THEORY, MODELLING AND SIMULATION

Computational material science

SUMMER TERM 2019

***HOMEWORK*:**

DISLOCATIONS INTERACTIONS AND DYNAMICS

Prof. Stefan Sanfeld

Submitted by:

DHAVAL RASHESHKUMAR PATEL

CMS-63940

Task 1: Internal stress calculation

In this task I wrote one function which takes input as initial parameters and takes the initial positions of different dislocations. The main purpose of writing this function is finding the internal stress of each dislocations. This internal stress of each dislocations is due to the presence of other dislocations. So, in this I have to sum up the stresses.

So that's why I have to use for loop here. In this for loop I used if loop because It is not possible to find the stress of any dislocations due to presence of itself. So, I put condition that if the dislocations are not same than only there is any stress field due to interaction.

$$\sigma_{xy} = Dx\frac{(x^2-y^2)}{(x^2+y^2)} \text{ , } D = \frac{Gb}{2\pi(1-\vartheta)} \text{ , } velocity(v) = \frac{Burger\ vector(b)}{drag\ coefficient} * shear\ stress\ \sigma(xy)$$

Here, G = shear modulus, ϑ = Poisson's ratio, b = Burgers vector, x, y = dislocation coordinates

Here, we are finding the stress value of 1D problem, so I took only X-directions positions and took Y=0. So, after that I am finding the distance between each dislocation and then use it into the stress formula. Here I use array to list the different stresses values of each dislocation. The return statement of this functions gives the internal stress values in terms of array.

```python
def get_internal_stress(dx,shear_modulus,poissons_ratio,burgers_vector):
    sigma_xy_list = []
    for j in range(0,len(dx)):
        sigma_xy = 0
        for i in range(0,(len(dx))):
            if i != j:
                dis = dx[j]-dx[i]
                D = shear_modulus*burgers_vector/(2*np.pi*(1-poissons_ratio))
                sigma_xy = sigma_xy + D*(1/(dis))
        sigma_xy_list.append(sigma_xy)
    return sigma_xy_list
```

Task 2: Implement a full Simulation as outlined above

In the first task we define a function to find the internal stresses of the dislocations which are statics. But now we know that due internal stress or stress field around the dislocations, they are now interacting or repelling each other and because of this they are moving, and they are changing their positions at every small-time difference. So now we can make this task dynamics with the help of above defined function and another for loop.

```python
for i in range(n_step):
    stress = get_internal_stress(np.copy(x_new[i]),shear_modulus,poissons_ratio,burgers_vector)
    stresses.append(stress)
    velocity = [x * bu_dr for x in stress]
    velocities.append(velocity)
    dt = np.array([x * delta_t for x in velocity])
    x_n = x_new[i] + dt
    x_new.append(x_n)
    delta = times[i] + delta_t
    times.append(delta)
```

In the for loop, first I am calling my defined function to get the stress value of each dislocation at its initial positions. After that I found the velocity of this dislocations due to this internal stress and I append it to the n-dimensional array. Due to this velocity every dislocation changes their position, so I found new position of every dislocation which is shown by x_n in this code and append it to n-dimensional array. After that I am increasing my time value and append it to one dimensional array. In this I choose the delta time as 0.1 ns and my total time is 10 ns. So, my number of steps is 100. I got the same answer as given in the homework pdf but I also getting the one more position array in additions.

Task 3: Post processing

In this task I defined one function to plot the updated positions of every dislocations with respect to time interval. I took two input in this function one is positions and second is time.

```
def plot_trajectories(times,positions):
    fig, ax = plt.subplots()
    positions = np.divide(positions,1e-6)
    times = np.divide(times,1e-9)
    ax.plot(positions,times,'-o',markevery = [0])
    ax.plot(positions,times,'D',markevery = [-1])
    plt.title('trajectory of dislocations (t0: circles, tend: diamonds)')
    plt.xlabel('positions of dislocations [μm]')
    plt.ylabel('time [ns]')
    plt.legend(('dislocation_1','dislocation_2','dislocation_3','dislocation_4'))
    fig.savefig('Trajectories_of_dislocations.png')
    plt.close(fig)
```

In **Figure 1**, I plot the trajectory of two different dislocations, whose initial positions are 0.2 μm and 0.25 μm. In this figure we can see that if there is no external stress and no boundary condition like grain boundary, they are repelling each other and going far away from each other, in addition it is also clear that they are repelling each other with same amount of force.

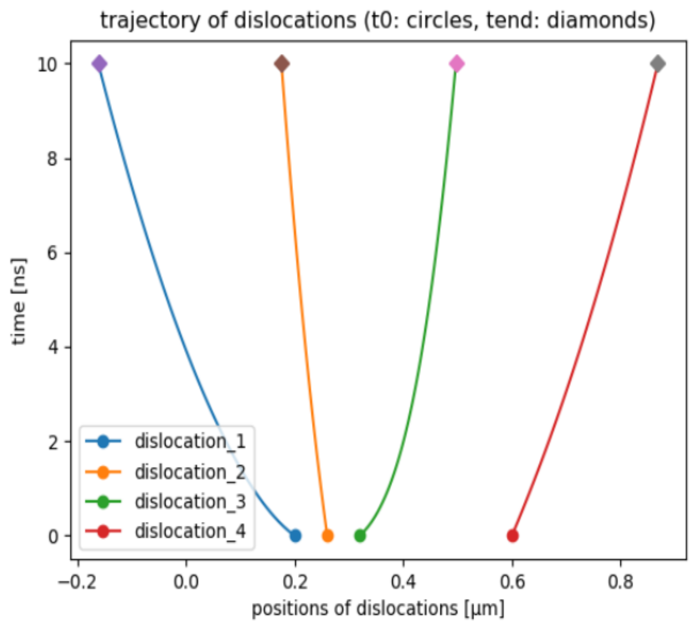

Figure 1 Trajectory of four dislocations

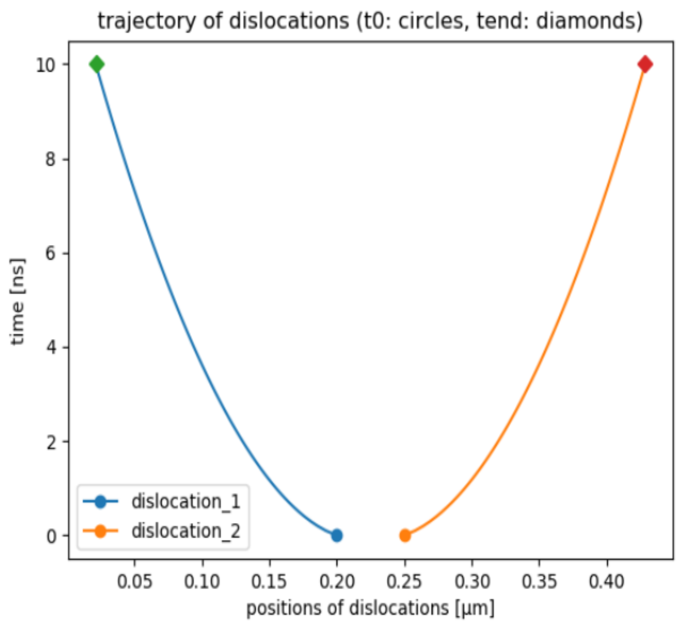

Figure 2 Trajectory of two dislocations

In **Figure 2,** I plot the trajectory of four different dislocations, whose initial positions are given as (2.0e-07,2.6e-07,3.2e-07,6.0e-07) m. In the task 3 still I didn't introduce any external stress and boundary conditions. So, we can see here that they are repelling each other and goes far from each other. The pattern of trajectory of dislocations are depend on their initial positions, if they are positioned at equidistance from each other than they are travelling exact amount of distance in opposite directions as shown in figure 1.

Task 4: External loading

In this task I introduce external shear stress, which means that system is externally loaded. This external load is directly added to the resulting internal stress value. This total stress value is affecting the dislocations positions and velocities. So, I used this total stress value to find the velocity of every dislocation under the external shear stress. In this task I have to change the external shear stress value such that the final position of leftmost(first) dislocations is remains same as its initial position (0.2 µm). I get the external shear stress value as 14.118725mpa such that it satisfies above condition.

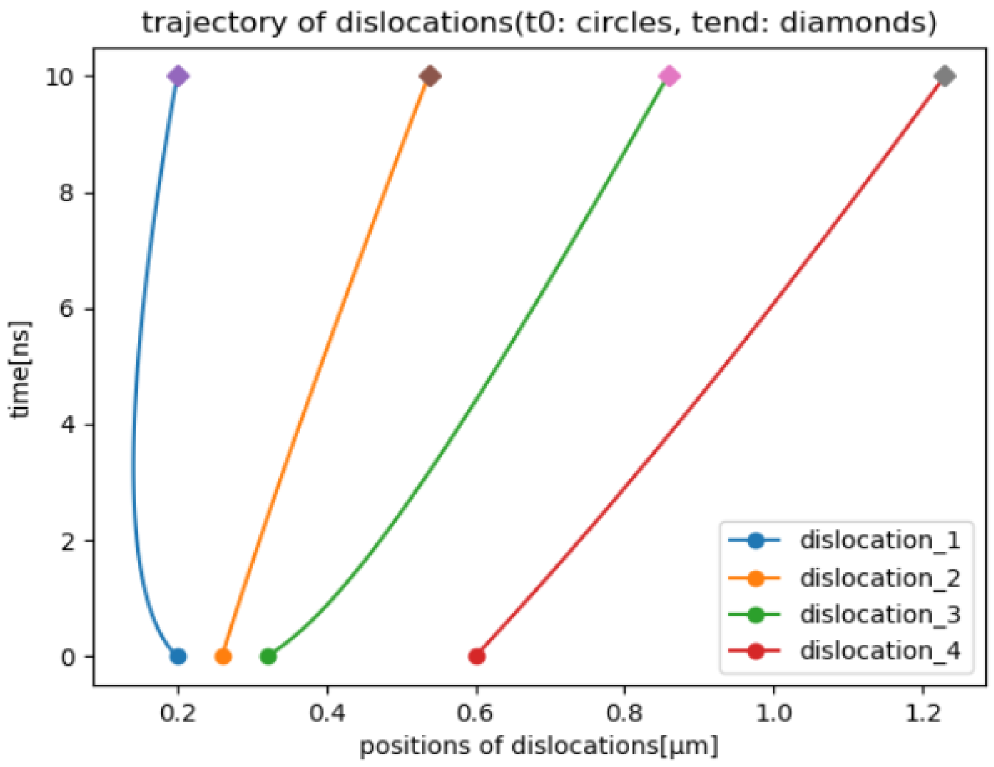

Figure 3 Trajectory of dislocations under external loading

Task 5: Incorporate an impenetrable boundary condition at X=0

In this task, I just implement the boundary condition in addition to the external loading, which I implemented in last task. I implemented grain boundary condition such that when the leftmost(first) dislocation reaches to the x<=0 I stopped this dislocation to x=0. In this task I just added some statements in for loop which I described in task 2.

I just checked the current position of dislocation in for loop and compare the first element of positions array (position of first dislocation) to the 0 in the if loop. I also set the value of first dislocation position to 0 if loop execute.

```
current_position = x_n
if current_position[0] <= 0:
    current_position[0] = 0
    x_new.append(current_position)
else:
    x_new.append(current_position)
```

Task 5a:

In this task, I just added -25mpa external shear load and plot the dislocation trajectory for two different time intervals with 10ns and 30ns and compare them. In this figure 4 and figure 5 we can easily observe that when we increased time after 20ns the dislocations are not moving.

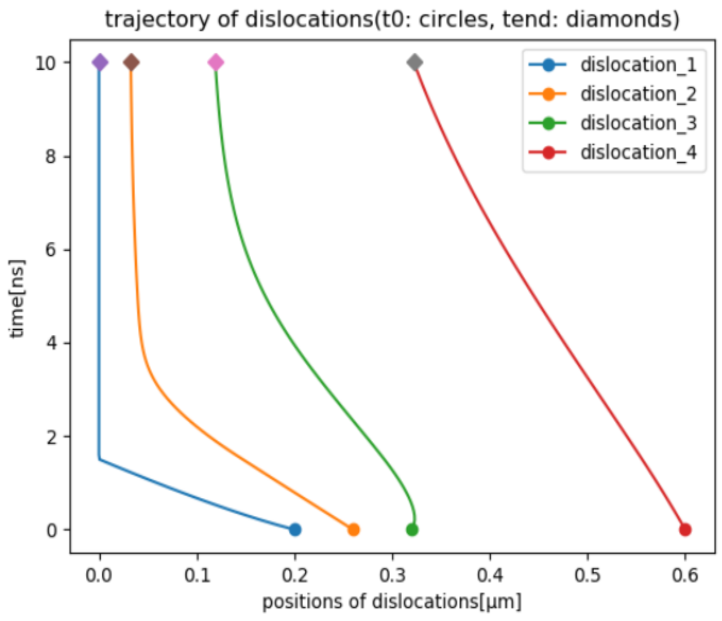

Figure 4 Trajectory of dislocations under external loading until 10ns

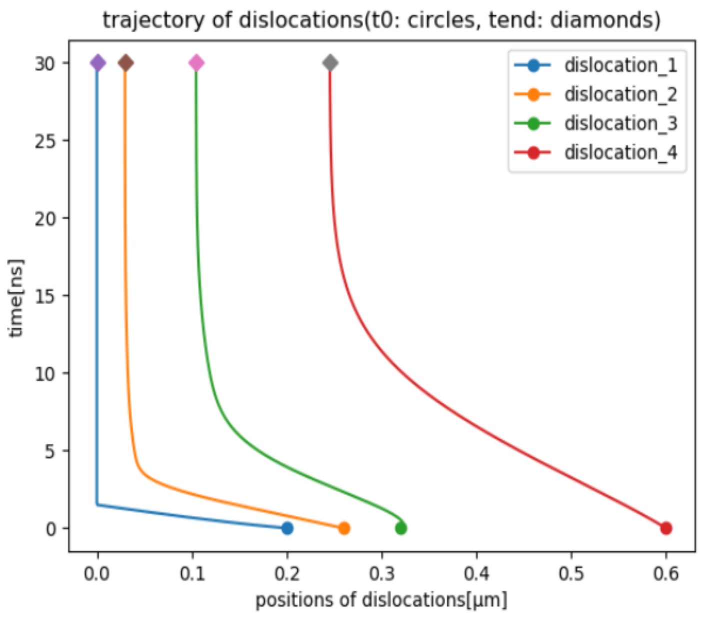

Figure 5 Trajectory of dislocations under external loading until 30ns

Task 5b:

In this task, I define the function to plot the velocities of dislocations with respect to time. Here, I attached the two figures to show the velocities of dislocations until 10ns and until 20ns. In these figures 6 and 7 one can easily examine that velocity of dislocations 2,3,4 are becoming to zero at about 20ns.

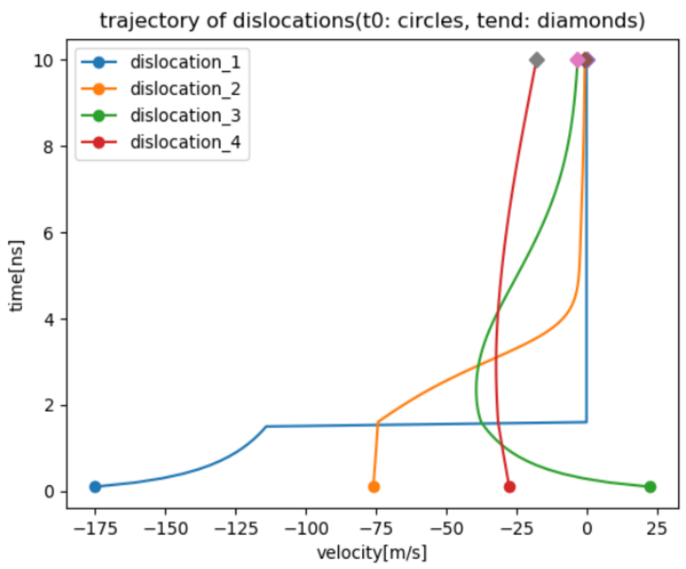

Figure 6 Trajectory of dislocations velocity until 10ns time

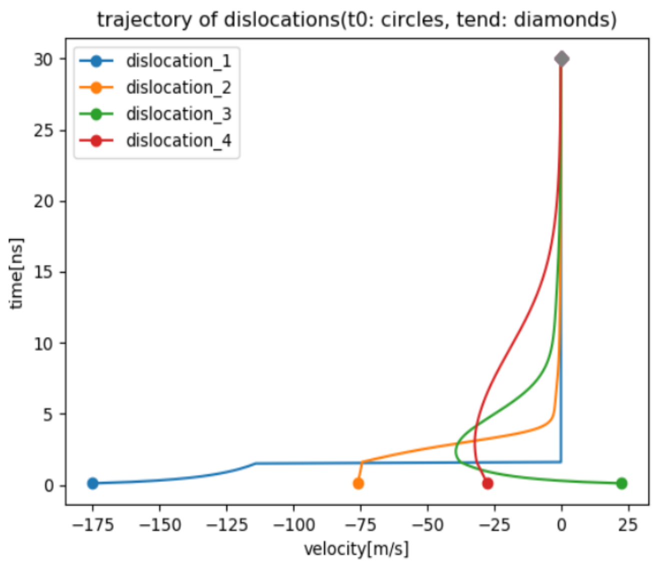

Figure 7 Trajectory of dislocations velocity until 20ns time

Task 5c:

In this task, I have compared two graphs first one is dislocations positions with respect to time and second one is dislocations velocity with respect to time until about 30 ns.

In these figures 4,5,6 and 7 we can observe that after some time when first dislocation reaches to x=0 it is fixed to 0, but other dislocations moving until 20ns and after that they are not moving, and their velocities become 0.

If there is no external shear stress i.e. external shear stress is zero instead of -25mpa than the first dislocation is after sometimes reaches to grain boundary and stuck to it but after that another 3 dislocations are moving opposite to first dislocations and go far with increasing time and they are never stable.

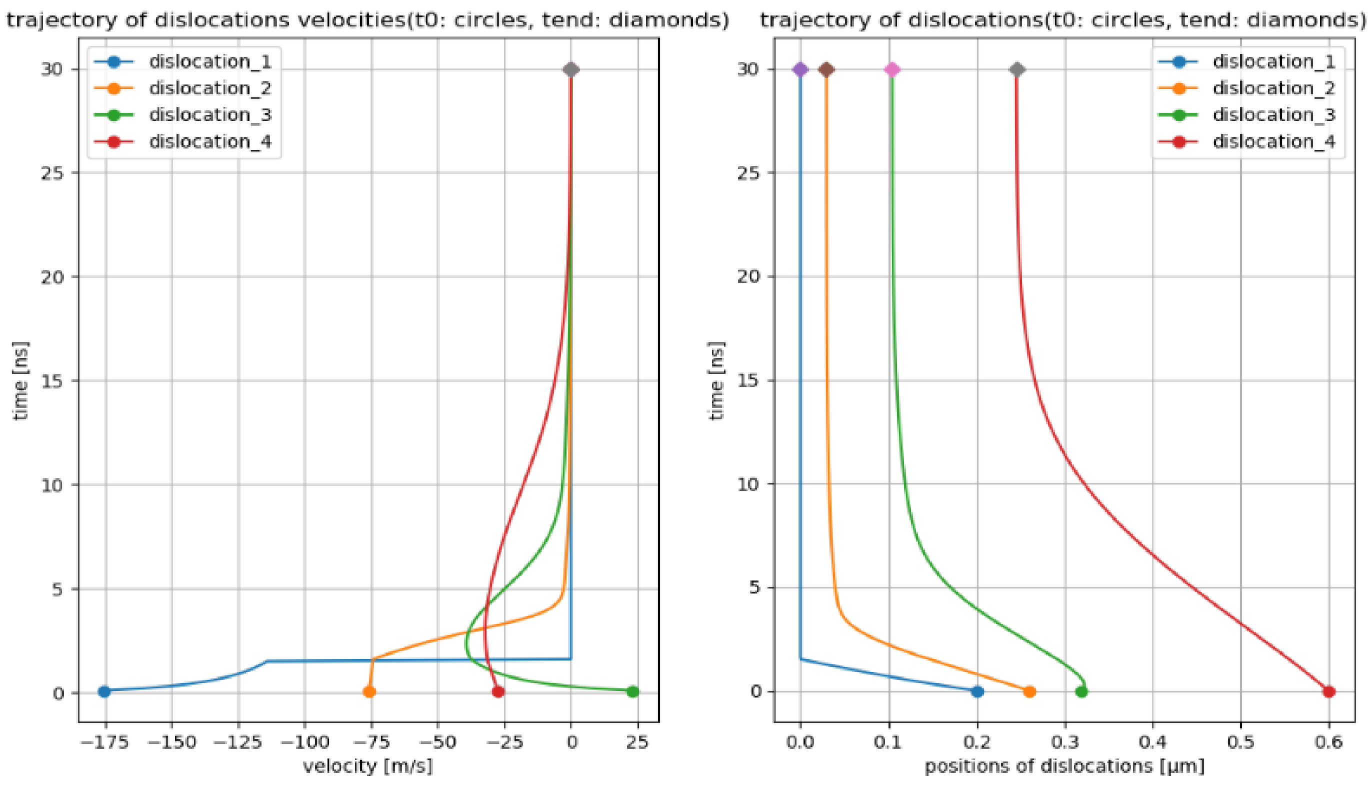

Figure 8 Comparison of dislocations velocities and positions.