Title

Subtitle

by

Mikkel Metzsch Jensen

THESIS

for the degree of

MASTER OF SCIENCE

Faculty of Mathematics and Natural Sciences
University of Oslo

Spring 2023

Title

Subtitle

Mikkel Metzsch Jensen

Title

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

Abstract

Abstract.

Acknowledgments

Acknowledgments.

Contents

List of symbols?

Maybe add list of symbols and where they are used like Trømborg.

Part I

Background Theory

Part II

Simulations

Chapter 1

Dataset study

1.1 Generating data

We generate the dataset by simulating various sheet cut configuratations under multiple combinations of normal load and stretch. For each configuration we sample 15 pseudo uniform (refer to relevant section here) strech values between zero and the rupture stretch found in the rupture test. The normal force is uniformly sampled in the range $[0.1, 10]$ nN. In total this gives 3×15 data points for each configuration. For the remaining parameters we use the values presented in the pilot study (see table **??**). We generate 68 configurations of the Tetrahedron pattern type, 45 of the Honeycomb type and 100 of the Random walk type which is shown in Appendix C?. A summary of the data points is given in table 1.1. The table shows that not all submitted data points "makes it" to the final dataset. This is due to the combination of our numerical procedure and small variations in the rupture stretch point. After performing the rupture test the simulation is restarted with a new substrate size corresponding to the measured rupture stretch limit and also with new random velocity and thermostat initializations values. The sheet is then stretched and checkpoints of the simulation state (LAMMPS restart files) are saved for each of the targeted stretch samples. However, if the rupture points arrives slightly early than syggested by the rupture test, some sampled stretch values might not get a corresponding checkpoint file. Thus, these data points are not included in the data set even though they ideally should have been noted as a rupture event. This could quite easily have been mittigated by a rewrite of that part of the code, but it was first discovered after the dataset had been created. However, the dataset still includeds 11.57 % rupture events and it most likely that the most cases with a lost rupturer event have a rupture event stored for the preeceding stretch value instead which captures the information of the sheet stretch limit on its own.

Table 1.1: Summary of the number of generated data points in the dataset. Due to slight deviations in the rupture stretch and the specific numerical procedure not all submitted simulations "makes it" to the final dataset. Notice that the Tetrahedon (7, 5, 2) and Honeycomb (2, 2, 1, 5) from the pilot study is rerun as a part of the Tetrahedon and the Honeycomb datasets seperately. In the latter datasets the reference point for the pattern is randomized and thus theese configurations is not fully identical. This is the idea behind the difference of 2 in the total sum.

Type	Configurations	Submitted data points	Final data points	Ruptures
Pilot study	3	270	261	25 (9.58 %)
Tetrahedon	68	3060	3015	391 (12.97 %)
Honeycomb	45	2025	1983	80 (4.03 %)
Random walk	100	4500	4401	622 (14.13 %)
Total	214 (216)	9855	9660	1118 (11.57 %)

1.2 Data analysis

In order to gain insight into the correlations between variables associated to the simulations we calculate the correlations coefficients between all variable combinations. More specific, we are going to calculate the Pearson

product-moment correlation coefficient (PPMCC) for which is defined, between data set X and Y, as

$$\mathrm{corr}(X,Y) = \frac{\mathrm{Cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{\langle (X-\mu_X)(Y-\mu_Y)\rangle}{\sigma_X \sigma_Y} \in [-1,1]$$

where $\mathrm{Cov}(X,Y)$ is the covariance, μ the mean value and σ the standard deviation. The correlation coefficients ranges from perfect negative correlation (-1) through no correlation (0) to a perfect positive correlation (1). The correlation coefficients is shown in figure 1.1

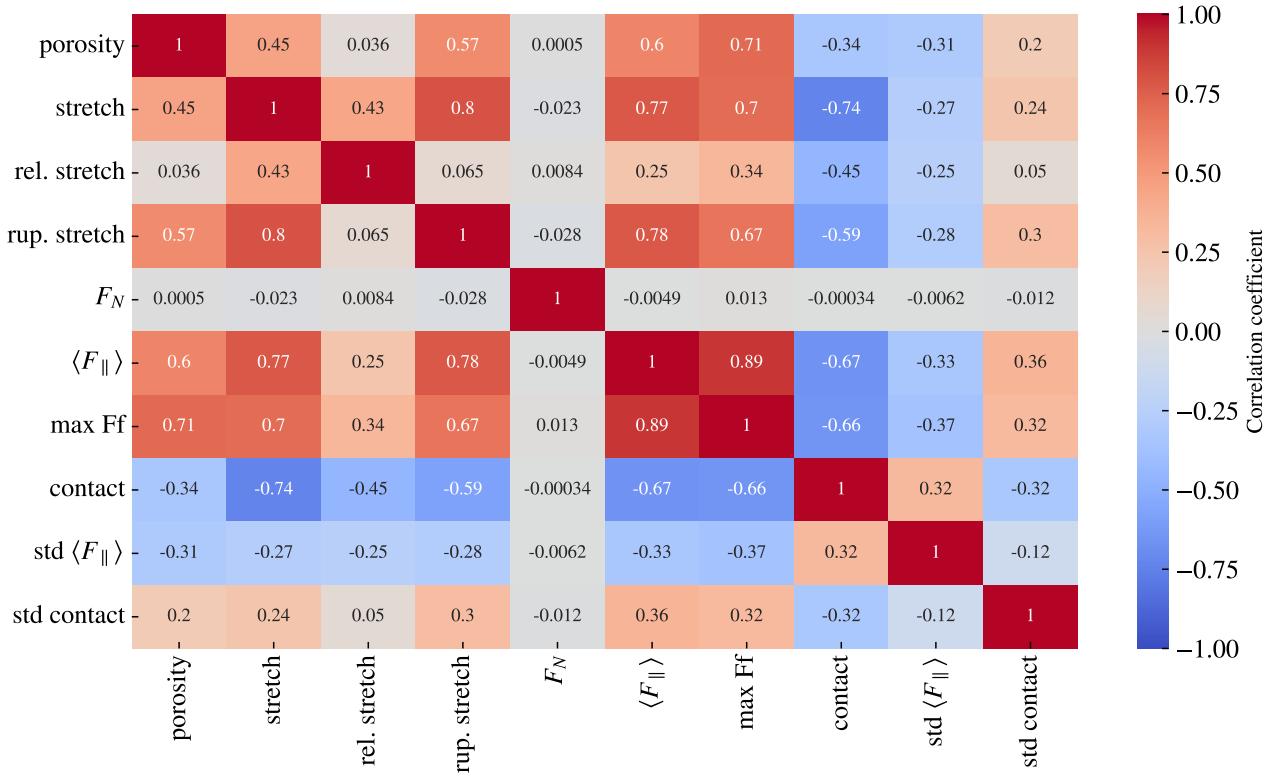

Figure 1.1: Pearson product-moment correlation coefficients for the full datset (see table 1.1).

From figure 1.1 we especially notice that the mean friction force $\langle F_\parallel \rangle$ has a signifciant positively correlation with stretch (0.77) and porosity (0.60) (void fraction). However, the relative stretch, which is scaled by the rupture stretch, has a weaker correlation of only 0.25 which indicates that it is the absolute stretch value that has the most significant impact on the friction force increase during stretching. This is further supported by the fact that the mean friction and the rupture stretch is also strongly positively correlated (0.78). From figure 1.1 we also observe that the contact bond count is negatively correlated with the mean friction (-0.67) and the stretch value (-0.74) which is consistent with the trend observed in the pilot study (figure **??** and **??**) of the contact decreasing with increasing stretch and mean friction. However, we must take note that the correlation coefficients is a measure of the strength and slope of a forced linear fit on the data. We clearly observed a non-linear relationship between stretch and mean friction for the tetrahedron and honeycomb pattern used in the pilot study (figure **??**) where the relationship was partwise characterized by a postive correlation for some stretch ranges and partwise negative correlation for other stretch ranges. Hence, interesting strong regime-specific correlations might not be accurately highlighted by the correlation coefficients shown in figure 1.1.

In figure **??** we have visualized the data (excluding the pilot study) for chosen pairs of variables on the axes. In addition to a visual confirmation of how the given correlations look in a 2D plot we also get a feeling for the coverage in various areas of the parameter space that we are eventually going to feed the neural network. The honeycomb pattern is spanning a significant larger range of stretch, contact and mean friction makes the data rather biased towards the Honeycomb pattern in thoose areas.

1.3 Properties of interest

From the Pilot study we found promising prospects regarding the achievement of a negative friction coefficient for certain kirigami cut configurations under the assumption of a system with coupled normal force F_N and stretch S. This stands as the main property of interest to explore further in the dataset. However, it is not obvious how one should quanity this in a rigorous manner. The friction coefficient is by our definition (see theory sec XXX) given as the slope of the friction vs. normal force curve. That is, for two data points $(F_{N,1}, F_{f,1}), (F_{N,2}, F_{f,2})$, $F_{N,1} < F_{N,2}$ we calculate the friction coefficient $\mu_{1,2}$ as

$$\mu_{1,2} = \frac{F_{f,2} - F_{f,1}}{F_{N,2} - F_{N,1}} = \frac{\Delta F_f}{\Delta F_N}$$

In the pilot study it became clear that the effects on the friction force when varying the normal force were neglible in comparison to that of varying the stretch. Thus, we assume $F(F_N, S) = F(F_N) + F(S) \approx F(S)$ such that for a coupling between stretch and normal force $F_N = RS + C$ with coupling ratio R we get

$$\mu_{1,2}(S_1, S_2) = \frac{\Delta F_f(S_1, S_2)}{R(S_2 - S_1) + C} \underset{\sim}{\propto} \frac{\Delta F_f(S_1, S_2)}{\Delta S},$$

where the last proportionality is achieved by assumping $C \ll 0$. From the above reasoning we practically exchanged F_N with S in the formula for calculating the friction coefficient. This means that we should look at the friction vs. stretch curve for which a steep decline indicates a strong negative coefficient for the assumed coupled system which we will work with later. By definition, the minimum slope value would give the lowest friction coefficient. However, this is also a uncertain measure as two data points with a small difference in stretch can easily achieve extreme slope values by very little error or noise in friction value. Even if such an extreme slope existed locally it would not be very interesting since the negative coefficient woud only be valid in that small range. We are essentially looking for the lowest coefficient in the longest possible stretch range. One way to capture a bit from both of these concepts is by looking for the biggest drop in friction when increasing stretch. By scanning over all local maxima we evaluate the difference in friction to all of the following local minema. The biggest drop will serve as our indicator for good prospects of showcasing a negative friction coefficient. By doing so we do not guarantee a monotonic decrease of friciton in the range that got the biggest drop, but we believe that this is a good evaluation function to target interesting properties in general.

In addition to the biggest forward drop in friction we also look a minimum abnd maximum friction along with the difference between theese extrema.

Table 1.2: Interesting properties

Tetrahedron	Configuration	Stretch	Value [nN]
Min F_{fric}	$(3, 9, 4)$	0.0296	0.0067
Max F_{fric}	$(5, 3, 1)$	0.1391	1.5875
Max ΔF_{fric}	$(5, 3, 1)$	$[0.0239, 0.1391]$	1.5529
Max drop	$(5, 3, 1)$	$[0.1391, 0.1999]$	0.8841

Honeycomb	Configuration	Stretch	Value [nN]
Min F_{fric}	$(2, 5, 1, 1)$	0.0267	0.0177
Max F_{fric}	$(2, 1, 1, 1)$	1.0654	2.8903
Max ΔF_{fric}	$(2, 1, 5, 3)$	$[0.0856, 1.4760]$	2.0234
Max drop	$(2, 3, 3, 3)$	$[0.5410, 1.0100]$	1.2785

Random walk	Configuration	Stretch	Value [nN]
Min F_{fric}	12	0.0562	0.0024
Max F_{fric}	96	0.2375	0.5758
Max ΔF_{fric}	96	$[0.0364, 0.2375]$	0.5448
Max drop	01	$[0.0592, 0.1127]$	0.1818

The stretch profiles for all the configurations are shown in appendix **??**.

1.4 Machine learning

Choices to make

- Learning rate: increasing LR to locate minimum in loss gradient (fastest learning) (LR range test)
- Architecture: VGGNet staircase type search
- Optimizers: Just try some different ones and look at convergence
- weight decay

Suggestion for hyperparameter tuning

1. Select complexity range of VGGNet staircase type architectures
2. Start with Adam optimizer at default settings
3. Perform LR range test to get information of a suitable LR and choose one that will work for most of the architectures.
4. Perform architecture grid search over depth and start number of channels (complexity). Choose best architecture from this.
5. Additionally add weight decay to further optimize learning for that architecture choice. Perhaps cyclic learning rates.

Start by mentioning the related articles using a VGGNet tyep network (and maybe also their learning rates).

Either submit with best lr for each or choose something that works for all. 0.0005 seemed as a good middle ground but the complex networks crash with these learning rates. It looks like 0.0001 (like Hanakata used) will do the job for all.

1.5 Accelerated Search

Having a network model that can predict friction force for a given configuration are able to search for some desired properties. Low and high friction and maximal negative friction coefficients

Here we pursue two different approaches for finding

1. Generate an enlarged dataset and run it through the ML model
2. Genetic algorithm

1.5.1 Markov-Chain Accelerated Genetic Algorithms

1.5.1.1 Talk about traditional method also?

1.5.1.2 Implementing for 1D chromosone (following article closely)

We have the binary population matrix $A(t)$ at time (generation) t consisting of N rows denoting chromosones and with L columns denoting the so-called locus (fixed position on a chromosome where a particular gene or genetic marker is located, wiki). We sort the matrix rowwise by the fitness of each chromosono evaluated by a fitness function f such that $f_i(t) \leq f_k(t)$ for $i \geq k$. We assume that there are a transistion probability between the current state $A(t)$ and the next state $A(t+1)$. We consider this transistion probability only to take into account mutation process (mutation only updating scheme). During each generation chromosones are sorted from most to least fitted. The chromosone at the i-th fitted place is assigned a row mutation probability $a_i(t)$ by some monotonic increasing function. This is taken to be

$$a_i(t) = \begin{cases} (i-1)/N', & i-1 < N' \\ 1, & \text{else} \end{cases}$$

for some limit N' (refer to first part of article talking about this). We use $N' = N/2$. We also defines the survival probability $s_i = 1 - a_i$. In thus wau a_i and s_i decide together whether to mutate to the other state (flip binary) or to remain in the current state. We use s_i as the statistical weight for the i-th chromosone given it a weight $w_i = s_i$.
Now the column mutation. For each locus j we define the count of 0's and 1's as $C_0(j)$ and $C_1(j)$ resepctively. These are normalized as

$$n_0(j,t) = \frac{C_0(j)}{C_0(j) + C_1(j)}, \quad n_1(j,t) = \frac{C_1(j)}{C_0(j) + C_1(j)}.$$

These are gathered into the vector $\mathbf{n}(j,t) = (n_0(j,t), n_1(j,t))$ which characterizes the state distribution of j-th locus. In order to direct the current population to a preferred state for locus j we look at the highest weight of row i for locus j taking the value 0 and 1 respectively.

$$C_0'(j) = \max\{W_i | A_{ij} = 0;\ i = 1, \cdots, N\}$$
$$C_1'(j) = \max\{W_i | A_{ij} = 1;\ i = 1, \cdots, N\}$$

which is normalized again

$$n_0(j,t+1) = \frac{C_0'(j)}{C_0'(j) + C_1'(j)}, \quad n_1(j,t+1) = \frac{C_1'(j)}{C_0'(j) + C_1'(j)}.$$

The vector $\mathbf{n}(j,t+1) = (n_0(j,t+1), n_1(j,t+1))$ then provides a direction for the population to evolve against. This characterizes the target state distribution of the locus j among all the chromosones in the next generation. We have

$$\begin{bmatrix} n_0(j,t+1) \\ n_1(j,t+1) \end{bmatrix} = \begin{bmatrix} P_{00}(j,t) & P_{10}(j,t) \\ P_{01}(j,t) & P_{11}(j,t) \end{bmatrix} \begin{bmatrix} n_0(j,t) \\ n_1(j,t) \end{bmatrix}$$

Since the probability must sum to one for the rows in the P-matrix we have

$$P_{00}(j,t) = 1 - P_{01}(j,t), \quad P_{11}(j,t) = 1 - P_{10}(j,t)$$

These conditions allow us to solve for the transition probability $P_{10}(j,t)$ in terms of the single variable $P_{00}j,t$.

$$P_{10}(j,t) = \frac{n_0(j,t+1) - P_{00}(j,t) n_0(j,t)}{n_1(j,t)}$$
$$P_{01}(j,t) = 1 - P_{00}(j,t)$$
$$P_{11}(j,t) = 1 - P_{10}(j,t)$$

We just need to know $P_{00}(j,t)$. We start from $P_{00}(j,t=0) = 0.5$ and then choose $P_{00}(j,t) = n_0(j,t)$

Summary

1.6 Summary and conclusion

1.7 Outlook / Perspective

- What did we not cover?
- What kind of further investigations does this study invite?

Things to include here

- Could be valuable to spend more time on the validation of the MD simulations. How does material choice and potential effects the results. How realistic is the simmulations?
- Are there any interesting approaches for compressed kirigami structures?
- How does these results scale? I imagaind that the nanomachine systems should be applied in small units to avoid scaling problems, but in general I could spend way more time on the scaling investigation.
- Since the normal force is applied at the pull blocks the normal force distribution changes from the sides more towards and even distirbution as the sheet is put under tension (stretched). If we imagined a sheet for which the center part was either a different material or had some kind of pre-placed asperity on it, could we then exploit this force distirbution to get exotic properties as well? By studying this we might get a clearer understanding of what is the cause of my results.
- Possibility to study hysteresis effects. Maybe the frictional behvaiours change significantly through repeated cycles of stretch and relax.

Appendices

Appendix A

Appendix B

Appendix C

Bibliography