantly until reaching the optimal value $\omega_b \approx 0.946$, and then explodes to infinity when reaching the instability point $\omega = 1$. The optimal value of ω lies within [0.945,0.946]

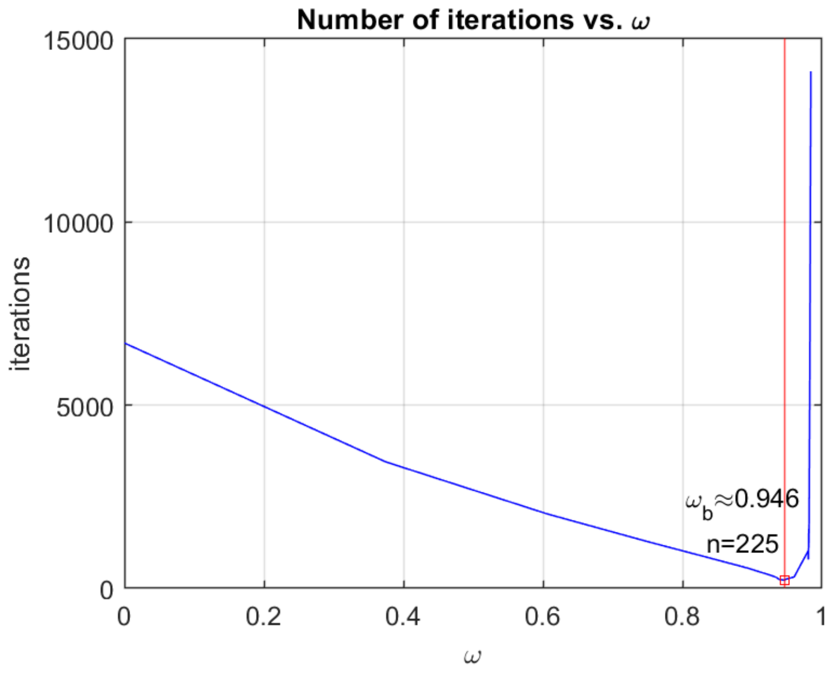
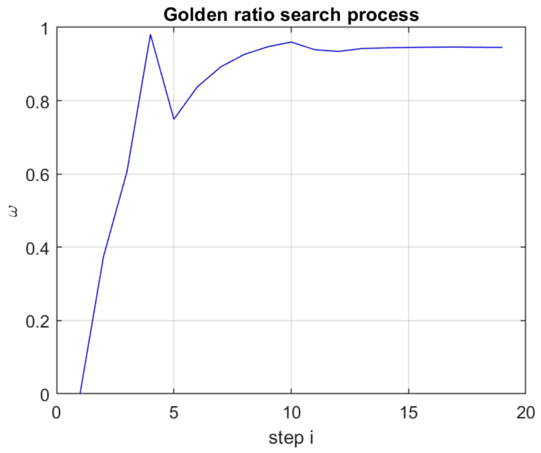
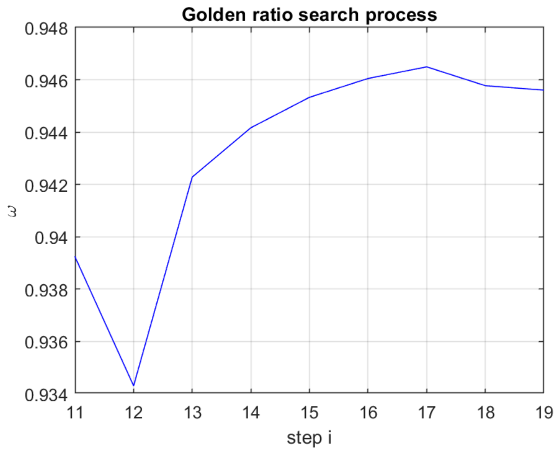


Fig. 7: *The relationship of the number of iterations with the value of overrelaxation parameter ω, for the solution for Poisson's equation. As ω increases from 0 until reaching around $\omega_b \approx 0.946$, the number of iterations decreases from arround 7000 to around 225. As it goes on increasing to 1, the number of iterations explodes to infinity.*

The evolution of ω with each step in the minimization process is shown in Figure. 8, while 8a shows the whole process, and 8b shows the details arround the final result.



(a) *The whole minimization process.*



(b) *The detailed process when near the final answer.*

Fig. 8: *The evolution of the overrelaxation parameter* ω *with each step* i *in the minimization process. After arround ten steps, it reaches the vincinity of the optimal value of* $\omega_b \approx 0.946$