Simple Discrete Dislocation Dynamics Toolkit
Final report

Alex Olar

University of Eötvös Loránd

Contents

I. Basics **3**
I..1. Theory 3
I..2. Installation 3
I..3. Plotting the stress matrix - setup 3

II. Visualization **4**

III. Numeric analysis of different stress fields **5**
III..1 Analysis conclusion 6

IV. Averaged inputs analysis **7**
IV..1 Analysis conclusion 7

V. Experimenting with external stress **7**
V..1. Average distance test 8

VI. Final conclusions **9**

Abstract

During these weeks my task was to experiment with the SDDDST software package [1] that is a dislocation dynamics simulation library written in C++ by Gábor Péterffy. This few weeks gave me a basic understanding of dislocations and the simulation itself. I was able to visualize, calculate predefined and theoretical dislocation fields and modify the code base itself.

I. Basics

I..1. Theory

Dislocations are an important class of defect in crystalline solids. All real crystals contain imperfections, which may be point, line, surface or volume defects, and which locally disturb the regular arrangement of the atoms. Their presence can significantly modify the properties of crystalline solids describe the basic geometry of an edge and a screw dislocation line. All the atoms in a perfect crystal are at specific atomic sites (ignoring thermal vibrations). [2]

The two primary types of dislocations are edge dislocations and screw dislocations.

Mathematically, dislocations are a type of topological defect, sometimes called a soliton. Dislocations behave as stable particles: they can move around, but maintain their identity. Two dislocations of opposite orientation can cancel when brought together, but a single dislocation typically cannot disappear on its own. [3]

I..2. Installation

Installing the software is straightforward on Debian, only some additional packages are required for the process which can be found easily. 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:

```
mkdir build
cd build
cmake ..
make
```

In the built folder after executing the following command we get helpful documentation regarding the input parameters:

```
./sdddst --help
```

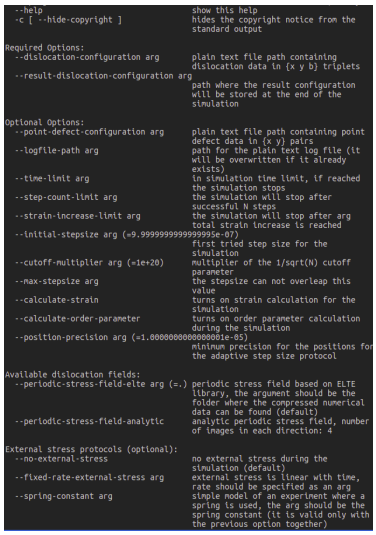


Figure 1: Argument documentation

I..3. Plotting the stress matrix - setup

I was able to extract the stress field around a dislocation in the origo. I extended the code with this functionality. The object oriented structure helped me to easily output the field to the standard file stream.

```
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";
 }

}
```

Where the function *xy(double x, double y)* calculates the field around the dislocation at coordinates i, j with a 0.005 resolution in a $-0.5, 0.5$ grid.

II. Visualization

The visualization was a hard task as it is not a piece of cake to visualize stress fields that are varying on a high scale. The stress field contains some enormous and many tiny numerical values therefore it must be considered during visualization how to handle this.

The *matplotlib* package provides a clean interface to visualize such 'images' on a symmetric logarithmic scale. This means that for a while the scale is linear but after a given threshold it becomes logarithmic, it is also symmetric since the fields very from negative to positive values.

I ran the code with different types of fields, such as the stress field code-named ELTE and several analytic fields. The analytic fields only differ in accuracy since they are calculated using N images.

The ELTE field looks the following way:

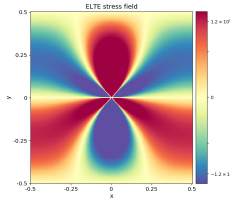
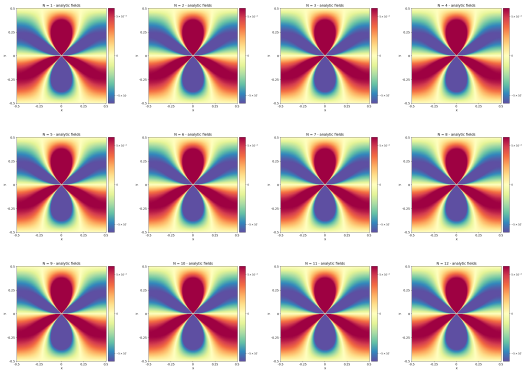


Figure 2: Stress field visualization

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



The difference doesn't really show on the images but getting the consecutive differences of images and displaying that one can achieve much better visualization of the fields:

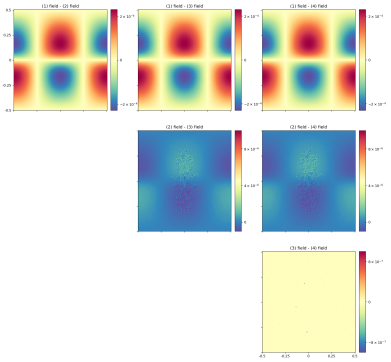


Figure 3: Difference of analytic fields

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.

Having plotted the difference of analytic fields I moved on with showing the differences of analytic and ELTE code-named stress fields.

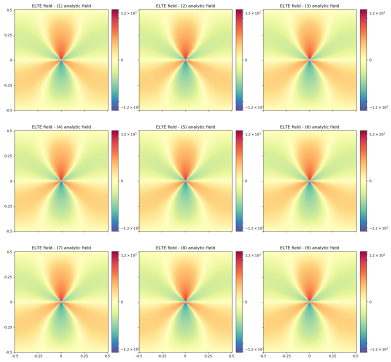


Figure 4: Difference with the ELTE field

From this plot it can be derived that they are equal almost everywhere but in the middle where they differ mostly.

III. Numeric analysis of different stress fields

Moving on to the log file that is produced after each simulation I had to create a random dislocation configuration with 64 dislocations to feed in for the numerical tests.

The log file contains the following information:

- simulation time
- number of successful steps
- number of failed steps
- worst error ratio squared
- average speed of the dislocations
- cutoff (used in the semi implicit scheme)
- order parameter
- value of the external stress
- computation time between the last two successful steps
- accumulated strain
- average v2
- energy of the system

Where I considered the average speed, average speed squared and energy variables in the function of time as relevant variables to test the system numerically.

The energy is better described as the energy loss of the system since the system equilibrates and loses energy.

I also saved the output dislocation file where I get the resulting dislocation coordinates but it is not interesting since it cannot be decided which point corresponds to which starting location.

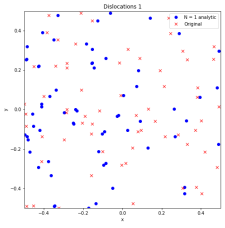


Figure 5: Starting and finishing dislocation location distribution

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 imagined 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

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

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

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

The off diagonal part is used in the simulation.

The previously seen ELTE stress field 4 which has a periodic boundary condition.

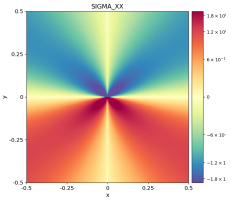


Figure 6: Theoretical σ_{xx} component

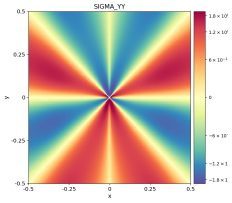


Figure 7: Theoretical σ_{yy} component

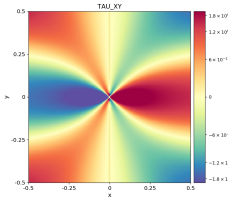
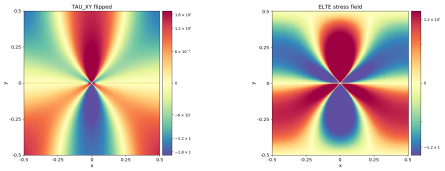


Figure 8: Theoretical τ_{xy} component

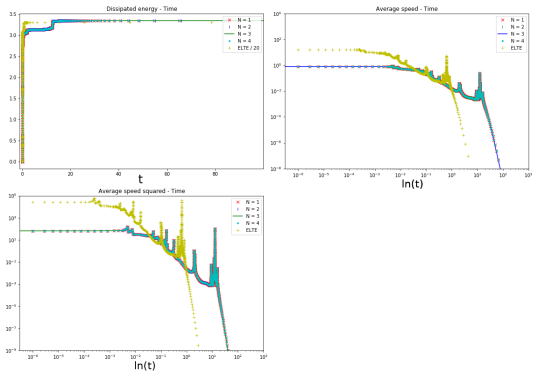
This resembles to the ELTE field as it should, since the off-diagonal components are were displayed previously by my visualization programs.



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

III..1. Analysis conclusion

Analyzing the log files and comparing them the following can be deduced:



- The ELTE field dissipates energy faster than the analytic fields which 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.

IV. Averaged inputs analysis

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

All the systems ran for different times. Due to this averaging was not straightforward at all I needed to find the lowest one and average until that one, which ran for ≈ 1033.7 time-steps defined by the simulation.

I plotted these results as I did previously:

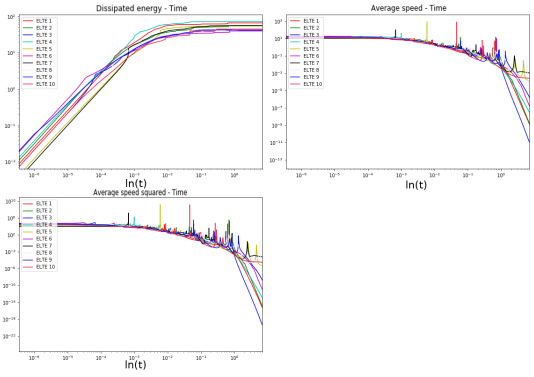


Figure 9: Each line is a different run with a different input file with the ELTE stress field applied

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

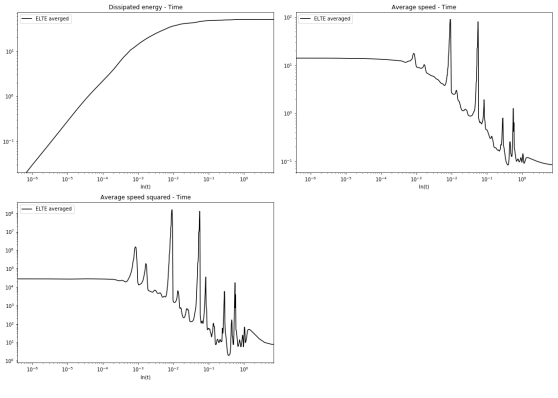


Figure 10: Ten averaged simulations with ELTE field applied

IV..1. Analysis conclusion

Generally it seems like the system doesn't depend much on the actual dislocations' and they all behave very similarly. Averaging more simulations would result in peaks disappearing on 10 and only the dissipating tendency could be seen.

V. Experimenting with external stress

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 provided input file for these runs was an earlier output from an equilibrated system.

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.

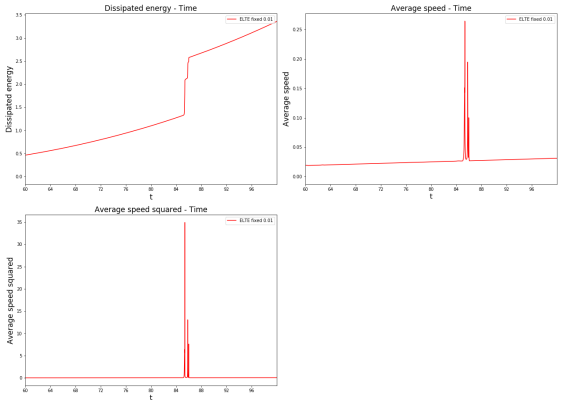


Figure 11: Fixed stress rate - 0.01

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.

Raising the external stress results in strange behavior.

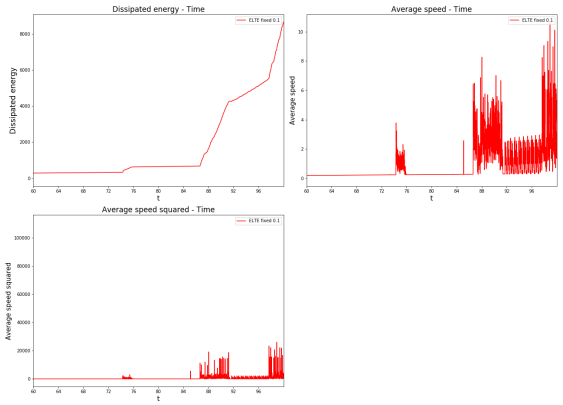


Figure 12: Fixed stress rate - 0.1

The interesting peaks can be the result of the system starting to behave as a fluid. I only show the regions on both figures which are interesting in this regerd.

V..1. Average distance test

Given a constant external stress the dislocations should get on average closer to each other. At each step I extracted the average of minimum pair distances. Meaning that at each step I calculated the minimum pair distances for each dislocation and output the average 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 provided the randomly generated data as input to the external stress applied run.

The system containing 32 dislocations ran fast for long maximum time but the one with 64 and 128 were slow to run for even a 100 limit simulation time.

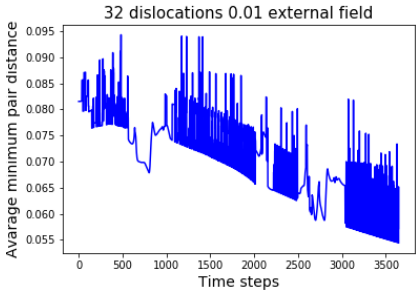


Figure 13: System with 32 dislocations

On this figure 13 it can be evaluated easily that the average minimum distances are lowering while there are still these wiggly peaks. These are the result of the following effect: some particles are repenting others are attracting each other due to the force they can get so close to each other that they 'jump through' each other

and start to repent/attract. When two or more dislocations reach this point these avalance like structures can be observed.

With bigger system sizes data grow fast and time did not increase that much while this minimum pair distance lowered less significantly.

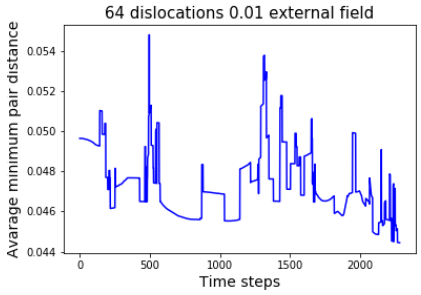


Figure 14: System with 64 dislocations

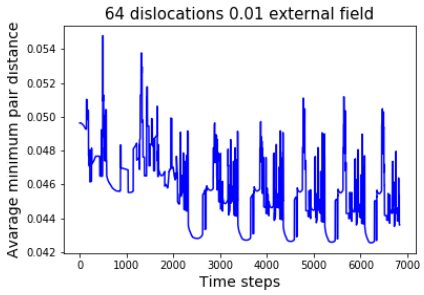


Figure 15: System with 64 dislocations

Where 15 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. The periodic condition can be evidently seen in 15 as particles repent each other but due to the periodic boundary condition they get close to each other again and again.

With a 128 system size the expected behavior cannot be evidently seen.

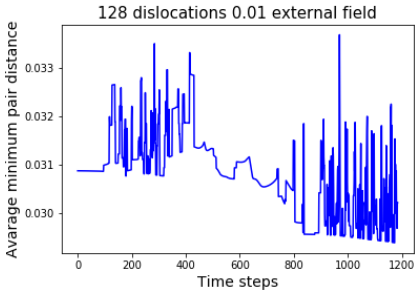


Figure 16: System with 128 dislocations

VI. Final conclusions

For these few weeks I got acquainted with the SDDDST package and its nooks and crannies. I created several visualizations and extended the code to acquire the required output. I also analyzed log files and tested the code's integrity. Not forgetting to mention that I did this in half the amount of the planned time.

I also contacted my supervisor several times in person and in mail as well. My progress and results were clear for all the time being and I always asked when I got stuck.

I consider my initial goal - finishing early successful and in general I consider this lab useful for those who weren't able to develop self discipline until now to be capable of meeting due dates.

References

[1] Gábor Péterffy. Sdddst - simple discrete dislocation dynamics simulation toolkit. 2018.

[2] D. Hull and D.J. Bacon. Introduction to dislocations. 2011.

[3] Dislocations. en.wikipedia.org/wiki/Dislocation.