Final report

Alex Olar

University of Eotvos Lorand

olaralex@caesar.elte.hu

April 1, 2019

Schedule

1. Download and build the source on your computer. Study the available documentation and the help menu. Play around with it ;) Create a configuration with two dislocations in non equilibrium state (with same and different Burgers-vectors) and relax the system. Analyse the results and the log file! DONE
2. Create a small program which is able to extract how the field of a dislocation looks like which is in the origo of the simulation cell. Plot it! DONE
3. Compare the available periodic fields. What happens if you change the following line in "include/constants.h" so fewer periodic images is taken into consideration during the analytic field calculation? DONE
4. Create a random dislocation system with 64 dislocations. Relax the system into equilibrium with the above N for (1,2,3,4) with the analytic periodic field. Compare the log files and the results. Can you see any difference? DONE

Schedule

1. From now on, use elte's periodic shear stress! What is the difference between a single dislocation's shear stress field and where we use periodic boundaries? DONE
2. Create several different random dislocation configurations with the same dislocation count. Relax them into equilibrium! Average the average dislocation speed - simulation time data. What can be observed? DONE
3. Experiment with the external stress protocols on your relaxed systems. What can you observe? What does it mean? DONE
4. Observe with different configurations and system sizes how the dislocation's relative distance change with the external stress. Make the appropiate modifications in the source if this is needed! DONE

Installation

Installating the software is straightforward on Debian, only some additional packages are required for the process which can be found easilly. These are the following:

- g++ or clang
- cmake
- make
- boost
- umfpack from suitesparse
- gsl

Given these packages the build is simple as well just typing this in terminal window after cloning the repository.
Building is done with *cmake*.

The C++ code

```
    fclose(fd);
}

void PeriodicShearStressELTE::outPutStress(){

    float resolution = 0.005;

    for(float i = -0.5; i < 0.5; i += resolution){
        for(float j = -0.5; j < 0.5; j += resolution){
            fout << xy(i, j) << ";";
        }
        fout << "\n";
    }

}

double PeriodicShearStressELTE::xy(double x, double y)
```

Visualizations

The ELTE field looks the following way:

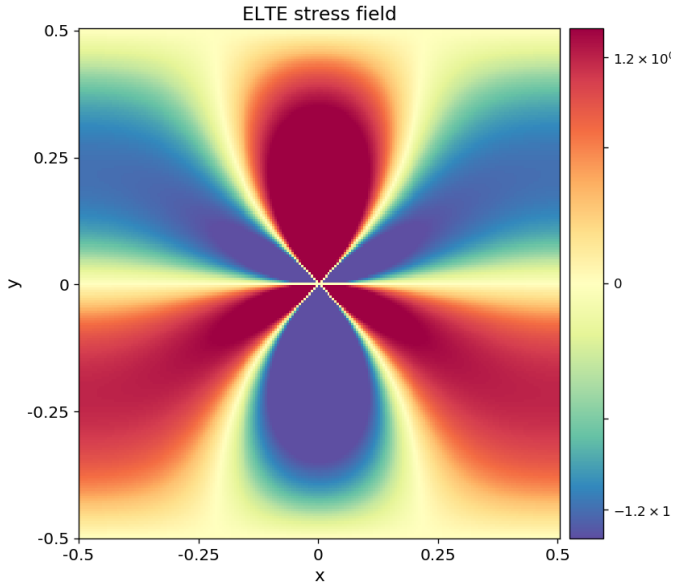


Figure: Stress field visualization

While changing N from 1 to 12 in the calculataion of analytic fields results are the following in order:

Difference of analytic fields

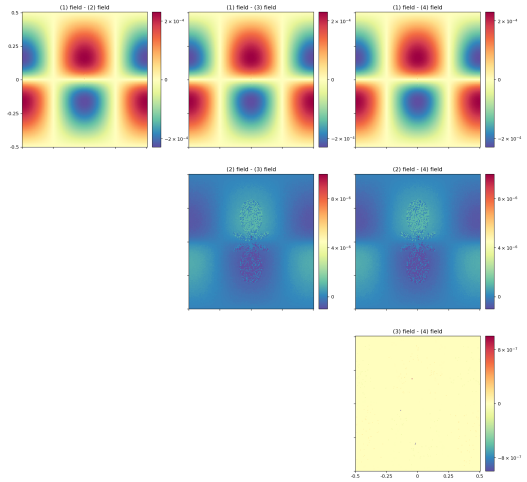
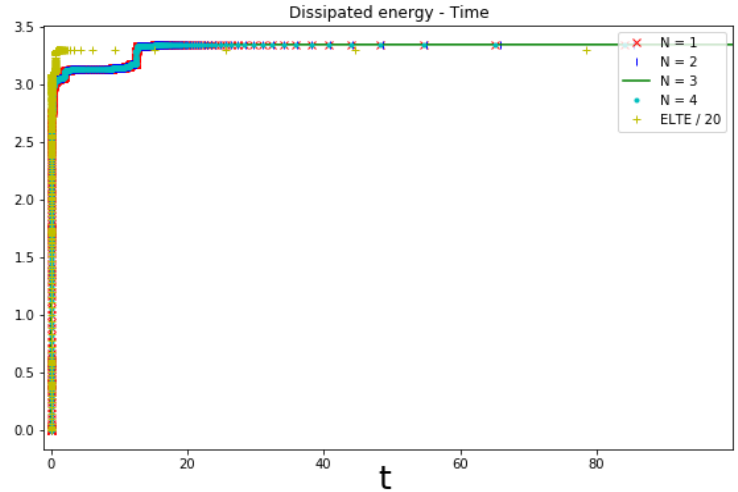
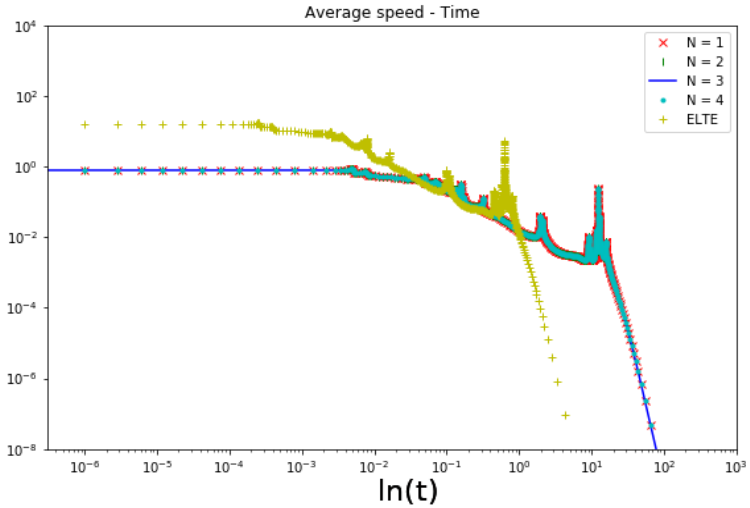


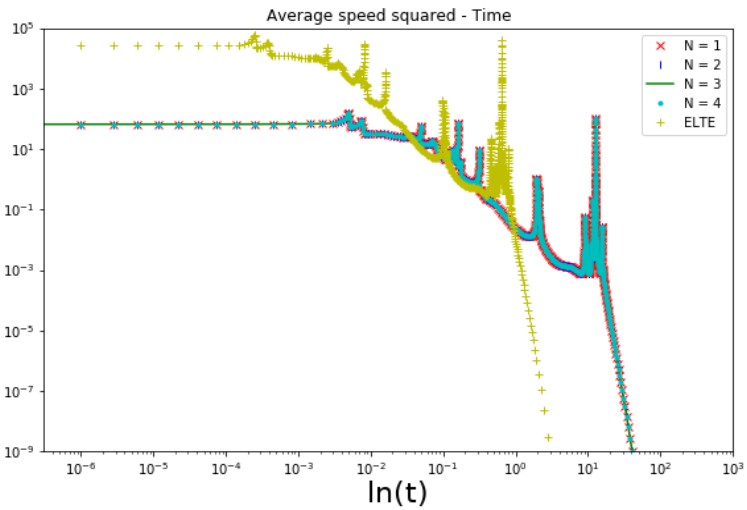
Figure: Difference of analytic fields

Where row - column ordering is created. It can be seen that the very last plot, that shows the difference between the 3rd and 4th analytic fields, where N=4 or N=3. It can be clearly seen that disregarding some points the analytic fields become equal from N=3 and therefore the difference is almost everywhere zero.

Analyzing the log files







Conclusion

- The ELTE field dissipates energy faster, the analytic fields do not differ that much.
- It is worth considering why we see peaks on the speed diagrams. It is due to dislocations getting close to each other and repenting each other with an immense force resulting in high velocity peaks.
- Otherwise the same structure can be seen on both systems.

Dislocations

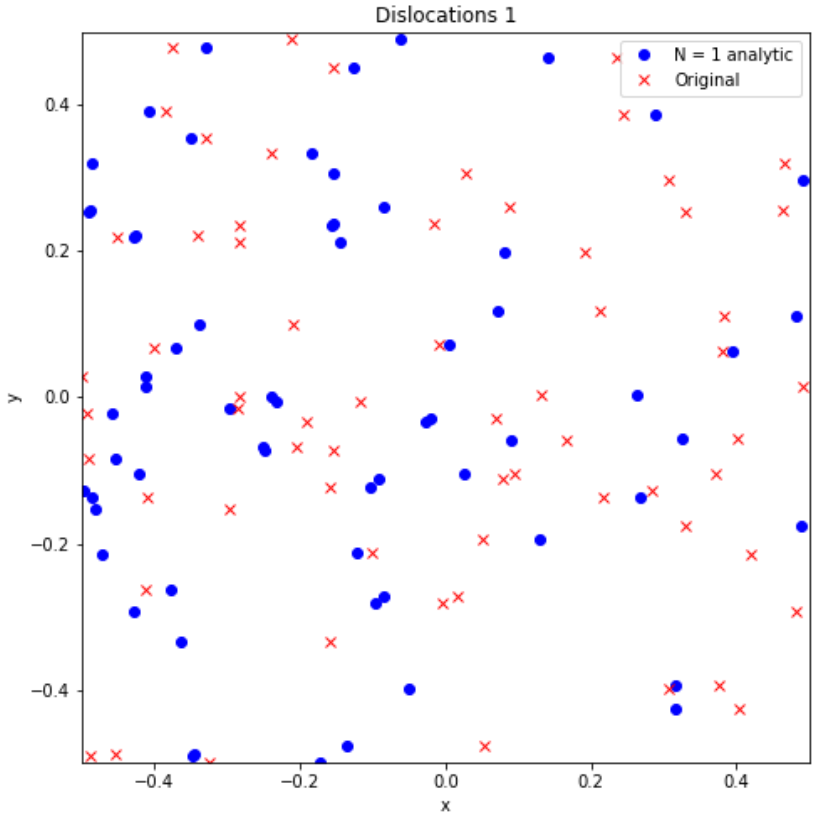


Figure: N = 1 analytic fields

What does N mean?

Previously I mentioned that the analytic field is calculated via taking into account N images.

- in the y-axis the periodic boundary condition can be analytically calculated by summing up infinitely many fields
- in the x-axis N images must be taken into account when the summation of fields is executed since there is no analytic formula
- it can be imagened as field stacking in the x-axis, when a dislocation crosses the simulation boundary on the right hand side it enters the same field stacked after the previous one on the left hand side, therefore the same field acts upon it

Stress field of an edge dislocation in (0, 0)

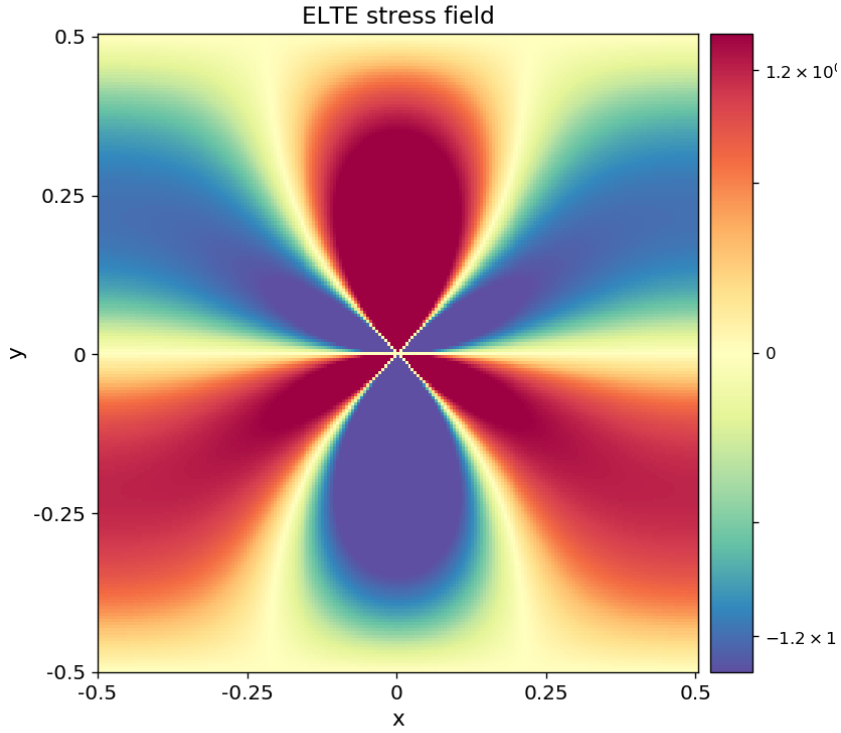
$$\sigma_{xx} = \frac{-Gb}{2\pi(1-\nu)} \frac{y(3x^2+y^2)}{(x^2+y^2)^2}$$

$$\sigma_{yy} = \frac{Gb}{2\pi(1-\nu)} \frac{y(x^2-y^2)}{(x^2+y^2)^2}$$

$$\tau_{xy} = \frac{Gb}{2\pi(1-\nu)} \frac{x(x^2-y^2)}{(x^2+y^2)^2}$$

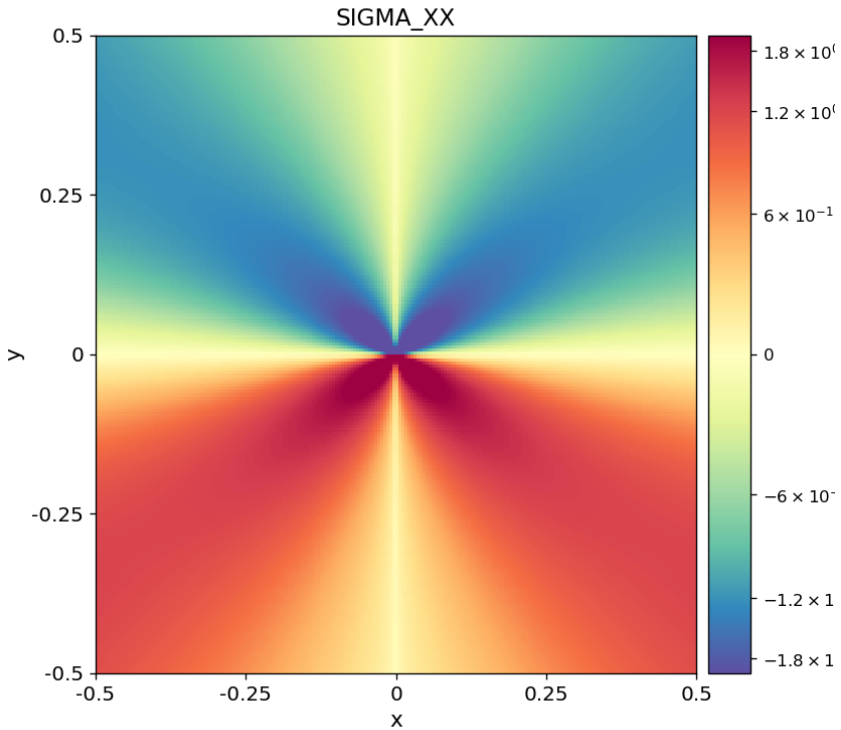
The off diagonal part is used in the simulation.

Stress field of an edge dislocation in (0, 0)



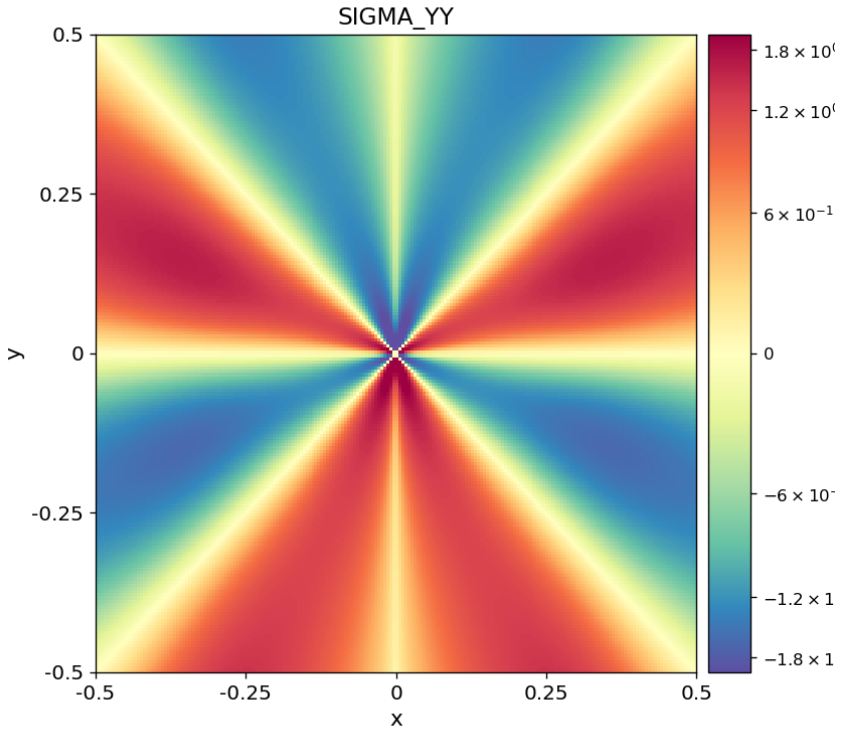
This is the previously seen ELTE stress field which has a periodic boundary condition.

Stress field of an edge dislocation in (0, 0)



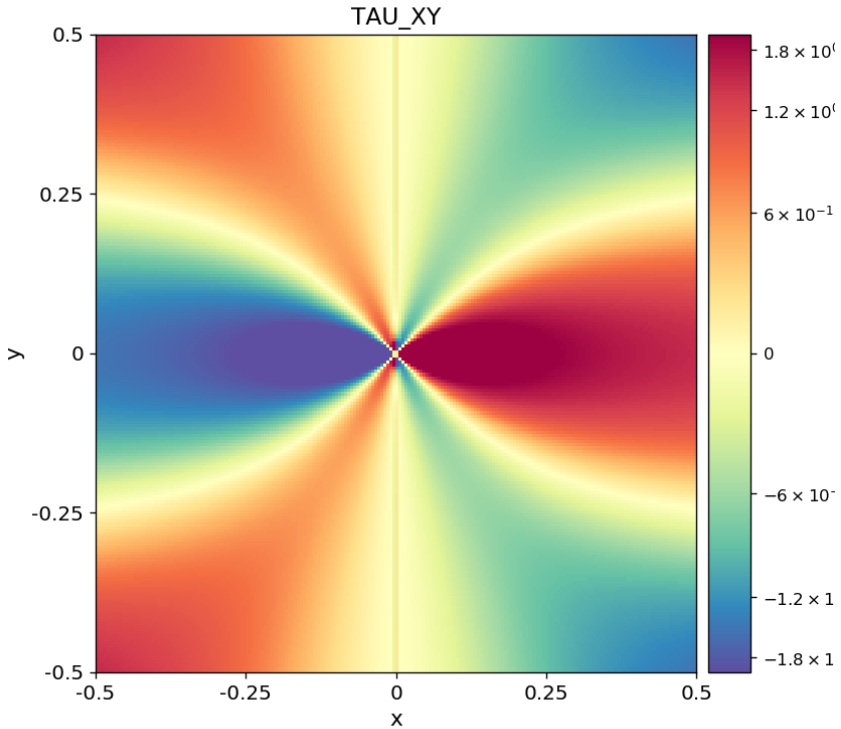
The σ_{xx} component.

Stress field of an edge dislocation in (0, 0)



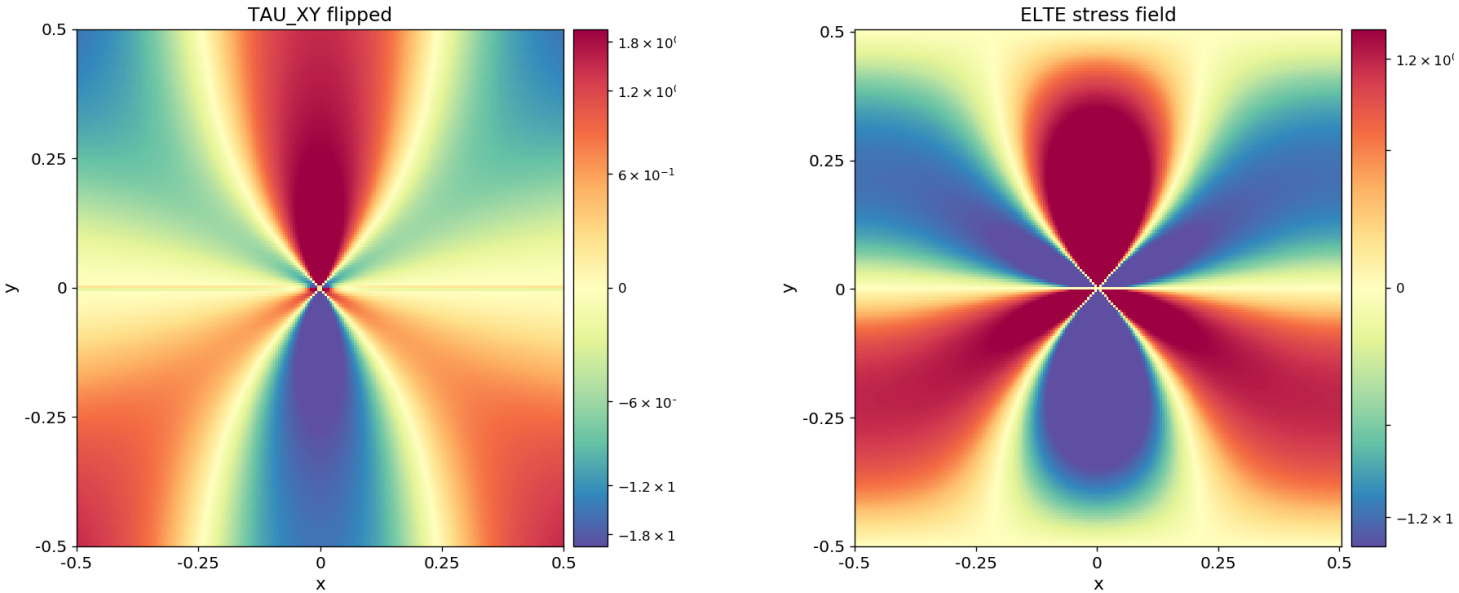
The σ_{yy} component.

Stress field of an edge dislocation in (0, 0)



The τ_{xy} component. This resembles to the ELTE field.

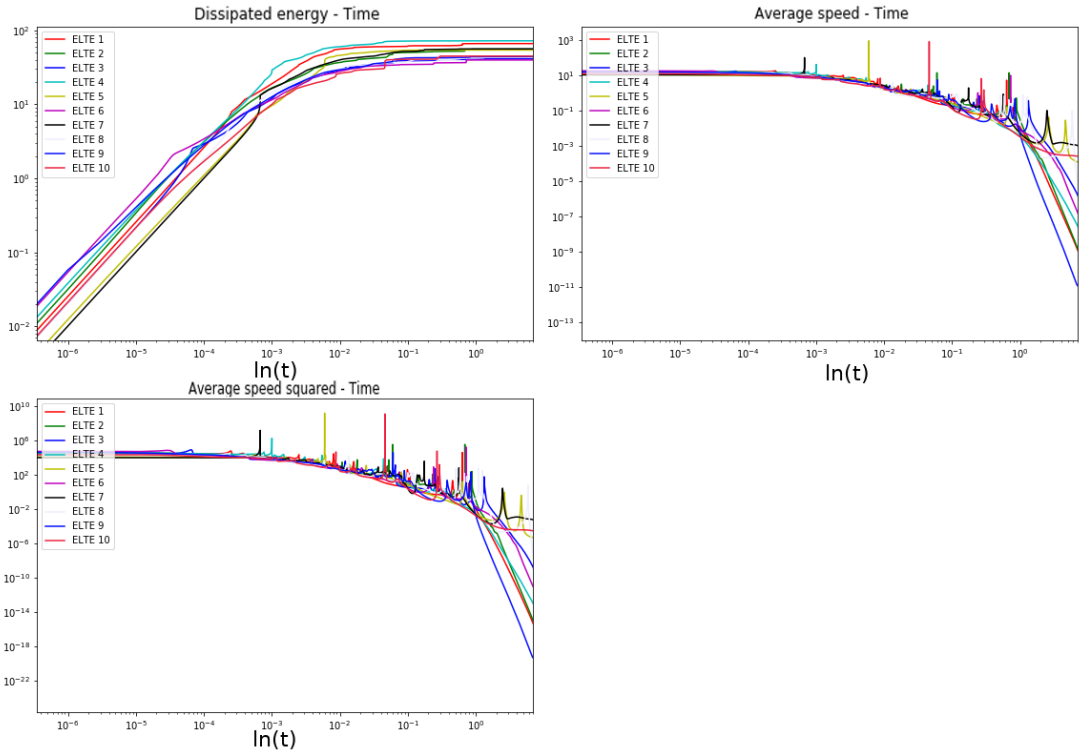
Stress field of an edge dislocation in (0, 0)



The τ_{xy}^{T} component and the ELTE field.

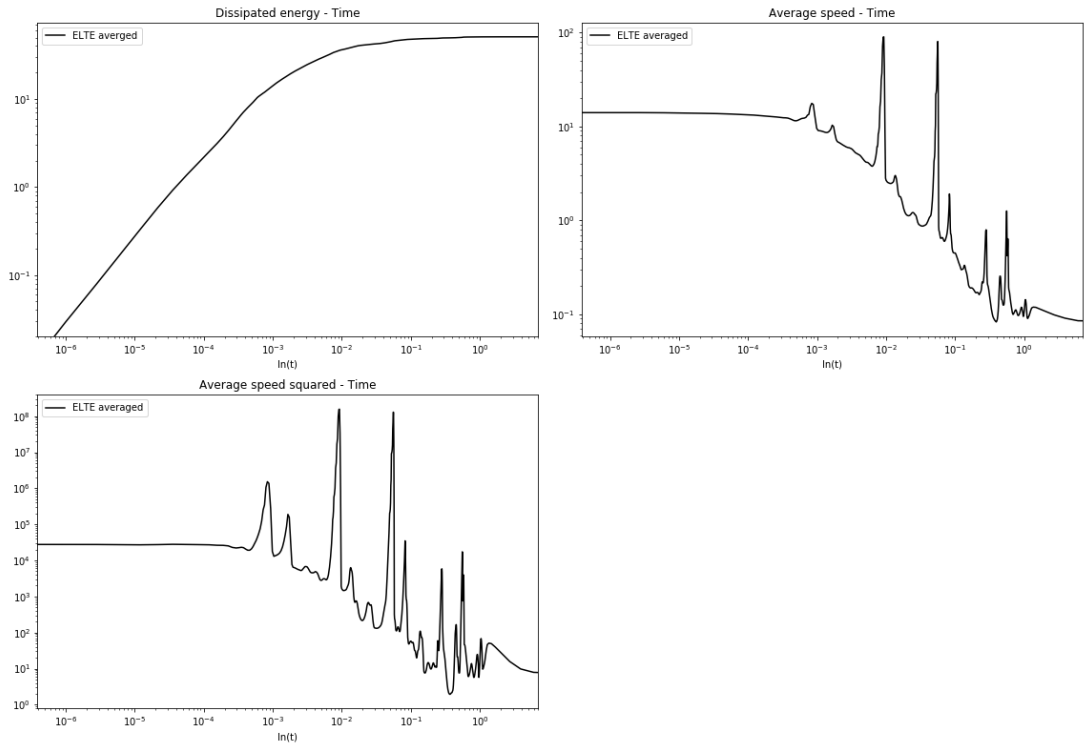
Averaging the output of different dislocation configurations

After having acquired the previous results I could move on with feeding the algorith with 10 different dislocation setups each containing 64 dislocations randomly generated in a $-0.5, 0.5$ grid in the x-y plane.
I plotted these results as I did previously:



Averaging the output of different dislocation configurations

Avaraging them took some effort since not all output files had the same row count since timesteps are defined, not time therefore the systems evolve for different but similair time periods. I took the shortest one and avaraged based on the length of that.



External fields

There is one external stress protocol defined, so far none was used in the previous simulation runs. A fixed rate external stress can be applied and its value defined as an input parameter.
The provied input file for these runs was an earlier output from an equilibrated system.

External field analysis

I ran two set-ups with external stress set to 0.1 and 0.01. Everything is dimensionless in the simulation so therefore they only have numeric number, but 0.1 is large and 0.01 is not so large.
I run the numerical analysis on these log files as well.

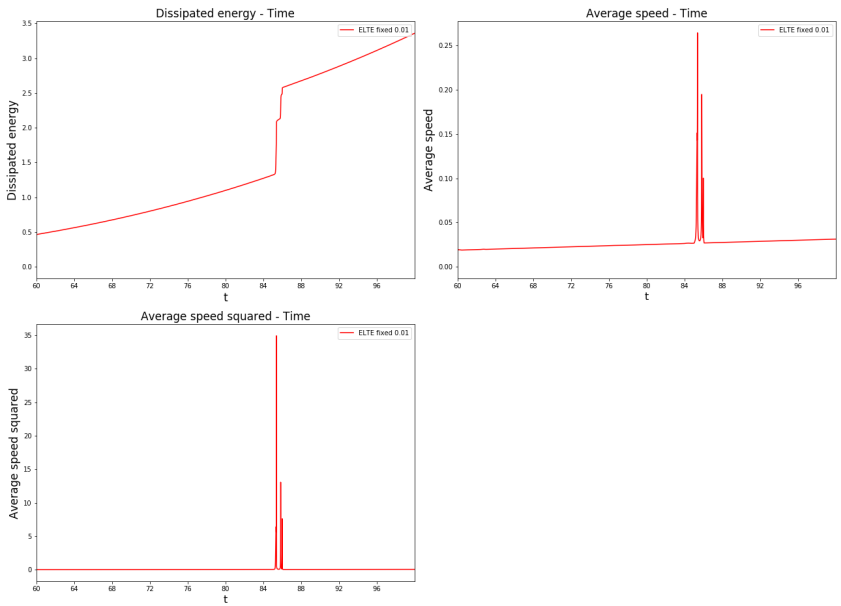


Figure: Fixed stress rate - 0.01

Conclusion

It can be seen that the system keeps on dissipating energy while time goes on on a fixed rate on log scale, therefore exponentially more on linear scale. Average speed and speed squared as well.

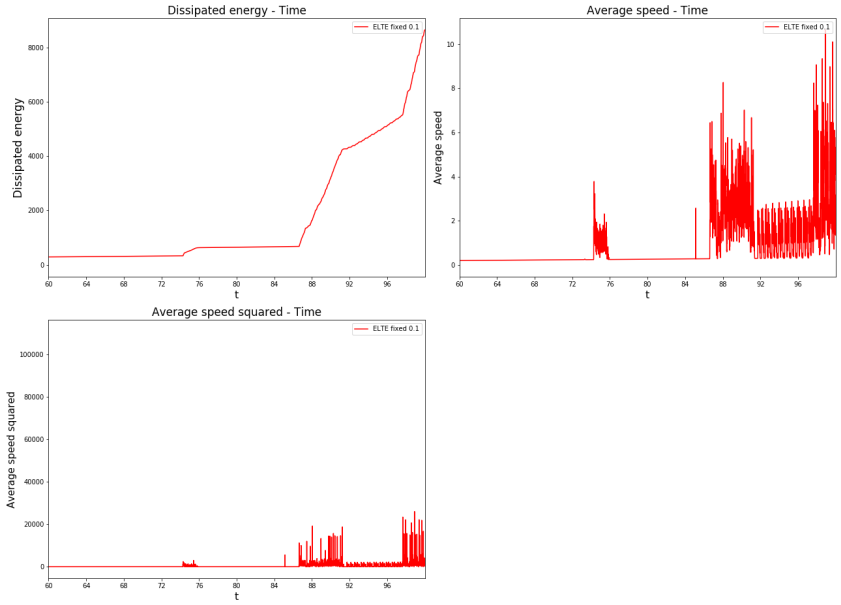


Figure: Fixed stress rate - 0.1

The interesting peaks can be the reuslt of the system starting to behave as a fluid due to the force and continually growing speeds.

Average distance test

Given a constant external stress the dislocations should get on average closer to each other. At each step I extracted the avarage of minimum pair distances. Meaning that at each step I calculated the minimum pair distances for each dislocation and output the avarage of that to the standard output.
I used the ELTE stress field, with a 0.01 external field applied and ran the simulation on 32, 64, 128 random dislocation sets.
I ran them through a simulation to relax the input system and then provied the randomly generated data as input to the external stress applied run.
The system containing 32 dislocations ran fastly for long maximum time but the one with 64 and 128 were slow to run for even a 100 limit simulation time.

Analysis

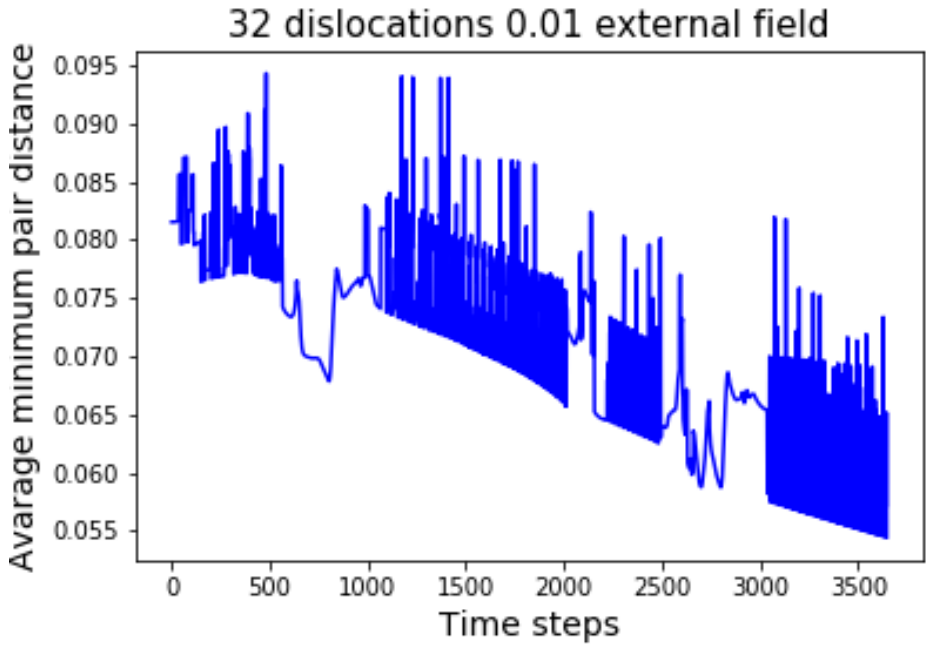


Figure: System with 32 dislocations

On this figure 6 it can be evaluated easilly that the average minimum distances are lowering while there are still these wiggly peaks.

Analysis

With bigger system sizes data grow fastly and time did not increase that much while this minimum pair distance lowered less significantly. Avalanche like behaviour!

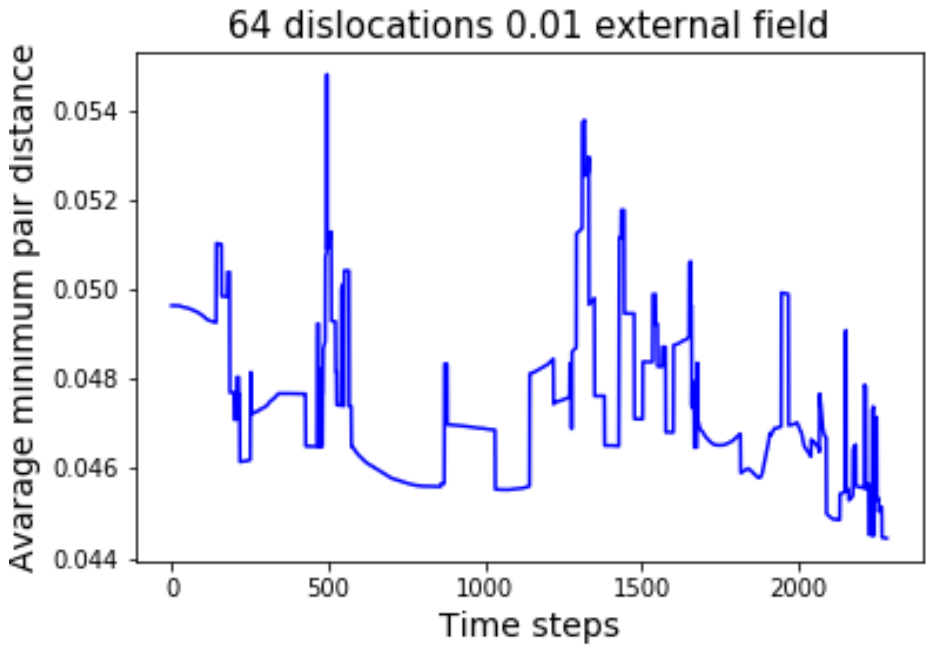


Figure: System with 64 dislocations

Analysis

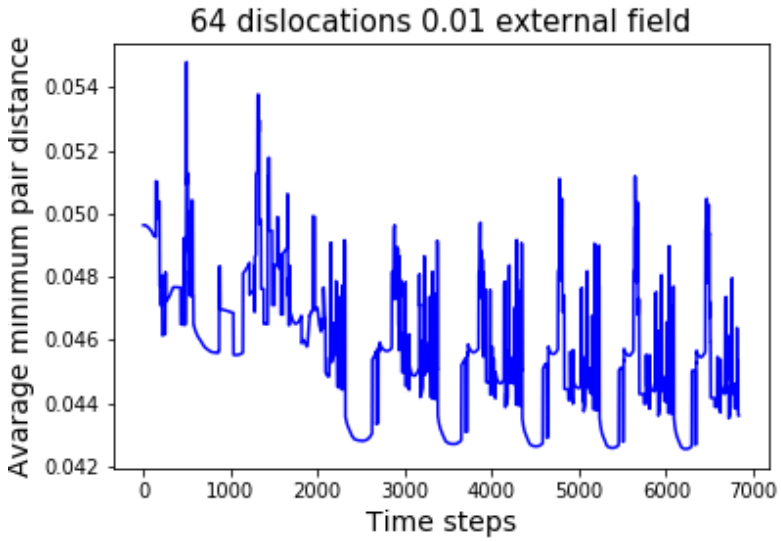


Figure: System with 64 dislocations

Where 8 is run for significantly longer (for twice the amount of original run time 100 provided). Lowering can be seen on these figures but with an expanding system size wiggliness is worsening.

Analysis

With a 128 system size:

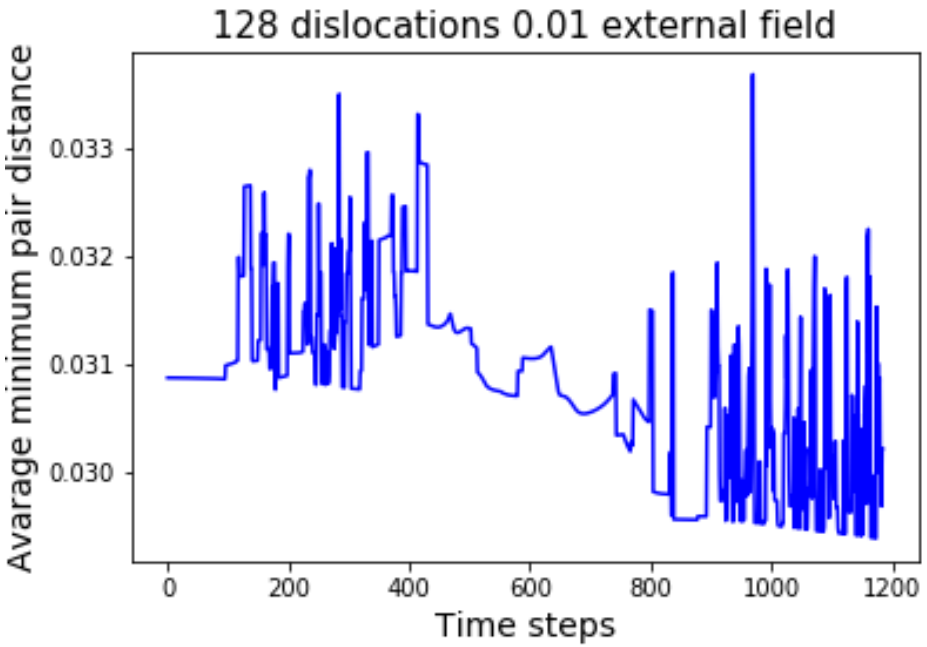


Figure: System with 128 dislocations

Final conclusion

- I added code and created visualizations
- Tested the code's expected behaviour
- Executed all necessary tasks
- Finished before due date